University of the Philippines Manila
College of Arts and Sciences
Department of Physical Sciences and Mathematics

A Personal Mobile Time Management and Motivation Application

A special problem in partial fulfillment
of the requirements for the degree of
Bachelor of Science in Computer Science

Submitted by:

Brendan D. de Guzman
May 2016

Permission is given for the following people to have access to this SP:

Available to the general public	Yes
Available only after consultation with author/SP adviser	No
Available only to those bound by confidentiality agreement	No

ACCEPTANCE SHEET

The Special Problem entitled "A Personal Mobile Time Management and Motivation Application" prepared and submitted by Brendan D. de Guzman in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance.

Ma. Sheila A. Magboo, M.Sc.
Adviser

EXAMINERS:

	Approved	**Disapproved**
1. Gregorio B. Baes, Ph.D. (*candidate*)		
2. Avegail D. Carpio, M.Sc.		
3. Richard Bryann L. Chua, M.Sc. (*cand.*)		
4. Perlita E. Gasmen, M.Sc. (*candidate*)		
5. Marvin John C. Ignacio, M.Sc. (*cand.*)		
6. Vincent Peter C. Magboo, M.D., M.Sc.		

Accepted and approved as partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.

Ma. Sheila A. Magboo, M.Sc.
Unit Head
Mathematical and Computing Sciences Unit
Department of Physical Sciences

and Mathematics

Marcelina B. Lirazan, Ph.D.
Chair
Department of Physical Sciences
and Mathematics

Leonardo R. Estacio Jr., Ph.D.
Dean
College of Arts and Sciences

Abstract

Time management is the ability to plan and control how you spend the hours in your day to effectively accomplish your goals. Some problems encountered by someone managing their time are underestimating how long a task can be accomplished, even if the task has been done before, and the non-compliance of an individual in their plan on managing their time. This project aims to help solving these problem through the creation of an application that reminds and map estimated start and end times against the actual start and end times of a task. With this application, users can map and see if their estimated time correspond to the actual time a task is done. The users can then adjust accordingly based on the monitored results.

Keywords: time management, motivation, android, reminder system, calendaring

Table of Contents

List of Figures

List of Tables

I. Introduction

A. Background of the Study

People now are living in a fast-paced world where every hour, every minute and seconds count. In some ways, the people need to be able to adapt and abreast themselves with the ever changing world. Fortunately, people nowadays have a faster instrument available to them when it comes to gathering information and that is the Internet. Even if the Internet is accessible to everybody, people are still looking for ways to make life a little better for them. There are still applications that needs to be developed or enhanced to cater some of our needs.

Time management is the ability to plan and control how you spend the hours in your day to effectively accomplish your goals. [1]. Some of the skills involved in managing time are planning for the future, setting goals, prioritizing what goals to accomplish, and monitoring the delegation of time to the tasks.

Multitasking is said to damage your brain, kill your performance, and impair your cognitive control, as a Stanford study suggested [2]. 100 students were put through a series of three tests. In each of the tests, the subjects are split into two groups: the group who regularly do a lot of media multitasking and those who dont. One of the experiments consists of showing the groups sets of two rectangles alone or surrounded by two, four or six blue rectangles. The configurations are flashed twice, and the participants determine if the two red rectangles in the second frame differ in position as opposed to the first frame. The blue rectangles are to be ignored. The heavy multitaskers performed horribly due to being distracted by the blue rectangles as opposed to the low multitaskers who had no problem answering. The second test, consisting of showing a sequence of alphabetical letters, also showed that the heavy multitaskers did not a poor job in remembering if a letter made a repeating appearance as opposed to the low multitaskers. A third test, consisting of images of letters and numbers where the subjects must focus on, also showed underperformance of heavy multitaskers. The researchers then concluded that heavy multitaskers were shown to underperform than the light multitaskers in terms of memory retention, focus, and filtering of irrelevant information. This is due to the intake of information presented to heavy multitaskers that they accept all information, may it be relevant or not.

Effective time management requires people to analyze their workload, assign priorities, and maintain focus on productive endeavors. Those people who are excellent time managers can eliminate distractions and enlist support from others to help accomplish their goals. They focus on the most important and time sensitive tasks and limit the amount of time wasted on non-essential duties.

Time management can be mistaken for time tracking. People can track time on their activities and everything they do for some time, but all it do was provided them was a number on how much time has been spent. And then they stop tracking the time they spent because they have not realized any positive changes. They have set aside a few events or activities and prioritized some other urgent activity, but they have not really managed anything. Time management is about making changes to the way you spend your time [3].

Students, professionals, and executives wherever field you came from, time is of the essence in ones daily task. In a corporate world, employees who manage their time well are more productive, more efficient and more likely to meet deadlines [4].

Motivation can be defined as the driving force behind all the actions of an individual [5]. It is based on the emotions and achievement-related goals of a person. Needs and desires of a person both provide motivation and impact to an individuals behavior. Motivational quotes and messages provide motivation and inspiration to many people, especially when things get tough. People who feel motivated and inspired by motivational quotes find the messages to be more resonant than those who do not find simple phrases and sayings to be particularly meaningful, says psychologist and motivation expert Jonathan Fader [6].

B. Statement of the Problem

Time management is one area in which we are having difficulty. A reason that can be seen as to why time management is difficult is due to the planning fallacy, wherein something that occurs when people underestimate how long it will take to finish a task, even if the task have been performed before [7].

For effective time management, you have to apply a time management tool or application that will help you see where changes can and should be made. The first step of time management is to analyze how you actually spend your time so you can determine what changes you want to make in relation to the activities being prioritized and spending time on [8].

Unfortunately, there are available systems that cater to time tracking but only a few when it comes to time management. Some of these applications, such as Remember The Milk and Schedule Planner Classic, require a paid license or a subscription in order to access some of the functionalities of the application such as sending reminders to the user and synchronizing with Google Calendar for shared calendars.

Most smartphones also contain a built-in calendar application and reminder system. However, this application does not contain a mapping ability that allows the user to compare the estimated starting times and ending times of a task that the user entered and the actual starting time the user starts the task and the actual ending time the user finishes the task.

C. Objectives of the Study

The objective of this application is to provide a time management and motivation application that would run in a mobile smartphone running the Android operating system and aid the user to manage, monitor, and be notified of the different tasks set by the user. The application would also provide motivations in the form customized text motivations set by the user. This application aims to promote efficient time management among the users and a better understanding through monitoring of tasks.

A task is defined to be anything that the user enters to be reminded of and for monitoring of the task's time.

A category is defined as a class or group where a task can fall and be grouped to. These categories can be user-defined. Some examples of the default categories are Work, Home, Leisure, and Misc.

The system has the following functionalities:

1. Allow the user to:
    - (a) Add a task and scheduled time of the task to the list
        - i. Enable multitasking by setting tasks in the same time and setting priority levels to each task
        - ii. Set task name
        - iii. Set priority level of the task
        - iv. Set category of the task
        - v. Set estimated time of completion of task
        - vi. Set estimated starting and ending dates of task
        - vii. Set alarm or reminder time
        - viii. Set frequency of reminder

    ix. Set notes or comments
    x. Tag a task as publishable to Google Calendar
    xi. Set task name to appear in Google Calendar

  (b) Edit current tasks in the list (priority, time of completion, alarm, frequency of reminder, notes)

  (c) View current task

  (d) Delete tasks in the list

  (e) Search for completed tasks

    i. Search for the name / category of the task
    ii. Search via priority level of the task
    iii. Search for the date range of the task

  (f) Add / delete custom text motivations

  (g) View the dashboard for the following

    i. New tasks
    ii. Delayed tasks
    iii. High priority tasks
    iv. Upcoming tasks

  (h) View a specific categorys statistics

    i. View a specific categorys average time
    ii. View all tasks under a specific category

  (i) View a specific tasks statistics

    i. View a specific tasks average time of differences between estimated and actual start and end time
    ii. View task details and occurrences

  (j) Add, edit, and delete user-defined categories

2. Allow the system to:

(a) Notify the user for an upcoming or imminent scheduled task

(b) Notify the user with a text motivation

(c) Synchronize the users personal schedule with Google Calendar upon connection to Gmail account as long as online (including data connection)

(d) Push schedule from the personal time management tool to Google Calendar

D. Significance of the Project

Time management is important, especially if the user is engaged in multiple jobs or have many simultaneous activities that needed to be attended to. With the use of the system, the user is able to list all tasks at hand, prioritize tasks needed to be done, and share tasks scheduled for a certain time through Google Calendar.

The system will allow the user to input the tasks required to be done and set the time when they are due, the task priority, category, estimated time of completion, alarm or reminder time, frequency of reminder, and some notes or comments.

The system also allows the monitoring of activities through the result and statistics that will be available to the user such as average time difference between estimated/planned and actual time, the lists of ongoing, completed, and delayed tasks.

The system will also remind the user of new entered tasks, high priority tasks, upcoming tasks, and delayed tasks through the dashboard. The system will also notify the user through notifications.

The tasks entered can also be added by the user to their Google Calendar, which is connected through their Google accounts.

E. Scope and Limitations

1. The system only considers the tasks entered by the user.
2. The system is only limited to scheduling tasks, notifying the user of specific tasks, generating statistics, and synchronizing with Google Calendar.
3. The system only provides motivation in the form of text sent as notification.
4. The system will generate the statistics based on the categories defined and the tasks are categorized on.
5. The system will only use the default ringtone and vibration for notification
6. The system will only push tasks to Google Calendar when there is an internet connection.

F. Assumptions

1. The user inputs the tasks they need to manage on.
2. The user is knowledgeable to categorize the tasks they entered.
3. The user has logged in his Google account in the phone where the application is installed.

II. Review of Related Literature

Time management is currently being an interesting field to conduct research on, especially during our time when there are many things to be done at once and in a fast manner. There are studies conducted about time management in relation to other things that show how time management provide a positive impact to productivity.

Jackson outlined some key time management steps to consider in creating a realistic time management plan [9]. The steps outlined were: goals, organization, delegation, relaxation, and ditching guilt. Each of these steps contribute largely to time management. Setting goals is critical to personal success by setting priority to different tasks needed to be done. Getting organized provide a stress-free environment by organizing stuff and organizing time. Delegating tasks to others, within reason, can also be a productive move for others may be effective in doing another task. Relaxing is needed for recharging and finding renewed strength to face the tasks needed to be done. Ditching guilt is also needed, for guilt brings anxiety and stress that may hinder in productivity and managing ones time. Some hindrances to time management were also mentioned, such as procrastination and perfection. Some hindrances to time management were also mentioned, such as procrastination and perfection.

The investigating study done by Langa yielded results, although not to be generalized for all students, that there is the necessity for students to allocate a significant amount of hours to individual study so as to achieve good academic results [10]. An optimum time management plan should be made known to the students by the teachers to help in shaping the students competences targeted by study programs, giving them the advantage in integrating into the labor market.

A research was made by Nadinloyi, Hajloo, Garamaleki, and Sadeghi regarding the effectiveness of time management training to the academic time management of students [11]. There are two groups of students, one which received a time management training while another group who did not. The results showed that the skill on academic time management of the students belonging on the group who received a time management training was better than those in the control group.

A case study done by Adebayo examined the impact of time management on students academic performance in the higher institutions, specifically in Ekiti State in Nigeria [12]. The study showed that there is a significant relationship between time management and academic performance, where students need proper use of time to effectively perform various tasks and assignments. The study also showed that there is a significant relationship between procrastination and academic performance.

The relationship between time management, perceived stress, gender, and academic achievement among United Arab Emirates college students was studied in the study of Al Khatib [13]. The results have shown that higher time management and lower perceived stress were associated with higher academic achievement. Time management, however, was the most significant predictor of academic achievement based on the study. The study emphasizes the role of time management in relation to academic achievement. It was implied that those unable to manage their time well are on the risk of underachievement.

Improvement in time management capacities of principals is seen as a worthwhile strategy for increasing focus on instructional leadership and pursuing school improvement in the study of Grissom, Loeb, and Mitani [14]. The study found out that school principals that possess better time management skills allocate more time to managing instruction to their schools. Other findings of the study include the association between increased student test score growth in math of students and time management and rating of high school principals performance are more positive when they have better time management skills.

A study, done by MacCann, Fogarty, and Roberts, was conducted about time management in relation with part-time and full-time community college students [15]. Time management has been a mediator of the relationship between conscientiousness and students academic achievement. The study demonstrated that conscientiousness is as important in community colleges as it is in high schools and universities. The study also showed that time management is a significant mediator for part-time students but not for full-time students. This suggested that non-cognitive constructs, such as time management, may be more critical for non-traditional students.

The tendency of procrastination among individual and collaborative tasks on an online environment was the focus of the study of Gafni and Geri [16]. The results that were gathered from an online discussion board involving 120 MBA students showed that individual tasks, compulsory or voluntary, were finished on time. However, those part of a collaborative task were delayed for the last three weeks of the semester for compulsory tasks and the voluntary tasks were not completed at all.

A study by Yang, Xu, and Zhu has found that time management disposition can moderate the relationship between media multitasking and psychological well-being [17]. Time management was hypothesized and seen as positively associated with wellbeing and negatively correlated with media multitasking. The results indicated that time management has made individuals more sensitive to the negative effect of media multitasking on wellbeing.

A prototype web-based academic time management system, created for the School of Information Technology of Universiti Utara Malaysia, has showed significant improvement in time management after a period of two months of evaluation [18]. The prototype system, developed to improve the productivity and quality of works in the School of Information Technology in Universiti Utara Malaysia, was comprised of three modules. These modules were the Academician Information System (which contains the organized information related to academic staffs), the Committee System (a subsystem used for school committee services), and the Available Time System (a subsystem used for accessing, arranging, and providing an available time for any activity).

III. Theoretical Framework

1. Time Management

   The modern concept of time management refers to the act of planning the amount of time you spend on which activities [19]. It has become a subject of the management field to address the goal of having an increased productivity, especially among workers in the industry where work output may be hard to measure. Effective time management has been defined to be the process of setting goals, prioritizing them, deciding how much time to allocate to different tasks, adjusting to changes, revisiting the goals and priorities set, and observing the results after.

   Personal time management is loosely defined as managing our time to waste less of it on doing the things we have to do so we have more to do the things we want to do [3]. Time management is often presented to be a set of skills for people to be more organized, more efficient, and happier in life. Skills included in personal time management are goal setting, planning, prioritizing, decision-making, delegating, and scheduling of the different goals and activities for the day. Many people find that using time management tools, such as Personal Information Managers and phone applications, help in managing their time more efficiently.

2. Reminder System

   A reminder system is a system used to prompt or aid the memory. The system can be a computerized reminder, color coding, telephone calls, or devices such as a letter and postcard [20].

   A reminder system is used to better keep track of things instead of relying on the users memory [21]. The system deals in making the user remember activities and tasks that needed to be attended to such as appointments, meetings, paper works, deadlines, etc. Reminder systems are most effective when used consistently, limited to a certain number as to minimize clutter, and when taken advantage and incorporated to the users environment and daily routines.

3. Smartphone

   A smartphone is defined as a cellular telephone with an integrated computer and other features not associated with telephones, such as an operating system, Web browsing, and the ability to run software applications [22].

   Some of the suggested requirements for a cellular telephone to be designated as a smartphone are a recognized mobile operating system, internet connectivity, a mobile browser, hardware and/or software-based QWERTY keyboard, wireless synchronization with other devices, ability to download applications and run them independently, ability to run multiple applications simultaneously, having a touchscreen display, and can be connected using Wi-Fi. In comparison to standard cellular telephones, the smartphones are more powerful and contains more features.

4. Android OS

   The Android OS is an open source operating system primarily used in mobile devices [23]. It was initially developed by Android Inc. and was purchased by Google in 2005. The Android code was based on the modified version of the Linux kernel version 2.6 and was released by Google under the Apache license, which is a free software and open source license. The Android OS consists of numerous Java applications and core libraries running under the Java-based object oriented application framework and the Dalvik Virtual Machine. Due to constraints in terms of processor speed and memory, Dalvik is important for the Android OS to run in mobile devices.

5. Google Calendar

   Google calendar is a free Internet calendar that lets you keep track of your own events and share your calendars with others [24]. It used in conjunction with the Google account to sync with all devices where the Google account is logged in, may it be a smartphone or a desktop computer. The Google calendar is said to be the ideal tool for managing personal and professional schedules.

Google Calendar API can be used to find and view calendar events. It is used to achieve deeper integration with the Google Calendar. By using the Calendar API, mobile and web applications can create, display, or synchronize with the Google Calendar data [25].

IV. Design and Implementation

1. Use Case Diagram

The figure below shows the overall view of the system functionality. The general and only user of the system is the mobile user, since the application is a personal time management and motivation application. The user can then add, edit, delete, view and search tasks. The user can also view his list of tasks, view the dashboard for reminders, and view statistics related to a specific category of tasks. The user can also add and delete customized text motivations and add, edit, and delete the user-defined categories.

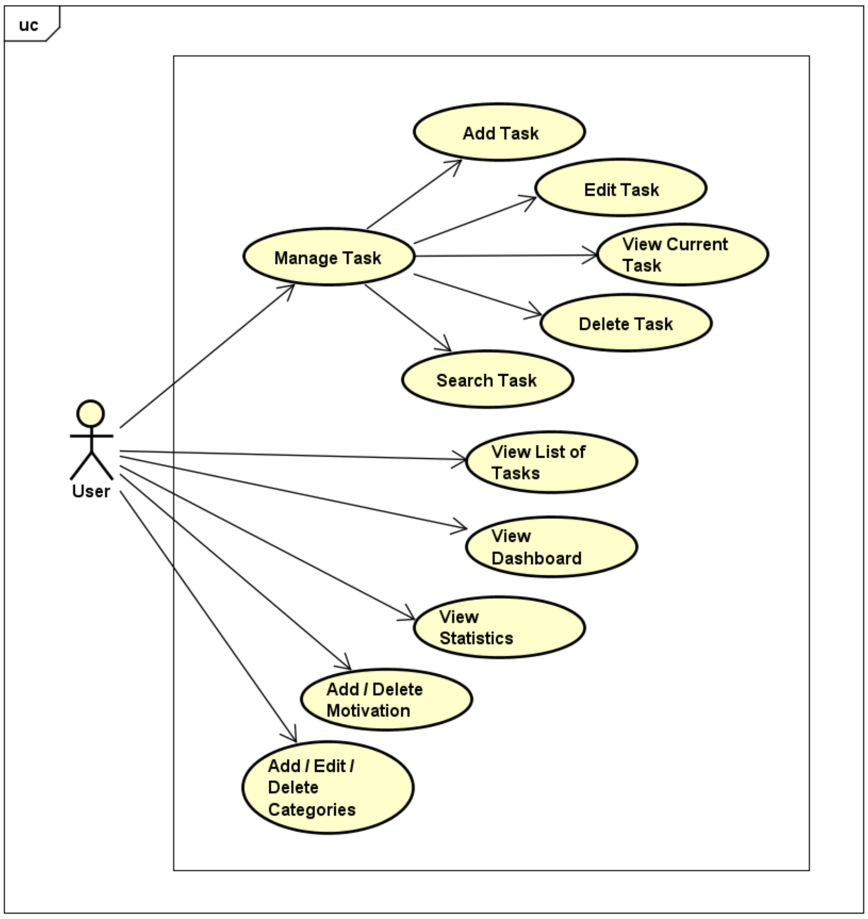


Figure 1: USE CASE DIAGRAM, for user interacting with Personal Mobile Time Management and Motivation Application.

2. Context Case Diagram

   There are two major entities that interact with the system: the mobile user and the user's Google calendar connected to it. The mobile user inputs and manages the tasks entered into the system, while the Google calendar of the user connected to the application can retrieve calendars entries and the application can push publishable task entries to the Google calendar.

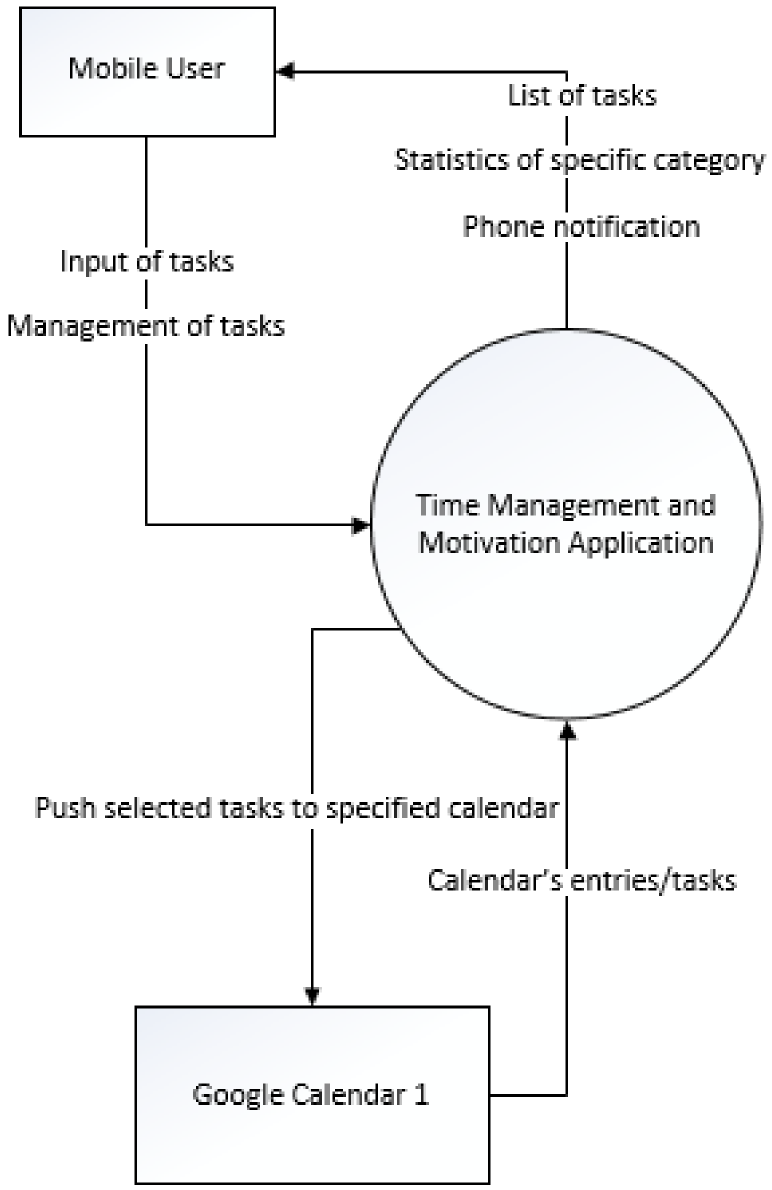


Figure 2: CONTEXT DIAGRAM

3. Entity Relationship Diagram

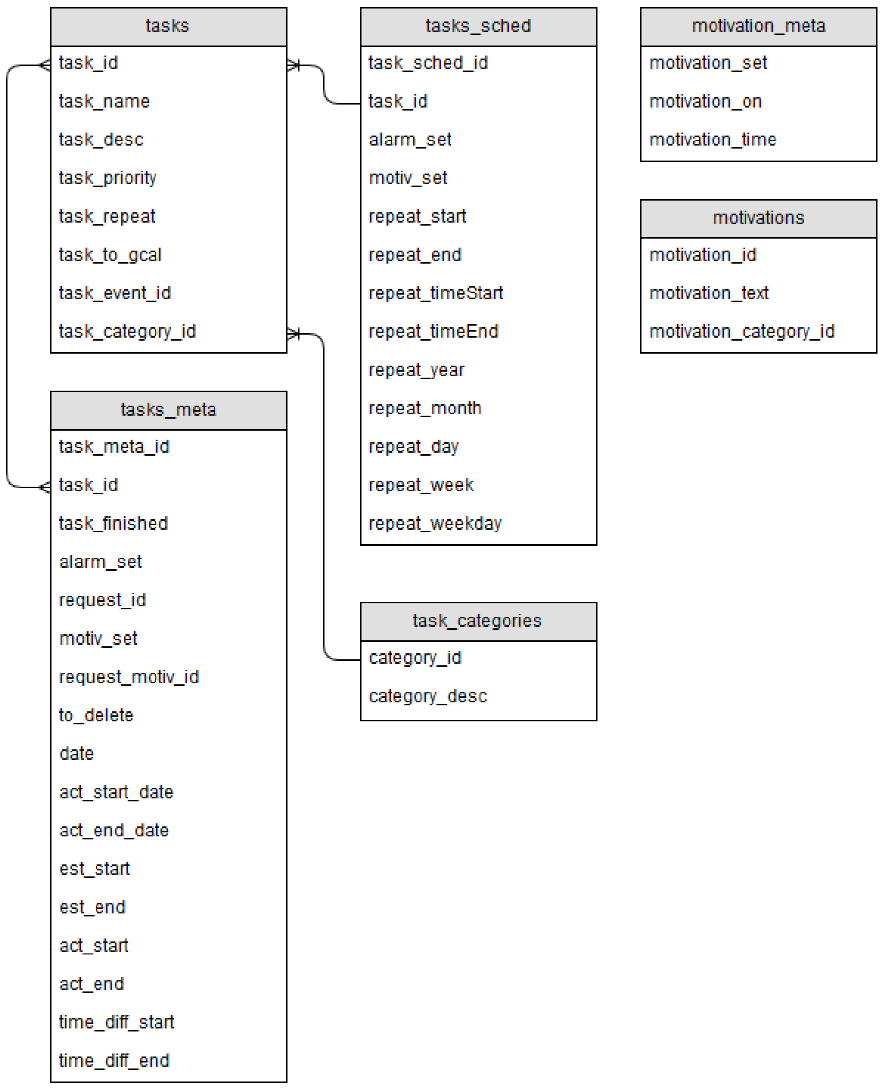


Figure 3: ENTITY RELATIONSHIP DIAGRAM, Personal Mobile Time Management and Motivation Application.

Figure 3 shows the tables to be used for the system.

Table ‘tasks‘ contains the information about the tasks being entered by the user, including data when task is pushed to Google Calendar and if notifications for motivations are set for the task.

Data for scheduling of alarm notification are stored in tasks_sched.

The estimated times and actual times are stored in the tasks_meta. Also stored in this table are the details regarding the specific alarm notification and specific motivation notification set for a specific task.

The tasks are then monitored through each specific categories. These categories are stored in the task_categories table.

The motivations table will contain the customized text motivation entered by the user.

The motivation_meta table contains details for motivation notification.

4. Data Dictionary

Table 1: TASK_CATEGORIES contains the stored categories used by the user.

Field	Data Type	Description	Key
category_id	INTEGER AUTOINCREMENT	unique ID for category	PRIMARY
category_desc	TEXT	category text	

Table 2: TASKS contain general information regarding the task.

Field	Data Type	Description	Key
task_id	INTEGER AUTOINCREMENT	unique ID for task	PRIMARY
task_name	TEXT	task name	
task_desc	TEXT	task description	
task_priority	INTEGER	priority of the task	
task_repeat	TEXT	repeat type of task	
task_to_gcal	INTEGER	boolean; if pushed to Google Calendar	
task_event_id	TEXT	event ID of task pushed	
task_category_id	INTEGER NOT NULL DEFAULT(0) REFERENCES task_categories (category_id)	foreign key; category	

Table 3: TASKS_SCHED contains information regarding the task's time of notification.

Field	Data Type	Description	Key
task_sched_id	INTEGER AUTOINCREMENT	unique ID for task sched	PRIMARY
task_id	INTEGER NOT NULL DEFAULT (0) REFERENCES tasks(task_id)	foreign key; task	
alarm_set	INTEGER	boolean; if alarm is set	
motiv_set	INTEGER	boolean; if motivation is set	
repeat_start	INTEGER	seconds representation of start date since Unix epoch	
repeat_end	INTEGER	seconds representation of end date since Unix epoch	
repeat_timeStart	INTEGER	seconds representation of start time	
repeat_timeEnd	INTEGER	seconds representation of end time	
repeat_year	TEXT	year; * if set	
repeat_month	TEXT	month; * if set	
repeat_day	TEXT	day; * if set	
repeat_week	TEXT	week; * if set	
repeat_weekday	TEXT	weekday; * if set	

Table 4: TASKS_META contains information regarding the specific task's statistics, alarm, and motivation settings

Field	Data Type	Description	Key
task_meta_id	INTEGER AUTOINCREMENT	unique ID for task meta	PRIMARY
task_id	INTEGER NOT NULL DEFAULT (0) REFERENCES tasks(task_id)	foreign key; task	
task_finished	INTEGER	boolean; if task is finished	
alarm_set	INTEGER	boolean; if alarm is set	
request_id	INTEGER	request id of alarm	
motiv_set	INTEGER	boolean; if motivation is set	
request_motiv_id	INTEGER	request id of motivation	
to_delete	INTEGER	boolean; if task is to be deleted	
date	INTEGER	seconds representation of date of specified task since Unix epoch	
act_start_date	INTEGER	seconds representation of actual date start of specified task since Unix epoch	
act_end_date	INTEGER	seconds representation of actual date end of specified task since Unix epoch	

est_start	INTEGER	entered estimated start time in seconds	
est_end	INTEGER	entered estimated end time in seconds	
act_start	INTEGER	actual start time in seconds	
act_end	INTEGER	actual end time in seconds	
time_diff_start	INTEGER	difference bet. estimated start and actual start time	
time_diff_end	INTEGER	difference bet. estimated end and actual end time	

Table 5: MOTIVATION_META contains information regarding a regular time a motivation is sent everyday

Field	Data Type	Description	Key
motivation_set	INTEGER	unique ID	PRIMARY
motivation_on	INTEGER	boolean; if alarm is set	
motivation_time	INTEGER	time of alarm	

Table 6: MOTIVATIONS contains the information about the motivations and the category a motivation belongs

Field	Data Type	Description	Key
motivation_id	INTEGER AUTOINCREMENT	unique ID for motivation	PRIMARY
motivation_text	TEXT	motivation text	
motivation_category_id	INTEGER NOT NULL DEFAULT(0) REFERENCES task_categories (category_id)	foreign key; category	

5. Technical Architecure

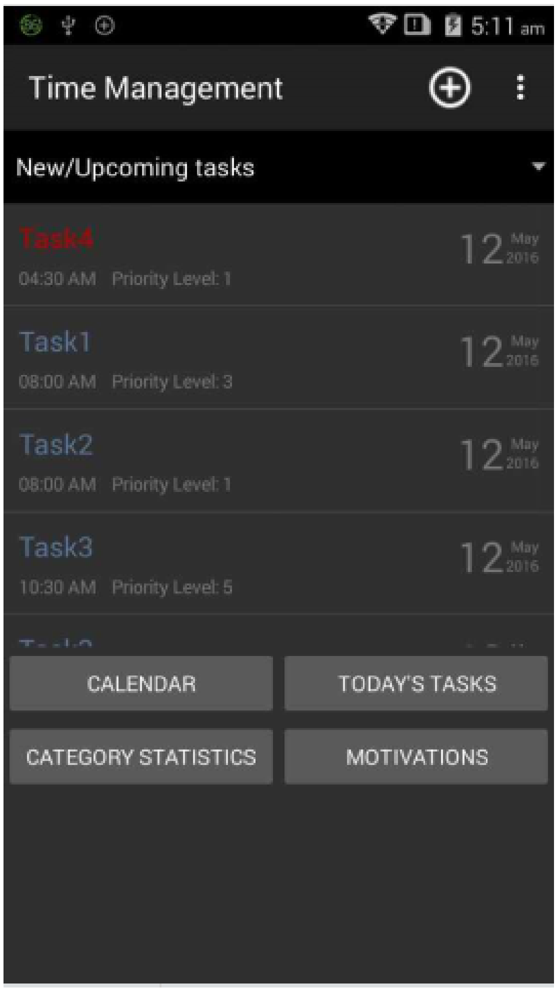
   These are the minimum requirements for the Personal Mobile Time Management and Motivation Application:

   - Server-side requirements:
     - Proper API keys setup for OAuth in Google
   - Clientside requirements:
     - Android SDK 16 / Jelly Bean 4.1 and above

V. Results

The Personal Mobile Time Management and Motivation application is an application used for entering tasks, reminding of tasks, mapping of estimated start and end time against actual start and end time of a task for monitoring, and pushing tasks details to the primary Google Calendar of the users Google account.

The main functionality of the system is to provide the user a mobile application that will be used for time management through the tasks entered and mapping of estimated times against the actual times the tasks were done. Figure 4 shows the main screen, the dashboard, of the application.



Task4
04:30 AM Priority Level: 1
12 May 2016

Task1
08:00 AM Priority Level: 3
12 May 2016

Task2
08:00 AM Priority Level: 1
12 May 2016

Task3
10:30 AM Priority Level: 5
12 May 2016

Figure 4: Task Dashboard. The task dashboard showing the main screen. A variety of tasks can be seen and selected. Buttons can be selected to go to different functionalities. The menu bar contains buttons that also provide access to other functionalities.

Clicking on the Add button on the upper right of the Task Dashboard (Figure 4) leads to the form where a task and details can be entered (Figure 5).

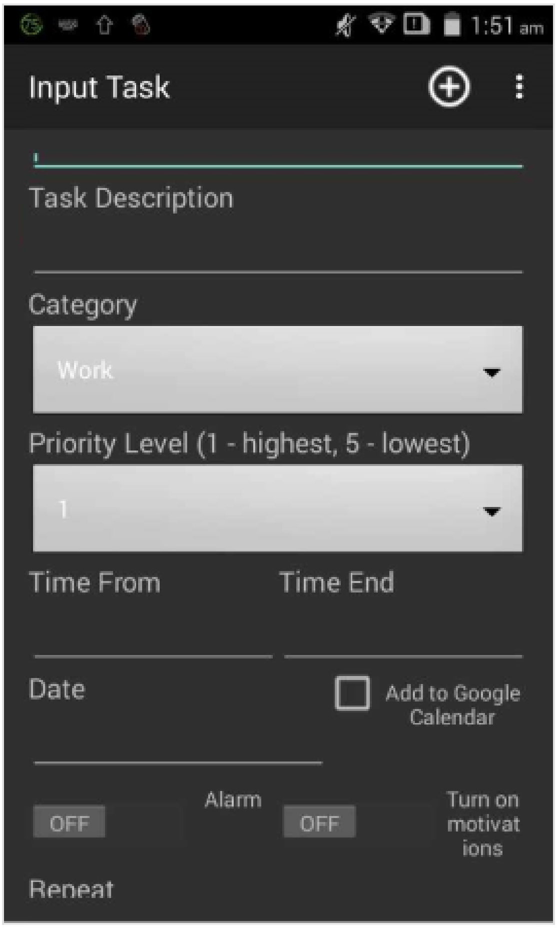
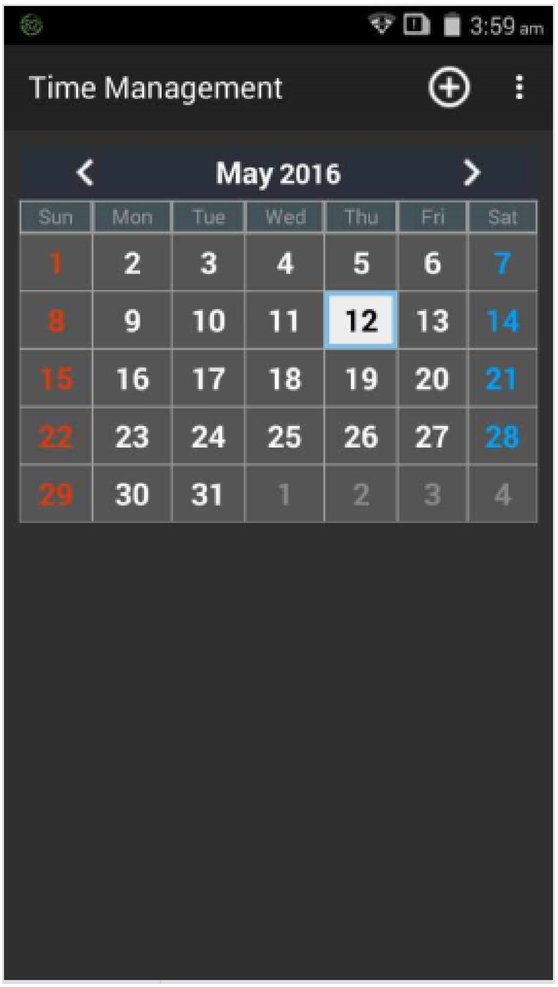


Figure 5: Task Input. The task input view showing the fields for to input details about the task.

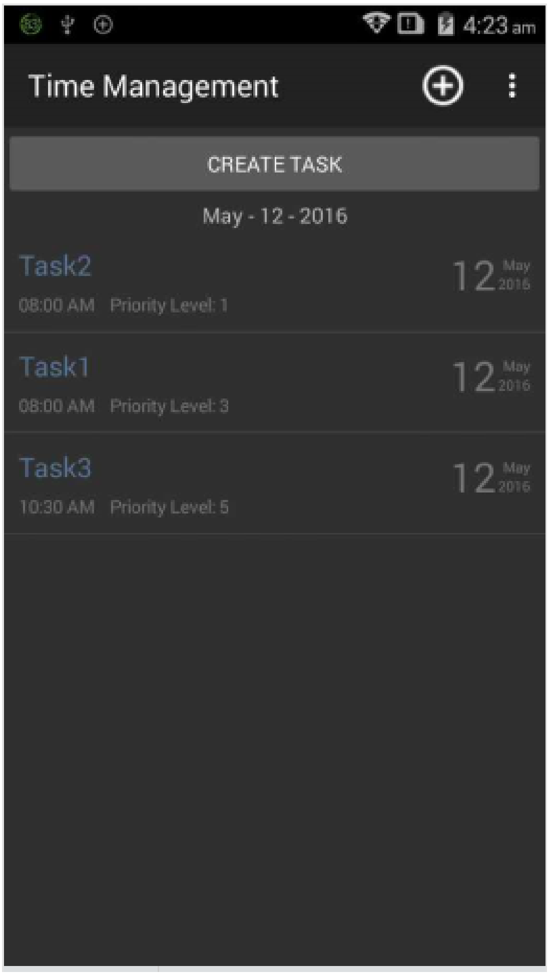
On the Task Dashboard (Figure 4), clicking the Calendar button brings the user to the calendar view, where a user can select a date to view tasks on that date (Figure 6).



Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4

Figure 6: Calendar. The calendar view showing the calendar where the user can select a date.

On the Task Dashboard (Figure 4), clicking on the Todays Tasks button shows the view that contains the tasks for the current day (Figure 7).



May - 12 - 2016

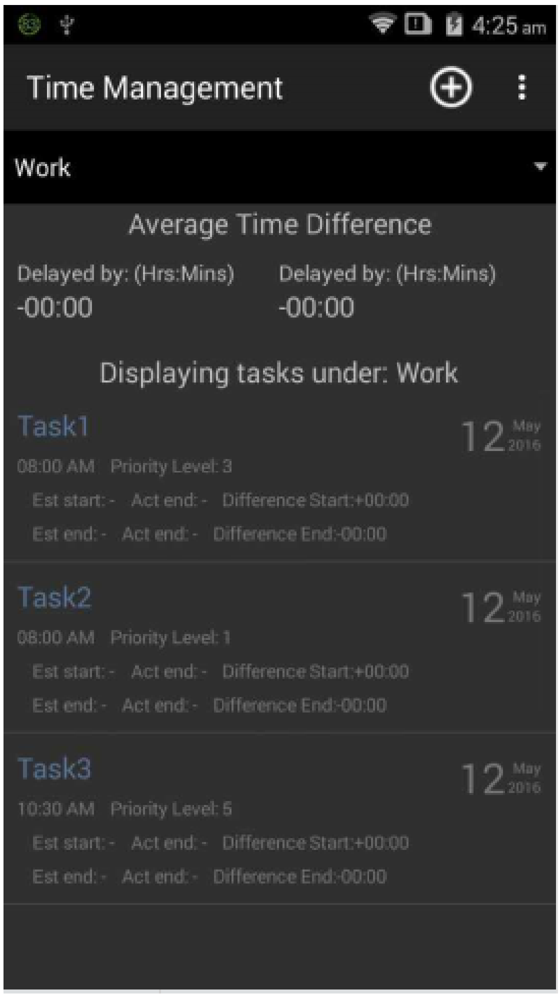
Task2
08:00 AM Priority Level: 1
12 May 2016

Task1
08:00 AM Priority Level: 3
12 May 2016

Task3
10:30 AM Priority Level: 5
12 May 2016

Figure 7: Current Tasks. The current tasks view showing the current tasks for the day.

On the Task Dashboard (Figure 4), clicking the Category Statistics button opens the view that contains details and mapping of estimated times against actual times of tasks under a specific category (Figure 8).


4:25 am

Time Management

Work

Average Time Difference

Delayed by: (Hrs:Mins) -00:00

Delayed by: (Hrs:Mins) -00:00

Displaying tasks under: Work

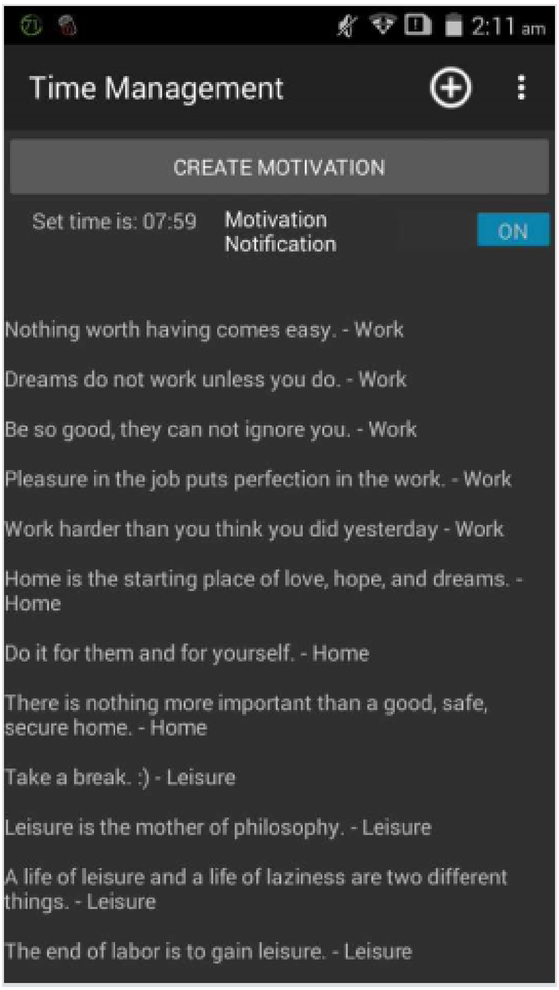
Task1 12 May 2016
08:00 AM Priority Level: 3
Est start: - Act end: - Difference Start:+00:00
Est end: - Act end: - Difference End:-00:00

Task2 12 May 2016
08:00 AM Priority Level: 1
Est start: - Act end: - Difference Start:+00:00
Est end: - Act end: - Difference End:-00:00

Task3 12 May 2016
10:30 AM Priority Level: 5
Est start: - Act end: - Difference Start:+00:00
Est end: - Act end: - Difference End:-00:00

Figure 8: Category Statistics. The category statistics view showing the current tasks for the selected category. The user can also view average time statistics.

On the Task Dashboard (Figure 4), clicking the Motivations button opens the view that contains the default motivations, user-defined motivations and settings for notifications of motivation texts (Figure 9).



Set time is: 07:59

Nothing worth having comes easy. - Work

Dreams do not work unless you do. - Work

Be so good, they can not ignore you. - Work

Pleasure in the job puts perfection in the work. - Work

Work harder than you think you did yesterday - Work

Home is the starting place of love, hope, and dreams. - Home

Do it for them and for yourself. - Home

There is nothing more important than a good, safe, secure home. - Home

Take a break. :) - Leisure

Leisure is the mother of philosophy. - Leisure

A life of leisure and a life of laziness are two different things. - Leisure

The end of labor is to gain leisure. - Leisure

Figure 9: Motivations. The motivations view showing the motivations added in the application. Motivations can be added through the Create Motivation button. Clicking the switch toggles the motivation at a regular interval to on/off. Time will be set when switching the switch on.

Clicking on the menu at the upper right corner, reveals the Categories and Search options. Clicking the Search option opens the Search view (Figure 10) and clicking the Categories option opens the Categories view (Figure 11).

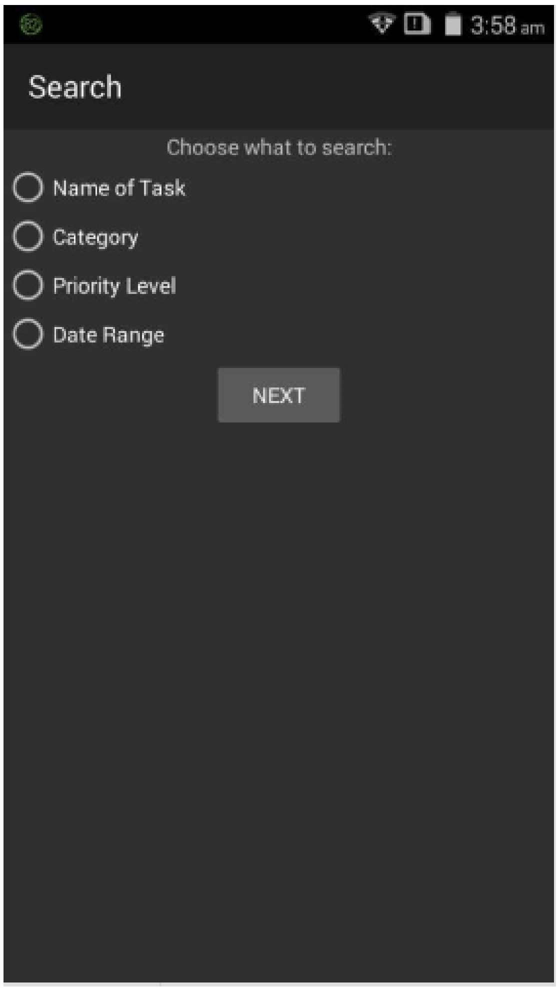


Figure 10: Search. The search view allows the user to search tasks using the task name, category, priority level, and date range.

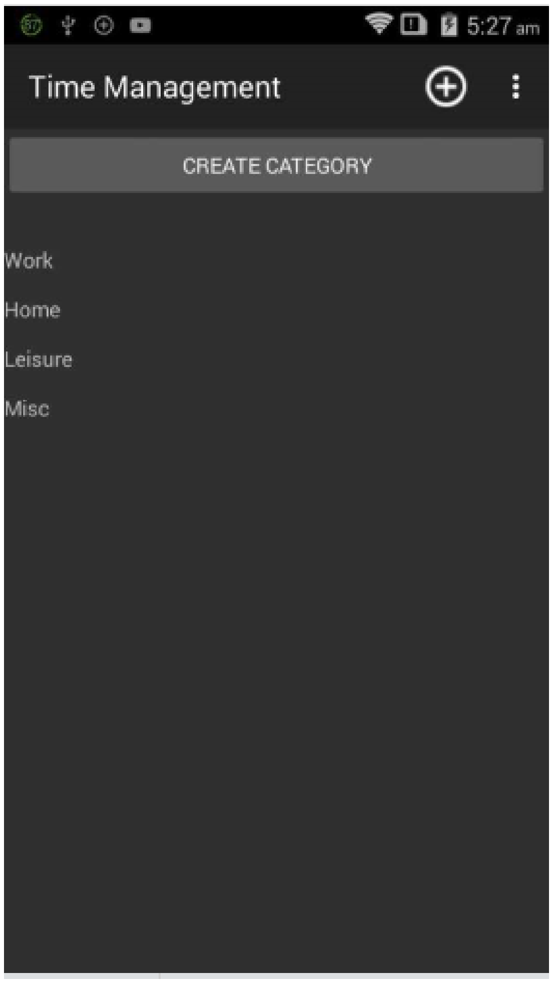
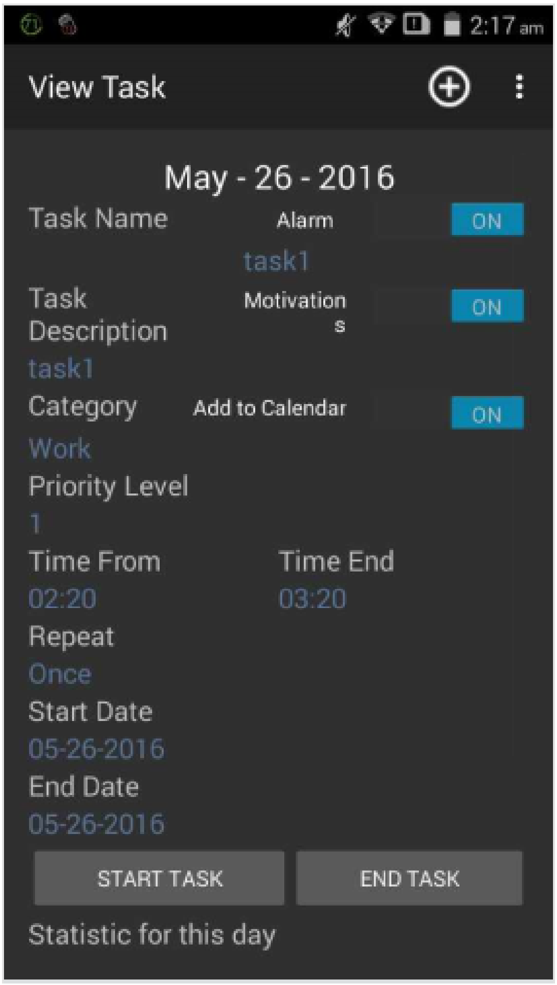


Figure 11: Categories. The categories view showing the categories added in the application. Categories can be added through the Create Category button.

On clicking a task in the dashboard or current tasks view opens the view where task details on the specific date specified will be displayed (Figure 12).


View Task

May - 26 - 2016

Task Name
task1

Alarm ON

Task Description
task1

Motivations ON

Category
Work

Add to Calendar ON

Priority Level
1

Time From
02:20

Time End
03:20

Repeat
Once

Start Date
05-26-2016

End Date
05-26-2016

START TASK

END TASK

Statistic for this day

Figure 12: Task View. The task view showing the details of the selected task for the day. Switches can be used to set the alarm notification, motivation notification, and pushing task to Google Calendar.

On long clicking the task item in the dashboard or current tasks view will show the options: Task Overview, Edit, Delete (Figure 13).

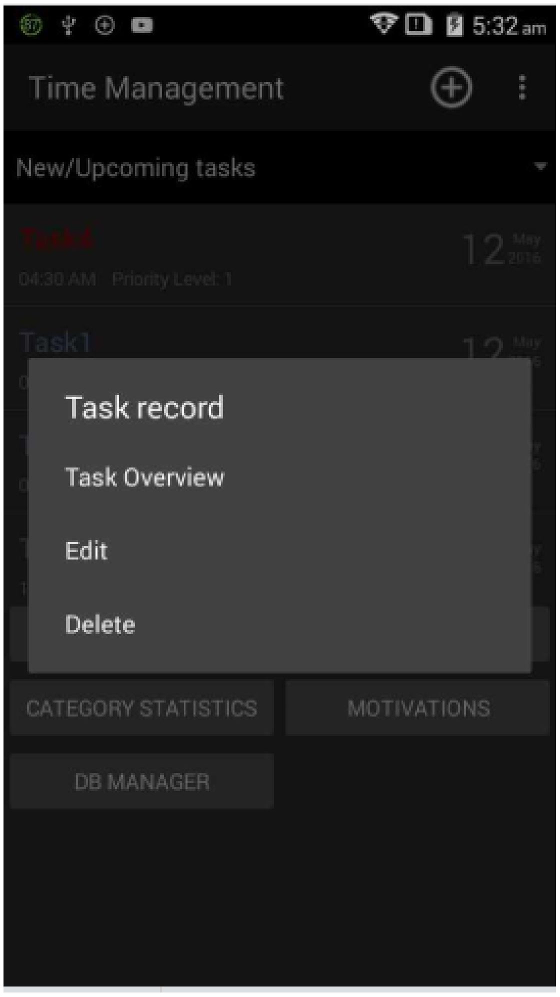
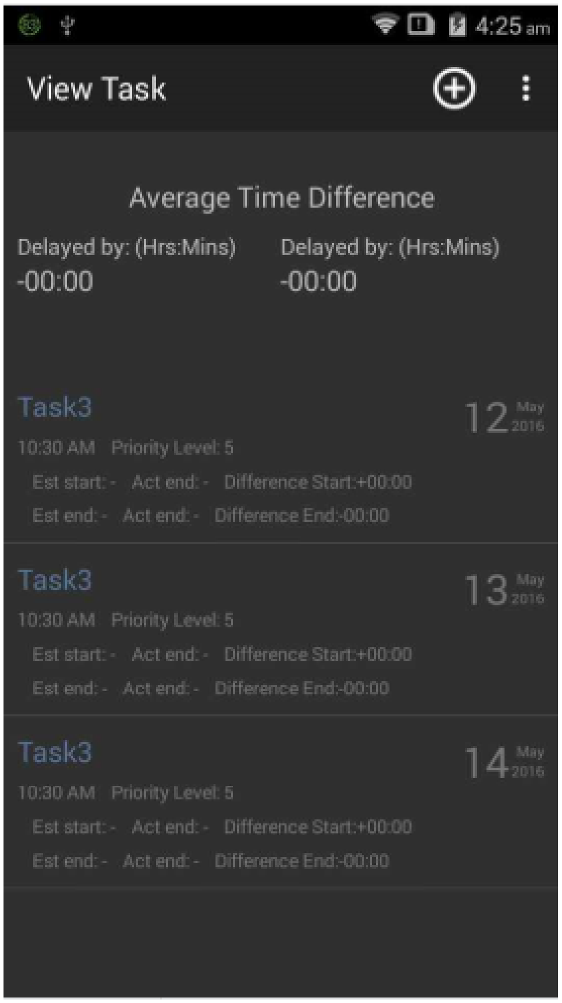


Figure 13: Task Dialog Box. The task dialog box showing options to show task statistics, edit task, and delete task.

Clicking Task Overview will open a view containing details of all tasks occurrences of the current task selected (Figure 14).



4:25 am

View Task

Average Time Difference

Delayed by: (Hrs:Mins) -00:00

Delayed by: (Hrs:Mins) -00:00

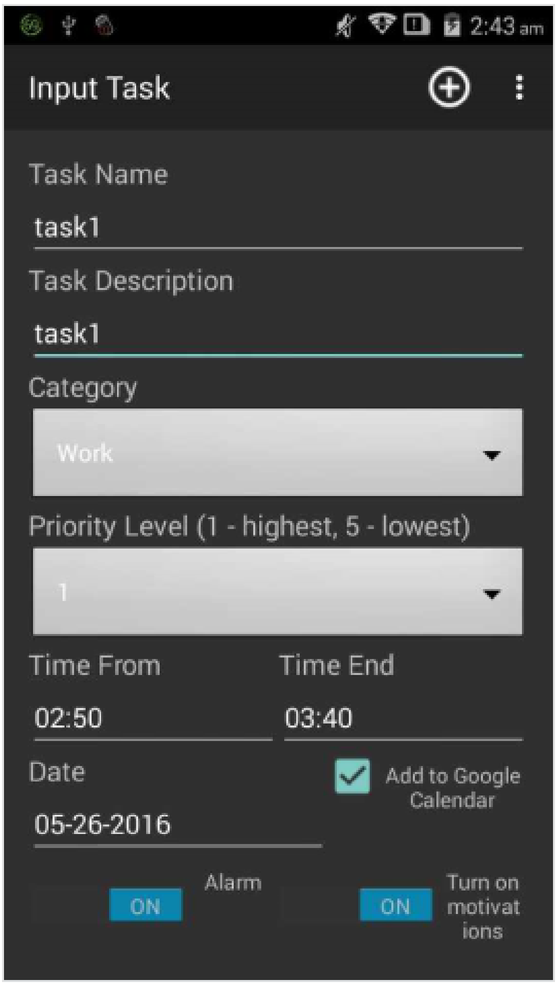
Task3 — 12 May 2016
10:30 AM Priority Level: 5
Est start: - Act end: - Difference Start:+00:00
Est end: - Act end: - Difference End:-00:00

Task3 — 13 May 2016
10:30 AM Priority Level: 5
Est start: - Act end: - Difference Start:+00:00
Est end: - Act end: - Difference End:-00:00

Task3 — 14 May 2016
10:30 AM Priority Level: 5
Est start: - Act end: - Difference Start:+00:00
Est end: - Act end: - Difference End:-00:00

Figure 14: Task Overview. The task overview showing all occurrences of the task selected and the average time of each occurence.

If a task is pushed to the Google Calendar (assuming an internet connection is available), the task will appear to the Google Calendar that the user connects to the application.


2:43 am

Input Task

Task Name

task1

Task Description

task1

Category

Work

Priority Level (1 - highest, 5 - lowest)

1

Time From	Time End
02:50	03:40

Date

05-26-2016

✓ Add to Google Calendar

Alarm: ON

Turn on motivations: ON

Figure 15: The task being entered with Add to Google Calendar checked.

If the task has been set to alarm, the system will notify the user with a notification when the task is starting.

If the task has been set to send a motivation notification, the system will notify the user with a notification of a motivation related to the task's category.

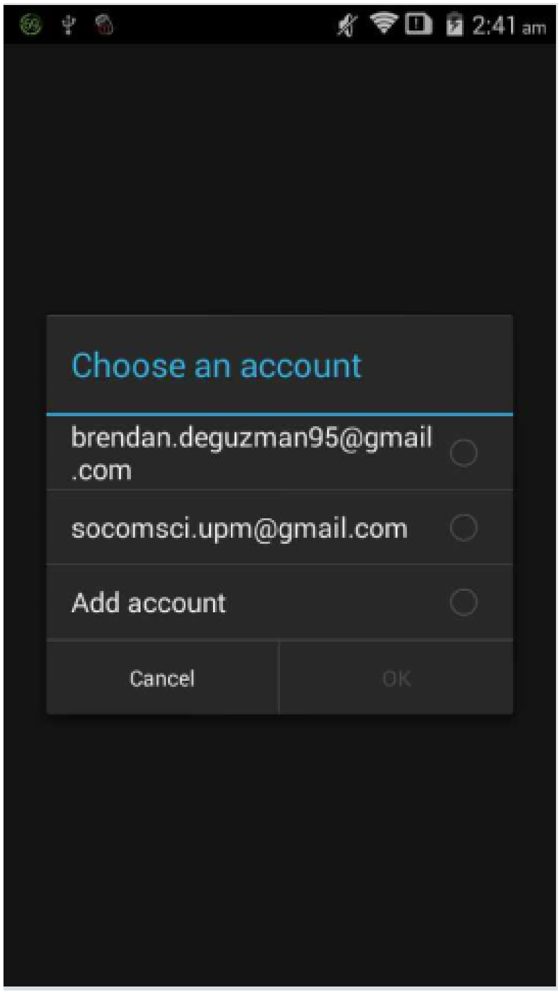


Figure 16: First time running of application will let the user choose the Google Account to associate with the application.

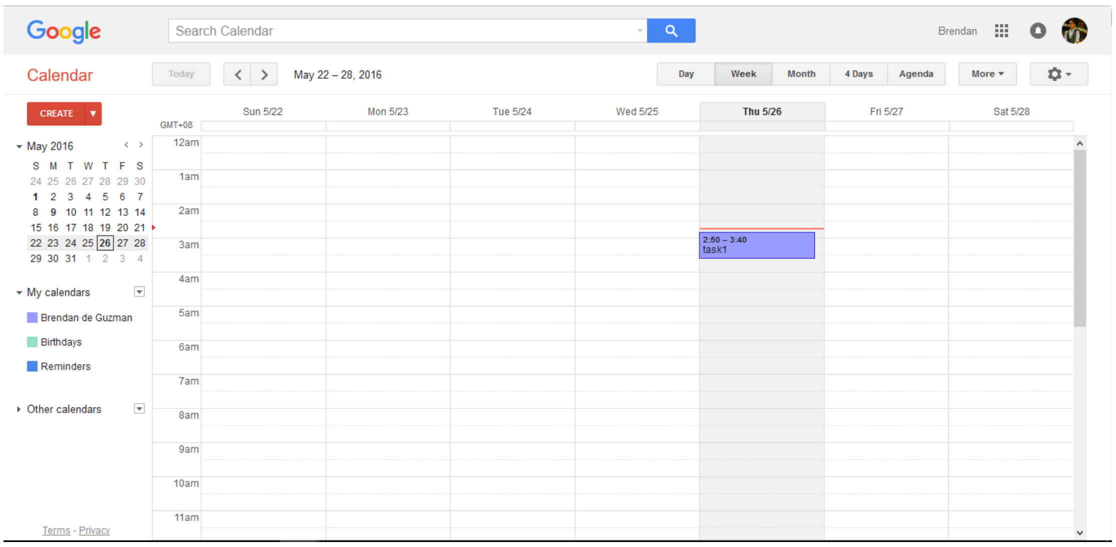
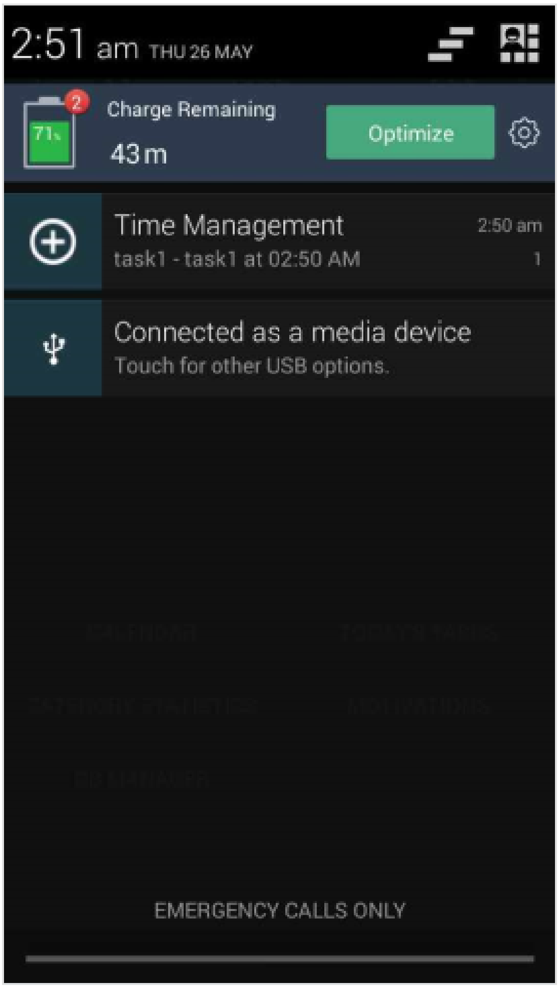


Figure 17: Task appearing in the Google Calendar.


2:51 am THU 26 MAY

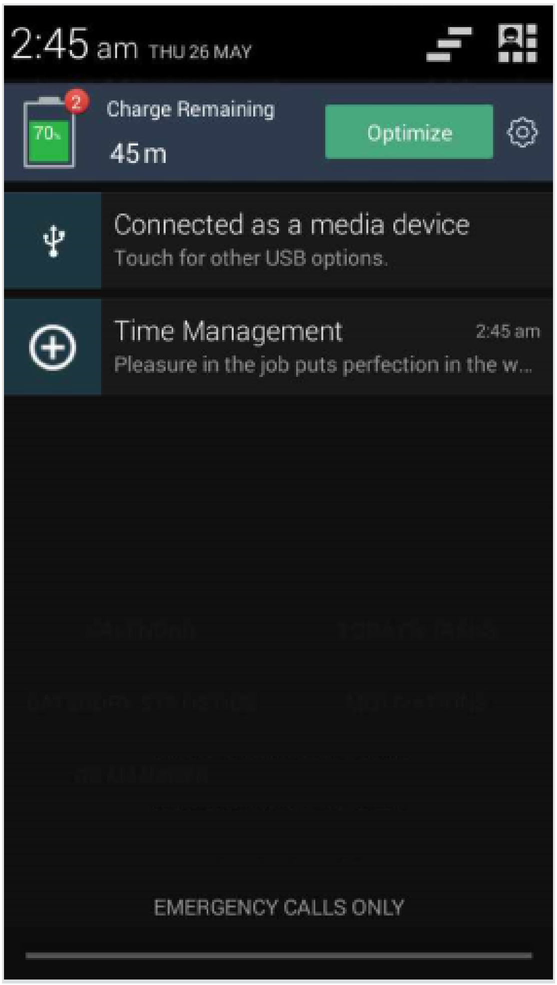
Charge Remaining
43 m

Time Management 2:50 am
task1 - task1 at 02:50 AM 1

Connected as a media device
Touch for other USB options.

EMERGENCY CALLS ONLY

Figure 18: Task notification notifying the user of the task.


2:45 am THU 26 MAY

Charge Remaining

45 m

Connected as a media device

Touch for other USB options.

Time Management 2:45 am

Pleasure in the job puts perfection in the w...

EMERGENCY CALLS ONLY

Figure 19: Notification with a motivation related to the task's category notifying the user.

VI. Discussions

The current working application version of the Time Management Application has functionalities that addresses the primary objectives of the project.

The primary objectives of the application are: 1) Enabling the user to input, edit, and delete tasks that will be monitored through the application, 2) the entered tasks are reminded to the user through notifications (if the notification setting is enabled for the specific task), 3) pushing the task to the users Google Calendar (if the setting to add to the calendar is enabled for the specific task), 4) enable users to add, edit and delete user-defined categories, 5) enable users to add, edit, and delete motivation texts, 6) enable viewing of mapping of estimated start and end times of tasks against the actual start and end times through the dashboard, specific task, task overview, and specific category, and 7) searching of tasks based on name, category, priority, and date range.

With the following functionalities, we can say that the Time Management and Motivation application can raise awareness regarding a specific tasks time estimation of the user and the actual time the user starts and finishes a task. The application can help in the improvement of a users knowledge on gauging how much time a user takes on performing a specific task, especially if the task is a repetitive task. The tool, depending if the person finds inspiration on self-motivational texts, can also help in improving the behavior and mood of a person towards a task.

VII. Conclusions

The Time Management and Motivation Application is an Android application that caters to the needs of people who are needing time management applications that can help in mapping their time estimations to the actual time performed on different tasks.

The users can input, edit and delete tasks that they entered into the application. The users can add the tasks to their Google Calendar and can set to notify them before the task starts. Users can tag tasks to different categories for easier distinction of different tasks.

The users can then view and monitor their estimated times and actual times based on a specific task, specific recurring tasks, or a specific category so the users can gauge on how much time they really take regarding a specific task. In the future, they can then adjust to minimize delays. The users can also add custom text motivations, should they find motivational texts to be helpful in improving their mood.

VIII. Recommendations

The Personal Mobile Time Management and Motivation Application has adequately addressed the problems that it aimed to solve such as the underestimation of how long a person will take to finish a task through the mapping of estimated start and end time of a task to its actual start and end time, the use of a time management tool or application to help in seeing the changes to be made by and individual with regards to time management, and the ability to push tasks to the user's Google Calendar. There are still improvements that are still needed to improve the system and make it more appealing and useful to the user.

One improvement is the addition of settings and customization for the display of time and default preferences. The application uses the default ringtone set in the Android phone the application is installed. The display of time is also set to a 24-hour format. Adding this improvement can improve the interaction of the user with the application and customization will allow the user to fit the application to his or her liking.

Since there are already default motivational texts in place aside from user-defined motivations, another improvement would be the downloading of new motivational texts from a certain site or server that will be used to update or replace the existing default motivational texts. This improvement can provide variety of motivational texts for the user. The motivational texts have a lesser probability to repeat again when the user is notified of a motivation.

IX. Bibliography

[1] "Time management." https://www.psychologytoday.com/basics/time-management. Accessed: 2015-11.

[2] A. Gorlick, "Media multitaskers pay mental price, stanford study shows." http://news.stanford.edu/news/2009/august24/multitask-research-study-082409.html,. Accessed: 2015-12-09.

[3] "11 time management tips." http://sbinfocanada.about.com/cs/timemanagement/a/timemgttips.htm. Accessed: 2015-10.

[4] "Time management skills." http://jobsearch.about.com/od/skills/fl/time-management-skills.htm. Accessed: 2015-10.

[5] "Effects of achievement motivation on behavior." http://jobsearch.about.com/od/skills/fl/time-management-skills.htm. Accessed: 2016-05-11.

[6] G. Moran, "The science behind why inspirational quotes motivate us." http://www.fastcompany.com/3051432/know-it-all/why-inspirational-quotes-motivate-us,. Accessed: 2015-05-11.

[7] "Time management for students: A psychological explanation of why we struggle." http://blog.online.colostate.edu/blog/online-education/time-management-for-students-a-psychological-explanation-of-why-we-struggle/. Accessed: 2016-05-11.

[8] "Time management?." http://sbinfocanada.about.com/od/timemanagement/g/timemanagement.htm. Accessed: 2015-11.

[9] V. Jackson, "Time management: A realistic approach," *Journal of the American College of Radiology*, vol. 6, no. 6, pp. 434–436, 2009.

[10] C. Langa, "Management of time resources for learning through individual study in higher education," *Procedia - Social and Behavioral Sciences*, vol. 76, pp. 13–18, 2013.

[11] K. Nadinloyi, N. Hajloo, N. Garamaleki, and H. Sadeghi, "The study efficacy of time management training on increase academic time management of students," *Procedia - Social and Behavioral Sciences*, vol. 84, pp. 134–138, 2013.

[12] F. Adebayo, "Time management and students academic performance in higher institutions, nigeria a case study of ekiti state," *International Research in Education*, vol. 3, no. 2, pp. 1–12, 2015.

[13] A. Al Khatib, "Time management and its relation to students stress, gender and academic achievement among sample of students at al ain university of science and technology, uae," *International Journal of Business and Social Research*, vol. 4, no. 5, pp. 47–58, 2014.

[14] J. Grissom, S. Loeb, and H. Mitani, "Principal time management skills," *Journal of Educational Administration*, vol. 53, no. 6, pp. 773–793, 2015.

[15] C. MacCann, G. Fogarty, and R. Roberts, "Strategies for success in education: Time management is more important for part-time than full-time community college students," *Learning and Individual Differences*, vol. 22, no. 5, pp. 618–623, 2012.

[16] R. Gafni and N. Geri, "Time management: Procrastination tendency in individual and collaborative tasks," *Interdisciplinary Journal of Information, Knowledge, and Management*, vol. 5, pp. 115–125, 2010.

[17] X. Yang, X. Xu, and L. Zhu, "Media multitasking and psychological wellbeing in chinese adolescents: Time management as a moderator," *Computers in Human Behavior*, vol. 53, pp. 216–222, 2015.

[18] Z. Udin, A. Arif, and A. Saman, "Development of a web-based academic time management system," in *Proceedings of the Sixth International Conference on Computer Supported Cooperative Work in Design (IEEE Cat. No.01EX472)*, pp. 572–574, 2001.

[19] "What is time management? - definition, examples & studies." http://study.com/academy/lesson/what-is-time-management-definition-examples-studies.html. Accessed: 2015-11.

[20] "Reminder systems (definition)." http://www.reference.md/files/D017/mD017010.html. Accessed: 2015-11.

[21] "Reminder system, appointments & meetings reminder, calendar scheduling systems, time management techniques - time thoughts." http://www.timethoughts.com/timemanagement/reminder-systems.htm. Accessed: 2015-11.

[22] "What is smartphone? - definition from whatis.com." http://searchmobilecomputing.techtarget.com/definition/smartphone. Accessed: 2015-11.

[23] "What is android os? - definition from techopedia." https://www.techopedia.com/definition/14873/android-os. Accessed: 2015-11.

[24] M. Karch, "Review of google calendar - internet organization was never easier." http://google.about.com/od/googlereviews/fr/calendarrev.htm,. Accessed: 2015-12.

[25] "Google calendar api — google developers." https://developers.google.com/google-apps/calendar/?hl=en,. Accessed: 2015-12.

X. Appendix

TimeManagement.java

```
package com.specialproblem.brendz.
    timemanagement;

import android.app.AlarmManager;
import android.app.Application;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.sqlite.SQLiteDatabase;
import android.preference.PreferenceManager;

import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .notification.AlarmSetterReceiver;

import java.util.Calendar;

/**
 *
 */
public class TimeManagement extends Application
     {

  public static TableController tblController;
  public static SQLiteDatabase db;
  public static SharedPreferences sp;
  public static AlarmManager alarmManager;

  public static final String TIME_OPTION = "
    time_option";
  public static final String DATE_RANGE = "
    date_range";
  public static final String DATE_FORMAT = "
    date_format";
  public static final String TIME_FORMAT = "
    time_format";
  public static final String VIBRATE_PREF = "
    vibrate_pref";
  public static final String RINGTONE_PREF = "
    ringtone_pref";

  public static final String
    DEFAULT_DATE_FORMAT = "MM-dd-yyyy";
  public static final String
    DEFAULT_TIME_FORMAT = "HH:mm";

  @Override
  public void onCreate() {
    super.onCreate();

    PreferenceManager.setDefaultValues(this, R.
    xml.settings, false);
    sp = PreferenceManager.
    getDefaultSharedPreferences(this);
    tblController = new TableController(this);

    //Start setting notifs for the day
    Intent intent = new Intent(this,
    AlarmSetterReceiver.class);
    alarmManager = (AlarmManager)
    getSystemService(Context.ALARM_SERVICE);
    PendingIntent pendingIntent = PendingIntent
    .getBroadcast(this, 1, intent,
    PendingIntent.FLAG_UPDATE_CURRENT);
    Calendar calendar = Calendar.getInstance();
    calendar.setTimeInMillis(System.
    currentTimeMillis());
    calendar.set(Calendar.SECOND, 0);
    calendar.set(Calendar.MINUTE, 0);
    calendar.set(Calendar.HOUR, 0);
    calendar.set(Calendar.AM_PM, Calendar.AM);
    alarmManager.setInexactRepeating(
    AlarmManager.RTC_WAKEUP, calendar.
    getTimeInMillis(), AlarmManager.
    INTERVAL_DAY, pendingIntent);
    // alarmManager.setInexactRepeating(
    AlarmManager.RTC_WAKEUP, System.
    currentTimeMillis()+1000, 1000*10,
    pendingIntent);
  }

  public static boolean showRemainingTime() {
    return "1".equals(sp.getString(TIME_OPTION,
     "0"));
  }

  public static int getDateRange() {
    return Integer.parseInt(sp.getString(
    DATE_RANGE, "0"));
  }

  public static String getDateFormat() {
    return sp.getString(DATE_FORMAT,
    DEFAULT_DATE_FORMAT);
  }

  public static boolean is24Hours() {
    return sp.getBoolean(TIME_FORMAT, true);
  }

  public static boolean isVibrate() {
    return sp.getBoolean(VIBRATE_PREF, true);
  }

  public static String getRingtone() {
    return sp.getString(RINGTONE_PREF, android.
    provider.Settings.System.
    DEFAULT_NOTIFICATION_URI.toString());
  }
}
```

ToolBarListener.java

```
package com.specialproblem.brendz.
    timemanagement.actionbar;

import android.content.Context;
import android.content.Intent;
import android.view.MenuItem;

import com.specialproblem.brendz.timemanagement
    .view.tasks.CategoryInput;
import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskInput;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskSearchChoice;

/**
 * Class that contains actions for the menu at
    the toolbar
 */
public class ToolBarListener {

  public boolean actionbar_action(MenuItem item
    , Context context){
    switch (item.getItemId()){
      case R.id.action_add:
        createTaskClicked(context);
        return true;
      case R.id.action_categories:
        categoriesClicked(context);
        return true;
      case R.id.action_search:
        searchClicked(context);
        return true;
    }
    return true;
  }

  private void searchClicked(Context context){
    context.startActivity(new Intent(context.
    getApplicationContext(), TaskSearchChoice.
    class));
  }

  private void categoriesClicked(Context
    context){
    context.startActivity(new Intent(context.
    getApplicationContext(), CategoryInput.
    class));
  }

  private void createTaskClicked(Context
    context){
    Intent createTask = new Intent(context,
    TaskInput.class);
    createTask.putExtra("INPUT_TYPE",1);
    context.startActivity(createTask);
  }
}
```

DatabaseHandler.java

```
package com.specialproblem.brendz.
    timemanagement.database_controller;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper
    ;
import android.text.TextUtils;

import com.specialproblem.brendz.timemanagement
    .TimeManagement;

import java.text.SimpleDateFormat;

/**
 * Database Handler
 */
public class DatabaseHandler extends
    SQLiteOpenHelper {

  private static final int DATABASE_VERSION =
    1;
  protected static final String DATABASE_NAME =
     "timemanagement.db";

  public static final SimpleDateFormat sdf =
    new SimpleDateFormat(TimeManagement.
    DEFAULT_DATE_FORMAT);
  public static final SimpleDateFormat stf =
    new SimpleDateFormat(TimeManagement.
    DEFAULT_TIME_FORMAT);

  protected static final String CREATE_TABLE =
    "CREATE TABLE ";
  protected static final String
    TBL_MOTIVATION_META = "motivation_meta";
  protected static final String TBL_MOTIVATIONS
     = "motivations";
  protected static final String
    TBL_TASK_CATEGORIES = "task_categories";
  protected static final String TBL_TASKS = "
    tasks";
  protected static final String TBL_TASKS_SCHED
     = "tasks_sched";
  protected static final String TBL_TASKS_META
    = "tasks_meta";

  //Motivation Meta
  protected static final String COL_MOTIV_SET =
     "motivation_set INTEGER PRIMARY KEY, ";
  protected static final String COL_MOTIV_ON =
    "motivation_on INTEGER, ";
  protected static final String COL_MOTIV_TIME
    = "motivation_time INTEGER ";

  //Motivation
  protected static final String COL_MOTIV_ID =
    "motivation_id INTEGER PRIMARY KEY
    AUTOINCREMENT, ";
  protected static final String COL_MOTIV_TEXT
    = "motivation_text TEXT, ";
  protected static final String
    COL_MOTIV_CATEG_ID = "
    motivation_category_id INTEGER NOT NULL
    DEFAULT (0) REFERENCES task_categories(
    category_id) ";

  //Task Categories
  protected static final String
    COL_TASK_CATEG_ID = "category_id INTEGER
    PRIMARY KEY AUTOINCREMENT, ";
  protected static final String
    COL_TASK_CATEG_DESC = "category_desc TEXT";

  //Tasks
  protected static final String COL_TASK_ID = "
    task_id INTEGER PRIMARY KEY AUTOINCREMENT,
    ";
  protected static final String COL_TASK_NAME =
     "task_name TEXT, ";
  protected static final String COL_TASK_DESC =
     "task_desc TEXT, ";
  protected static final String
    COL_TASK_PRIORITY = "task_priority INTEGER,
     ";
  protected static final String COL_TASK_REPEAT
     = "task_repeat TEXT, ";
  protected static final String
    COL_TASK_ON_GCAL = "task_to_gcal INTEGER, "
    ;
  protected static final String
    COL_TASK_EVENT_ID = "task_event_id TEXT, ";
  protected static final String
    COL_TASK_CATEG_FK_ID = "task_category_id
    INTEGER NOT NULL DEFAULT (0) REFERENCES
    task_categories(category_id) ";

  //Tasks Sched
  protected static final String
    COL_TASK_SCHED_ID = "task_sched_id INTEGER
    PRIMARY KEY AUTOINCREMENT, ";
  protected static final String
    COL_TASK_SCHED_FK_ID = "task_id INTEGER NOT
     NULL DEFAULT (0) REFERENCES tasks(task_id)
    , ";
  protected static final String
    COL_TASK_SCHED_ALARM_SET = "alarm_set
    INTEGER, ";
  protected static final String
    COL_TASK_SCHED_MOTIV_SET = "motiv_set
    INTEGER, ";
  protected static final String
    COL_TASK_RPT_START = "repeat_start INTEGER,
     ";
  protected static final String
    COL_TASK_RPT_END = "repeat_end INTEGER, ";
  protected static final String
    COL_TASK_RPT_TIMESTART = "repeat_timeStart
    INTEGER, ";
  protected static final String
    COL_TASK_RPT_TIMEEND = "repeat_timeEnd
    INTEGER, ";
  protected static final String
    COL_TASK_RPT_YEAR = "repeat_year TEXT, ";
  protected static final String
    COL_TASK_RPT_MONTH = "repeat_month TEXT, ";
  protected static final String
    COL_TASK_RPT_DAY = "repeat_day TEXT, ";
  protected static final String
    COL_TASK_RPT_WEEK = "repeat_week TEXT, ";
  protected static final String
    COL_TASK_RPT_WEEKDAY = "repeat_weekday TEXT
     ";

  //Tasks Meta
  protected static final String
    COL_TASK_META_ID = "task_meta_id INTEGER
    PRIMARY KEY AUTOINCREMENT, ";
  protected static final String
    COL_TASK_META_FK_ID = "task_id INTEGER NOT
    NULL DEFAULT (0) REFERENCES tasks(task_id),
     ";
  protected static final String
    COL_TASK_META_TASK_FINISHED = "
    task_finished INTEGER, ";
  protected static final String
    COL_TASK_META_ALARM_SET = "alarm_set
    INTEGER, ";
  protected static final String
    COL_TASK_META_ALARM_REQID = "request_id
    INTEGER, ";
  protected static final String
    COL_TASK_META_MOTIV_SET = "motiv_set
    INTEGER, ";
  protected static final String
    COL_TASK_META_REQ_MOTIV_SET = "
    request_motiv_id INTEGER, ";
  protected static final String
    COL_TASK_META_TO_DELETE = "to_delete
    INTEGER, ";
  protected static final String
    COL_TASK_META_DATE = "date INTEGER, ";
  protected static final String
    COL_TASK_META_ACT_START_DATE = "
    act_start_date INTEGER, ";
  protected static final String
    COL_TASK_META_ACT_END_DATE = "act_end_date
    INTEGER, ";
  protected static final String
    COL_TASK_META_EST_START = "est_start
  INTEGER, ";
protected static final String
  COL_TASK_META_EST_END = "est_end INTEGER, "
  ;
protected static final String
  COL_TASK_META_ACT_START = "act_start
  INTEGER, ";
protected static final String
  COL_TASK_META_ACT_END = "act_end INTEGER, "
  ;
protected static final String
  COL_TASK_META_TIME_DIFF_S = "
  time_diff_start INTEGER, ";
protected static final String
  COL_TASK_META_TIME_DIFF_E = "time_diff_end
  INTEGER ";

public DatabaseHandler(Context context) {
  super(context, DATABASE_NAME, null,
  DATABASE_VERSION);
}

@Override
public void onCreate(SQLiteDatabase db) {

  //Motivation Meta
  String sql = TextUtils.concat(CREATE_TABLE,
   TBL_MOTIVATION_META, " ( ",
      COL_MOTIV_SET, COL_MOTIV_ON,
  COL_MOTIV_TIME, " )").toString();
  db.execSQL(sql);

  //Motivations
  sql = TextUtils.concat(CREATE_TABLE,
  TBL_MOTIVATIONS, " ( ",
      COL_MOTIV_ID, COL_MOTIV_TEXT,
  COL_MOTIV_CATEG_ID, " )").toString();
  db.execSQL(sql);

  //Task Categories
  sql = TextUtils.concat(CREATE_TABLE,
  TBL_TASK_CATEGORIES, " ( ",
      COL_TASK_CATEG_ID, COL_TASK_CATEG_DESC,
   " )").toString();
  db.execSQL(sql);

  //Tasks
  sql = TextUtils.concat(CREATE_TABLE,
  TBL_TASKS, " ( ",
      COL_TASK_ID ,COL_TASK_NAME,
  COL_TASK_DESC, COL_TASK_PRIORITY,
  COL_TASK_REPEAT,
      COL_TASK_ON_GCAL, COL_TASK_EVENT_ID,
  COL_TASK_CATEG_FK_ID, " ) ").toString();
  db.execSQL(sql);

  //Tasks Sched
  sql = TextUtils.concat(CREATE_TABLE,
  TBL_TASKS_SCHED, " ( ",
      COL_TASK_SCHED_ID, COL_TASK_SCHED_FK_ID
  , COL_TASK_SCHED_ALARM_SET,
      COL_TASK_SCHED_MOTIV_SET,
  COL_TASK_RPT_START, COL_TASK_RPT_END,
      COL_TASK_RPT_TIMESTART,
  COL_TASK_RPT_TIMEEND, COL_TASK_RPT_YEAR,
  COL_TASK_RPT_MONTH,
      COL_TASK_RPT_DAY, COL_TASK_RPT_WEEK,
  COL_TASK_RPT_WEEKDAY, " )").toString();

  db.execSQL(sql);

  //Tasks Meta
  sql = TextUtils.concat(CREATE_TABLE,
  TBL_TASKS_META, " ( ",
      COL_TASK_META_ID, COL_TASK_META_FK_ID,
  COL_TASK_META_TASK_FINISHED,
  COL_TASK_META_ALARM_SET,
      COL_TASK_META_ALARM_REQID,
  COL_TASK_META_MOTIV_SET,
  COL_TASK_META_REQ_MOTIV_SET,
      COL_TASK_META_TO_DELETE,
  COL_TASK_META_DATE,
  COL_TASK_META_ACT_START_DATE,
      COL_TASK_META_ACT_END_DATE,
  COL_TASK_META_EST_START,
      COL_TASK_META_EST_END,
  COL_TASK_META_ACT_START,
  COL_TASK_META_ACT_END,
      COL_TASK_META_TIME_DIFF_S,
  COL_TASK_META_TIME_DIFF_E, " ) ").toString
  ();

  db.execSQL(sql);

  insertCategories(db);
}

@Override
public void onUpgrade(SQLiteDatabase db, int
  oldVersion, int newVersion) {
  db.execSQL("DROP TABLE IF EXISTS " +
  TBL_MOTIVATIONS);
  db.execSQL("DROP TABLE IF EXISTS " +
  TBL_MOTIVATION_META);
  db.execSQL("DROP TABLE IF EXISTS " +
  TBL_TASK_CATEGORIES);
  db.execSQL("DROP TABLE IF EXISTS " +
  TBL_TASKS);
  db.execSQL("DROP TABLE IF EXISTS " +
  TBL_TASKS_META);
  db.execSQL("DROP TABLE IF EXISTS " +
  TBL_TASKS_SCHED);

  onCreate(db);
}

private void insertCategories(SQLiteDatabase
  db){

  //Default Category Values -- cannot be
  deleted
  String sql = "INSERT INTO "+
  TBL_TASK_CATEGORIES +" VALUES (null, 'Work
  '); ";
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_TASK_CATEGORIES +
  " VALUES (null, 'Home'); ";
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_TASK_CATEGORIES +
  " VALUES (null, 'Leisure'); ";
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_TASK_CATEGORIES +
  " VALUES (null, 'Misc'); " ;
  db.execSQL(sql);

  //Default Motivation Meta and Motivations
  sql = "INSERT INTO "+ TBL_MOTIVATION_META +
  " VALUES (1, 1, -3600); ";
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'Nothing worth having comes
  easy.', 1); ";
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'Dreams do not work unless
  you do.', 1); " ;
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'Be so good, they can not
  ignore you.', 1); " ;
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'Pleasure in the job puts
  perfection in the work.', 1); " ;
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'Work harder than you think
  you did yesterday', 1); " ;
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'Home is the starting place
  of love, hope, and dreams.', 2); " ;
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'Do it for them and for
  yourself.', 2); " ;
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'There is nothing more
  important than a good, safe, secure home.',
   2); " ;
  db.execSQL(sql);
  sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
  VALUES (null, 'Take a break. :)', 3); " ;
```

```
    db.execSQL(sql);
    sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
    VALUES (null, 'Leisure is the mother of
    philosophy.', 3); " ;
    db.execSQL(sql);
    sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
    VALUES (null, 'A life of leisure and a life
     of laziness are two different things.', 3)
    ; " ;
    db.execSQL(sql);
    sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
    VALUES (null, 'The end of labor is to gain
    leisure.', 3); " ;
    db.execSQL(sql);
    sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
    VALUES (null, 'Keep on fighting!', 4); " ;
    db.execSQL(sql);
    sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
    VALUES (null, 'Keep on moving forward!', 4)
    ; " ;
    db.execSQL(sql);
    sql = "INSERT INTO "+ TBL_MOTIVATIONS +"
    VALUES (null, 'You can do it!', 4); " ;
    db.execSQL(sql);
    //Default Tasks

    //
  }
}
```

InputProcessing.java

```
package com.specialproblem.brendz.
    timemanagement.database_controller;

import com.specialproblem.brendz.timemanagement
    .task.ObjectTask;
import com.specialproblem.brendz.timemanagement
    .task.ObjectTaskMeta;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.Locale;
import java.util.TimeZone;

/*
 *  Class for processing dates and time
 */
public class InputProcessing {

  public ObjectTask processingInputDates(
    ObjectTask objectTask, String startDate,
    String endDate, String startTime, String
    endTime, String repeatType){

    Calendar cal = Calendar.getInstance();
    cal.setMinimalDaysInFirstWeek(1);

    SimpleDateFormat dateFormat = new
    SimpleDateFormat("MM-dd-yyyy", Locale.US);
    SimpleDateFormat timeFormat = new
    SimpleDateFormat("HH:mm", Locale.US);
    timeFormat.setTimeZone(TimeZone.getDefault
    ());
    timeFormat.setTimeZone(TimeZone.getDefault
    ());
    Date dateStart = null;

    try {
      cal.setTime(timeFormat.parse(startTime));
      objectTask.setRepeat_timeStart(cal.
    getTimeInMillis() / 1000);
      cal.setTime(timeFormat.parse(endTime));
      objectTask.setRepeat_timeEnd(cal.
    getTimeInMillis() / 1000);
      dateStart = dateFormat.parse(startDate);
      cal.setTime(dateStart);
      objectTask.setRepeat_start(cal.
    getTimeInMillis() / 1000);
      if (repeatType.equals("Once")){
        cal.set(Calendar.HOUR_OF_DAY, 23);
        cal.set(Calendar.MINUTE, 59);
        cal.add(Calendar.SECOND, 59);
        objectTask.setRepeat_end(cal.
  getTimeInMillis() / 1000);
      }else{
        cal.setTime(dateFormat.parse(endDate));
        cal.set(Calendar.HOUR_OF_DAY, 23);
        cal.set(Calendar.MINUTE, 59);
        cal.add(Calendar.SECOND, 59);
        objectTask.setRepeat_end(cal.
  getTimeInMillis() / 1000);
      }
    } catch (ParseException e) {
      e.printStackTrace();
    }

    if (repeatType.equals("Once")){
      objectTask.setTask_repeat_day(null);
      objectTask.setTask_repeat_month(null);
      objectTask.setTask_repeat_year(null);
      objectTask.setTask_repeat_week(null);
      objectTask.setTask_repeat_weekday(null);
    }else{
      switch (repeatType){
        case "Daily":
          objectTask.setTask_repeat_day("*");
          objectTask.setTask_repeat_month("*");
          objectTask.setTask_repeat_year("*");
          objectTask.setTask_repeat_week("*");
          objectTask.setTask_repeat_weekday("*"
  );
          break;
        case "Weekly":
          cal.setTime(dateStart);
          objectTask.setTask_repeat_day("*");
          objectTask.setTask_repeat_month("*");
          objectTask.setTask_repeat_year("*");
          objectTask.setTask_repeat_week("*");
          objectTask.setTask_repeat_weekday(
  Integer.toString(cal.get(Calendar.
  DAY_OF_WEEK)));
          break;
        case "Monthly":
          cal.setTime(dateStart);
          objectTask.setTask_repeat_day(Integer
  .toString(cal.get(Calendar.DAY_OF_MONTH)));
          objectTask.setTask_repeat_month("*");
          objectTask.setTask_repeat_year("*");
          objectTask.setTask_repeat_week("*");
          objectTask.setTask_repeat_weekday("*"
  );
          break;
        default:
          break;
      }
    }
    return objectTask;
  }

  public ObjectTask editInputProcessing(
    ObjectTask objectTask){
    Calendar cal = Calendar.getInstance();
    SimpleDateFormat dateFormat = new
    SimpleDateFormat("MM-dd-yyyy", Locale.US);
    SimpleDateFormat timeFormat = new
    SimpleDateFormat("HH:mm", Locale.US);

    cal.setTimeInMillis(objectTask.
    getRepeat_start()*1000);
    String startDate = dateFormat.format(cal.
    getTime());
    objectTask.setDateStart(startDate);
    cal.setTimeInMillis(objectTask.
    getRepeat_end()*1000);
    String endDate = dateFormat.format(cal.
    getTime());
    objectTask.setDateEnd(endDate);
    cal.setTimeInMillis(objectTask.
    getRepeat_timeStart()*1000);
    String startTime = timeFormat.format(cal.
    getTime());
    objectTask.setTimeStart(startTime);
    cal.setTimeInMillis(objectTask.
    getRepeat_timeEnd()*1000);
    String endTime = timeFormat.format(cal.
    getTime());
    objectTask.setTimeEnd(endTime);

    return objectTask;
  }

  public ObjectTaskMeta editInputMetaProcessing
    (ObjectTaskMeta objectTaskMeta){

    Calendar cal = Calendar.getInstance();
    SimpleDateFormat timeFormat = new
    SimpleDateFormat("HH:mm", Locale.US);

    if(objectTaskMeta.getEst_start()!=null&&
    objectTaskMeta.getEst_start()!=0L){
      cal.setTimeInMillis(objectTaskMeta.
    getEst_start()*1000);
      objectTaskMeta.setStr_est_start(
    timeFormat.format(cal.getTime()));
    }else{
      objectTaskMeta.setStr_est_start("-");
    }
    if(objectTaskMeta.getAct_start()!=null&&
    objectTaskMeta.getAct_start()!=0L){
      cal.setTimeInMillis(objectTaskMeta.
    getAct_start()*1000);
      objectTaskMeta.setStr_act_start(
    timeFormat.format(cal.getTime()));
    }else{
      objectTaskMeta.setStr_act_start("-");
    }
    if(objectTaskMeta.getEst_end()!=null&&
    objectTaskMeta.getEst_end()!=0L){
      cal.setTimeInMillis(objectTaskMeta.
    getEst_end()*1000);
      objectTaskMeta.setStr_est_end(timeFormat.
    format(cal.getTime()));
    }else{
      objectTaskMeta.setStr_est_end("-");
    }
    if(objectTaskMeta.getAct_end()!=null&&
    objectTaskMeta.getAct_end()!=0L){
      cal.setTimeInMillis(objectTaskMeta.
    getAct_end()*1000);
      objectTaskMeta.setStr_act_end(timeFormat.
    format(cal.getTime()));
    }else{
      objectTaskMeta.setStr_act_end("-");
    }
    return objectTaskMeta;
  }
}
```

SearchTask.java

```
package com.specialproblem.brendz.
    timemanagement.database_controller;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.os.Bundle;
import android.support.v4.widget.
    SimpleCursorAdapter;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity
    ;
import android.view.Menu;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .task.ObjectTask;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskView;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Calendar;
```

```
/**
 * Activity for searching
 */
public class SearchTask extends
    AppCompatActivity {

  private TableController tblController;

  private TextView mTextView;
  private ListView mListView;

  private String searchType;
  private String query;
  private String dateStart;
  private String dateEnd;

  @Override
  public void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.
    activity_search_task);
    tblController = TimeManagement.
    tblController;
    mTextView = (TextView) findViewById(R.id.
    text);
    mListView = (ListView) findViewById(R.id.
    list);
    Bundle extras = getIntent().getExtras();
    if(extras!=null){
      searchType = extras.getString("
    SEARCH_TYPE");
      if(searchType.equals("name")) {
        query = extras.getString("NAME");
        showResults(query, null, null);
      }else if(searchType.equals("category")) {
        query = extras.getString("CATEGORY");
        showResults(query, null, null);
      }else if(searchType.equals("priority")) {
        query = extras.getString("PRIORITY");
        showResults(query, null, null);
      }else if(searchType.equals("date")){
        dateStart = extras.getString("
    DATE_START");
        dateEnd = extras.getString("DATE_END");
        query = dateStart + " to " + dateEnd;
        showResults(query, dateStart, dateEnd);
      }
    }
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_search, menu);
    return true;
  }

  private void showResults(String query, String
     dateStart, String dateEnd) {

    Cursor cursor = null;
    String[] columns = new String[] {"_id", "
    task_name", "task_desc"};
    if(searchType.equals("date")) {
      cursor = tblController.getWordMatches(
    dateStart, dateEnd, columns, searchType);
    }else {
      cursor = tblController.getWordMatches(
    query, null, columns, searchType);
    }

    if (cursor == null) {
      // There are no results
      mTextView.setText(getString(R.string.
    no_results, new String[] {query}));
    } else {
      // Display the number of results
      int count = cursor.getCount();
      String countString = getResources().
    getQuantityString(R.plurals.search_results,
          count, new Object[] {count, query});
      mTextView.setText(countString);
  // Specify the columns we want to display
 in the result
  String[] from = new String[] {"task_name"
,
      "task_desc" };
  // Specify the corresponding layout
elements where we want the columns to go
  int[] to = new int[] { R.id.task_searched
,
      R.id.task_definition };
  // Create a simple cursor adapter for the
 definitions and apply them to the ListView
  final SimpleCursorAdapter words = new
SimpleCursorAdapter(this,
      R.layout.display_search, cursor, from
, to);
  mListView.setAdapter(words);

  // Define the on-click listener for the
list items
  mListView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
    public void onItemClick(AdapterView<?>
parent, View view, int position, long id) {
      final Context context = view.
getContext();
      final int posID = (int)words.
getItemId(position);
      final ArrayAdapter<String>
arrayAdapter = new ArrayAdapter<String>(
          context,
          android.R.layout.
select_dialog_singlechoice);
      ObjectTask objTask = tblController.
readSingleRecord(posID);

      final SimpleDateFormat dateFormat =
new SimpleDateFormat(TimeManagement.
DEFAULT_DATE_FORMAT);
      Calendar cal = Calendar.getInstance()
;

      cal.setTimeInMillis(objTask.
getRepeat_end()*1000);
      Long endDate = cal.getTimeInMillis();

      cal.setTimeInMillis(objTask.
getRepeat_start()*1000);

      switch (objTask.getTask_repeat()){
        case "Once":
          arrayAdapter.add(dateFormat.
format(cal.getTime()));
          break;
        case "Daily":
          while(cal.getTimeInMillis()<=
endDate){
            arrayAdapter.add(dateFormat.
format(cal.getTime()));
            cal.add(Calendar.DAY_OF_MONTH,
1);
          }
          break;
        case "Weekly":
          while(cal.getTimeInMillis()<=
endDate){
            arrayAdapter.add(dateFormat.
format(cal.getTime()));
            cal.add(Calendar.WEEK_OF_YEAR,
1);
          }
          break;
        case "Monthly":
          while(cal.getTimeInMillis()<=
endDate){
            arrayAdapter.add(dateFormat.
format(cal.getTime()));
            cal.add(Calendar.MONTH, 1);
          }
          break;
      }

      AlertDialog.Builder builder = new
AlertDialog.Builder(context)
          .setTitle("Select Task Date")
          .setNegativeButton("Cancel",
              new DialogInterface.
OnClickListener() {
                @Override
                public void onClick(
DialogInterface dialog, int which) {
                  dialog.dismiss();
                }
              });

          builder.setAdapter(arrayAdapter,
              new DialogInterface.
OnClickListener() {
                @Override
                public void onClick(
DialogInterface dialog, int which) {
                  String strName = arrayAdapter
.getItem(which);
                  Calendar selectedDate =
Calendar.getInstance();
                  try {
                    selectedDate.setTime(
dateFormat.parse(strName));
                  } catch (ParseException e) {
                    e.printStackTrace();
                  }
                  Intent wordIntent = new
Intent(getApplicationContext(), TaskView.
class);
                  wordIntent.putExtra("TASK_ID"
,posID);
                  wordIntent.putExtra("
CURR_MONTH",selectedDate.get(Calendar.MONTH
)+1);
                  wordIntent.putExtra("CURR_DAY
",selectedDate.get(Calendar.DAY_OF_MONTH));
                  wordIntent.putExtra("
CURR_YEAR",selectedDate.get(Calendar.YEAR))
;
                  startActivity(wordIntent);
                  System.out.println(posID+" "+
 selectedDate.get(Calendar.MONTH) +" " +
selectedDate.get(Calendar.DAY_OF_MONTH) +"
"+selectedDate.get(Calendar.YEAR));
                }
              }
          );
          builder.show();
        }
      });
    }
  }

}
```

TableController.java

```
package com.specialproblem.brendz.
    timemanagement.database_controller;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.DatabaseUtils;
import android.database.SQLException;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.
    SQLiteQueryBuilder;
import android.util.Log;
import android.database.MatrixCursor;

import com.specialproblem.brendz.timemanagement
    .task.ObjectCategory;
import com.specialproblem.brendz.timemanagement
    .task.ObjectMotivation;
import com.specialproblem.brendz.timemanagement
    .task.ObjectTask;
import com.specialproblem.brendz.timemanagement
    .task.ObjectTaskMeta;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.HashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.TimeZone;

/**
 * Class for database manipulation
 */
public class TableController extends
    DatabaseHandler {

  private SimpleDateFormat dateFormat = new
    SimpleDateFormat("MM d yyyy HH mm ss",
    Locale.US);
  private Calendar cal = Calendar.getInstance()
    ;

  public TableController(Context context) {
    super(context);
  }

  //ALARM
    ------------------------------------------------------------------------------------------------------------------------

  public Cursor setAlarm(int currMonth, int
    currDay, int currYear){
    cal.setMinimalDaysInFirstWeek(1);
    String dayStart = currMonth+" "+currDay+" "
    +currYear+ " 00 00 00";
    String dayEnd = currMonth+" "+currDay+" "+
    currYear+ " 23 59 59";

    Date dayStartDate = null;
    Date dayEndDate = null;
    try {
      dayStartDate = dateFormat.parse(dayStart)
    ;
      dayEndDate = dateFormat.parse(dayEnd);
    } catch (ParseException e) {
      e.printStackTrace();
    }

    cal.setTime(dayStartDate);
    int week = cal.get(Calendar.WEEK_OF_MONTH);
    int day = cal.get(Calendar.DAY_OF_WEEK);

    String unixStartTime = Long.toString(
    dayStartDate.getTime() / 1000);
    String unixEndTime = Long.toString(
    dayEndDate.getTime() / 1000);
    String sql = "SELECT TS.‘task_id‘ AS ‘_id‘,
     TS.‘task_name‘, TS.‘task_desc‘, TS.‘
    task_priority‘, " +
        "TS1.‘repeat_timeStart‘, TS1.‘alarm_set
    ‘, TS1.‘motiv_set‘, TS.‘task_category_id‘ "
     +
        "FROM ‘tasks‘ TS JOIN ‘tasks_sched‘ TS1
     " +
        "ON TS.‘task_id‘ = TS1.‘task_id‘ " +
        "WHERE (( ? - repeat_end >= 0 AND
    repeat_start - ? >= 0)" +
        " OR " +
        "( " +
        " (repeat_year = ? OR repeat_year = ’*’
     ) " +
        " AND (repeat_month = ? OR repeat_month
     = ’*’ ) " +
        " AND (repeat_day = ? OR repeat_day =
    ’*’ ) " +
        " AND (repeat_week = ? OR repeat_week =
     ’*’ ) " +
        " AND (repeat_weekday = ? OR
    repeat_weekday = ’*’ ) " +
        " AND repeat_start <= ? AND repeat_end
    >= ? )"
        + ") ORDER BY ‘task_priority‘ DESC"
        ;
    SQLiteDatabase db = this.
    getWritableDatabase();
    Cursor cursor = db.rawQuery(sql, new String
    [] { unixEndTime, unixStartTime,
        Integer.toString(currYear), Integer.
    toString(currMonth), Integer.toString(
    currDay),
        Integer.toString(week), Integer.
    toString(day), unixStartTime, unixEndTime
    });
    return cursor;
}

public Cursor setRealAlarm(int taskId, int
  currMonth, int currDay, int currYear){
  cal.setMinimalDaysInFirstWeek(1);
  String dayStart = currMonth+" "+currDay+" "
  +currYear+ " 00 00 00";
  Date dayStartDate = null;
  try {
    dayStartDate = dateFormat.parse(dayStart)
  ;
  } catch (ParseException e) {
    e.printStackTrace();
  }

  String unixStartTime = Long.toString(
  dayStartDate.getTime() / 1000);
  String sql = "SELECT TM.‘alarm_set‘ AS ‘
  _real_alarm‘, TM.‘request_id‘, TM.‘
  to_delete‘, " +
      "TM.‘motiv_set‘, TM.‘request_motiv_id‘
  "+
      "FROM ‘tasks_meta‘ TM " +
      "WHERE TM.‘task_id‘ = ? AND TM.‘date‘ =
   ? ";
  SQLiteDatabase db = this.
  getWritableDatabase();
  Cursor cursor = db.rawQuery(sql, new String
  [] { Integer.toString(taskId),
  unixStartTime });
  return cursor;
}

public boolean alarmChange(int taskId, int
  month, int day, int year, int alarmSet){
  SQLiteDatabase db = this.
  getWritableDatabase();
  ContentValues values = new ContentValues();
  int requestID = (int) System.
  currentTimeMillis();
  cal.setTimeInMillis(0);
  cal.set(year, month, day,0,0,0);

  Long date = cal.getTimeInMillis()/1000;
  values.put("request_id", requestID);
  values.put("alarm_set", alarmSet);
  String where = "task_id = ? AND date = ?";
  String[] whereArgs = { Integer.toString(
  taskId), Long.toString(date) };

  boolean setAlarm = db.update("tasks_meta",
  values, where, whereArgs) > 0;
  return setAlarm;
}

public boolean updateRequestID(int taskId,
  int month, int day, int year, int requestID
  , int alarmSet, int toDelete){

  SQLiteDatabase db = this.
  getWritableDatabase();

  ContentValues values = new ContentValues();

  cal.setTimeInMillis(0);
  cal.set(year, month, day,0,0,0);
  Long date = cal.getTimeInMillis()/1000;

  values.put("task_id", taskId);
  values.put("date", date);
  values.put("request_id", requestID);
  values.put("alarm_set", alarmSet);
  values.put("to_delete", toDelete);

  String where = "task_id = ? AND date = ?";
  String[] whereArgs = { Integer.toString(
  taskId), Long.toString(date) };

  boolean updateSuccessfulTasksMeta = db.
  update("tasks_meta", values, where,
  whereArgs) > 0;
  boolean createSuccessfulTaskMeta = false;
  if(!updateSuccessfulTasksMeta){
    createSuccessfulTaskMeta = db.insert("
  tasks_meta", null, values) > 0;
  }
```

```
    return createSuccessfulTaskMeta ||
    updateSuccessfulTasksMeta;
  }

  public boolean motivChange(int taskId, int
    month, int day, int year, int motivSet){
    SQLiteDatabase db = this.
    getWritableDatabase();
    ContentValues values = new ContentValues();
    int requestID = (int) System.
    currentTimeMillis();
    cal.setTimeInMillis(0);
    cal.set(year, month, day,0,0,0);

    Long date = cal.getTimeInMillis()/1000;
    values.put("request_motiv_id", requestID);
    values.put("motiv_set", motivSet);
    String where = "task_id = ? AND date = ?";
    String[] whereArgs = { Integer.toString(
    taskId), Long.toString(date) };

    boolean setMotiv = db.update("tasks_meta",
    values, where, whereArgs) > 0;
    return setMotiv;
  }

  public boolean updateRequestMotivID(int
    taskId, int month, int day, int year, int
    requestID, int motivSet){
    SQLiteDatabase db = this.
    getWritableDatabase();
    ContentValues values = new ContentValues();

    cal.setTimeInMillis(0);
    cal.set(year, month, day,0,0,0);
    Long date = cal.getTimeInMillis()/1000;

    values.put("task_id", taskId);
    values.put("date", date);
    values.put("request_motiv_id", requestID);
    values.put("motiv_set", motivSet);

    String where = "task_id = ? AND date = ?";
    String[] whereArgs = { Integer.toString(
    taskId), Long.toString(date) };

    boolean updateSuccessfulTasksMeta = db.
    update("tasks_meta", values, where,
    whereArgs) > 0;
    boolean createSuccessfulTaskMeta = false;
    if(!updateSuccessfulTasksMeta){
      createSuccessfulTaskMeta = db.insert("
    tasks_meta", null, values) > 0;
    }
    return createSuccessfulTaskMeta ||
    updateSuccessfulTasksMeta;
  }

  public int getRequestID(int taskId, int month
    , int day, int year){
    SQLiteDatabase db = this.
    getWritableDatabase();
    int requestId = 0;
    cal.setTimeInMillis(0);
    cal.set(year, month, day,0,0,0);
    Long date = cal.getTimeInMillis()/1000;
    String sql = "SELECT TM.`request_id` FROM `
    tasks_meta` TM WHERE TM.`task_id` = ? AND
    TM.`date` = ?";
    Cursor cursor = db.rawQuery(sql, new String
    []{Integer.toString(taskId),Long.toString(
    date)});
    if(cursor.moveToFirst()){
      requestId = Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("request_id
    ")));
    }
    return requestId;
  }

  public int getRequestMotivID(int taskId, int
    month, int day, int year){
    SQLiteDatabase db = this.
    getWritableDatabase();
    int requestId = 0;
    cal.setTimeInMillis(0);
    cal.set(year, month, day,0,0,0);
    Long date = cal.getTimeInMillis()/1000;
    String sql = "SELECT TM.`request_motiv_id`
    FROM `tasks_meta` TM WHERE TM.`task_id` = ?
     AND TM.`date` = ?";
    Cursor cursor = db.rawQuery(sql, new String
    []{Integer.toString(taskId),Long.toString(
    date)});
    if(cursor.moveToFirst()){
      requestId = Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("
    request_motiv_id")));
    }
    return requestId;
  }

  //CALENDAR SET/UNSET
    -------------------------------------------------------

  public boolean setToGoogleCalendar(int taskId
    , int toGcal){

    SQLiteDatabase db = this.
    getWritableDatabase();
    ContentValues values = new ContentValues();

    values.put("task_to_gcal", toGcal);
    String where = "task_id = ?";
    String[] whereArgs = { Integer.toString(
    taskId) };

    boolean updateSuccessfulGcal = db.update("
    tasks", values, where, whereArgs) > 0;
    return updateSuccessfulGcal;
  }

  public boolean setEventID(int taskId, String
    eventID){
    ContentValues values = new ContentValues();
    SQLiteDatabase db = this.
    getWritableDatabase();
    values.put("task_event_id", eventID);
    String where = "task_id = ?";
    String[] whereArgs = { Integer.toString(
    taskId) };
    boolean updateSuccessfulEventID = db.update
    ("tasks", values, where, whereArgs) > 0;
    return updateSuccessfulEventID;
  }

  public String getEventID(int taskId){
    SQLiteDatabase db = this.
    getWritableDatabase();
    String sql = "SELECT TS.`task_event_id`
    FROM `tasks` TS WHERE task_id = ?";
    Cursor cursor = db.rawQuery(sql, new String
    []{Integer.toString(taskId)});
    String eventID = null;
    if (cursor.moveToFirst()) {
      eventID = cursor.getString(cursor.
    getColumnIndex("task_event_id"));
    }
    return eventID;
  }

  //CATEGORY
    -------------------------------------------------------

  public boolean insertCategory(String
    category_name){
    ContentValues values = new ContentValues();
    values.put("category_desc", category_name);
    SQLiteDatabase db = this.
    getWritableDatabase();
    boolean createSuccessful = db.insert("
    task_categories", null, values) > 0;
    return createSuccessful;
  }

  public boolean deleteCategory(int id){
    boolean deleteSuccessful;
    SQLiteDatabase db = this.
    getWritableDatabase();
    String where = "category_id = ?";
    String[] whereArgs = { Integer.toString(id)
     };
    deleteSuccessful = db.delete("
    task_categories", where, whereArgs) > 0;
    return deleteSuccessful;
  }

  public boolean updateCategory(ObjectCategory
    objectCategory){
    ContentValues values = new ContentValues();
    values.put("category_desc", objectCategory.
    getCategory_desc());
    String where = "category_id = ?";
    String[] whereArgs = { Integer.toString(
    objectCategory.getId()) };
    SQLiteDatabase db = this.
    getWritableDatabase();
    boolean updateSuccessful = db.update("
    task_categories", values, where, whereArgs)
     > 0;
    return updateSuccessful;
  }

  public List<ObjectCategory> getAllCategories
    (){
    List<ObjectCategory> categories = new
    ArrayList<ObjectCategory>();

    // Select All Query
    String selectQuery = "SELECT * FROM
    task_categories";

    SQLiteDatabase db = this.
    getReadableDatabase();
    Cursor cursor = db.rawQuery(selectQuery,
    null);

    // looping through all rows and adding to
    list
    if (cursor.moveToFirst()) {
      do {
        ObjectCategory obj = new ObjectCategory
    ();
        obj.setId(Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("
    category_id"))));
        obj.setCategory_desc(cursor.getString(
    cursor.getColumnIndex("category_desc")));
        categories.add(obj);
      } while (cursor.moveToNext());
    }
    // closing connection
    cursor.close();
    // returning categories
    return categories;
  }

  public ObjectCategory getSingleCategory(int
    category_id){
    ObjectCategory objCategory = new
    ObjectCategory();
    String sql = "SELECT * from task_categories
     WHERE category_id = ?";

    SQLiteDatabase db = this.
    getWritableDatabase();
    Cursor cursor = db.rawQuery(sql, new String
    [] {Integer.toString(category_id)});

    if (cursor.moveToFirst()){
      objCategory.setId(Integer.parseInt(cursor
    .getString(cursor.getColumnIndex("
    category_id"))));
      objCategory.setCategory_desc(cursor.
    getString(cursor.getColumnIndex("
    category_desc")));
    }
    cursor.close();
    return objCategory;
  }

  //MOTIVATION
  public boolean insertMotivation(
    ObjectMotivation obj){
    ContentValues values = new ContentValues();
    values.put("motivation_text", obj.
    getMotivation_text());
    values.put("motivation_category_id", obj.
    getCategory_id());
    SQLiteDatabase db = this.
    getWritableDatabase();
    boolean createSuccessful = db.insert("
    motivations", null, values) > 0;
    return createSuccessful;
  }

  public boolean deleteMotivation(int id){
    boolean deleteSuccessful;
    SQLiteDatabase db = this.
    getWritableDatabase();
    String where = "motivation_id = ?";
    String[] whereArgs = { Integer.toString(id)
     };
    deleteSuccessful = db.delete("motivations",
     where, whereArgs) > 0;
    return deleteSuccessful;
  }

  public boolean updateMotivation(
    ObjectMotivation obj){

    SQLiteDatabase db = this.
    getWritableDatabase();
    String sql = "SELECT TC.`category_id` FROM
    `task_categories` TC WHERE category_desc =
    ?";
    Cursor cursor = db.rawQuery(sql, new String
    []{obj.getCategory_desc()});
    int category_id = 0;
    if (cursor.moveToFirst()) {
      category_id = Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("
    category_id")));
    }
    ContentValues values = new ContentValues();
    values.put("motivation_text", obj.
    getMotivation_text());
    values.put("motivation_category_id",
    category_id);
    String where = "motivation_id = ?";
    String[] whereArgs = { Integer.toString(obj
    .getId()) };

    boolean updateSuccessful = db.update("
    motivations", values, where, whereArgs) >
    0;
    return updateSuccessful;
  }

  public List<ObjectMotivation>
    getAllMotivations(){
    List<ObjectMotivation> motivations = new
    ArrayList<ObjectMotivation>();

    // Select All Query
    String selectQuery = "SELECT * FROM
    motivations";

    SQLiteDatabase db = this.
    getReadableDatabase();
    Cursor cursor = db.rawQuery(selectQuery,
    null);

    // looping through all rows and adding to
    list
    if (cursor.moveToFirst()) {
      do {
        ObjectMotivation obj = new
    ObjectMotivation();
        obj.setId(Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("
    motivation_id"))));
        obj.setMotivation_text(cursor.getString
    (cursor.getColumnIndex("motivation_text")))
    ;
        int category_id = Integer.parseInt(
    cursor.getString(cursor.getColumnIndex("
    motivation_category_id")));
        String category_desc = null;
        String sql = "SELECT TC.`category_desc`
     FROM `task_categories` TC WHERE
    category_id = ?";
        Cursor cursorCategory = db.rawQuery(sql
  , new String[]{Integer.toString(category_id
  )});
      if (cursorCategory.moveToFirst()) {
        category_desc = cursorCategory.
  getString(cursorCategory.getColumnIndex("
  category_desc"));
      }
      cursorCategory.close();
      obj.setCategory_desc(category_desc);
      obj.setCategory_id(category_id);
      motivations.add(obj);
    } while (cursor.moveToNext());
  }
  // closing connection
  cursor.close();

  return motivations;
}

public List<ObjectMotivation>
  getMotivationsCategory(int category_id){
  List<ObjectMotivation> motivations = new
  ArrayList<ObjectMotivation>();

  // Select All Query
  String selectQuery = "SELECT * FROM
  motivations WHERE motivation_category_id =
  ?";

  SQLiteDatabase db = this.
  getReadableDatabase();
  Cursor cursor = db.rawQuery(selectQuery,
  new String[]{ Integer.toString(category_id)
  });

  // looping through all rows and adding to
  list
  if (cursor.moveToFirst()) {
    do {
      ObjectMotivation obj = new
  ObjectMotivation();
      obj.setId(Integer.parseInt(cursor.
  getString(cursor.getColumnIndex("
  motivation_id"))));
      obj.setMotivation_text(cursor.getString
  (cursor.getColumnIndex("motivation_text")))
  ;
      String category_desc = null;
      String sql = "SELECT TC.`category_desc`
   FROM `task_categories` TC WHERE
  category_id = ?";
      Cursor cursorCategory = db.rawQuery(sql
  , new String[]{Integer.toString(category_id
  )});
      if (cursorCategory.moveToFirst()) {
        category_desc = cursorCategory.
  getString(cursorCategory.getColumnIndex("
  category_desc"));
      }
      cursorCategory.close();
      obj.setCategory_desc(category_desc);
      obj.setCategory_id(category_id);
      motivations.add(obj);
    } while (cursor.moveToNext());
  }
  // closing connection
  cursor.close();

  return motivations;
}

public ObjectMotivation getSingleMotivation(
  int motivation_id){
  ObjectMotivation objectMotivation = new
  ObjectMotivation();
  String sql = "SELECT * from motivations
  WHERE motivation_id = ?";

  SQLiteDatabase db = this.
  getWritableDatabase();
  Cursor cursor = db.rawQuery(sql, new String
  [] {Integer.toString(motivation_id)});
  DatabaseUtils.dumpCursor(cursor);
  if (cursor.moveToFirst()){
    objectMotivation.setId(Integer.parseInt(
  cursor.getString(cursor.getColumnIndex("
  motivation_id"))));
    objectMotivation.setMotivation_text(
  cursor.getString(cursor.getColumnIndex("
  motivation_text")));
  }
  cursor.close();
  return objectMotivation;
}

public boolean setMotivationMeta(int
  toggleMotiv, long time){
  ContentValues values = new ContentValues();
  values.put("motivation_on", toggleMotiv);
  values.put("motivation_time", time);
  String where = "motivation_set = 1";
  SQLiteDatabase db = this.
  getWritableDatabase();
  boolean updateSuccessful = db.update("
  motivation_meta", values, where, null) > 0;
  return updateSuccessful;
}

public int[] getMotivationMeta(){
  int[] res = new int[2];
  String selectQuery = "SELECT * FROM
  motivation_meta WHERE motivation_set = 1";
  SQLiteDatabase db = this.
  getReadableDatabase();
  Cursor cursor = db.rawQuery(selectQuery,
  null);
  if (cursor.moveToFirst()) {
    res[0] = Integer.parseInt(cursor.
  getString(cursor.getColumnIndex("
  motivation_on")));
    res[1] = Integer.parseInt(cursor.
  getString(cursor.getColumnIndex("
  motivation_time")));
  }
  cursor.close();
  return res;
}

//CATEGORY STAT
  ---------------------------------------------------------

public Cursor getTasksUnderCategories(String
  category){
  String sql = "SELECT TS.`task_id` AS `_id`,
   TS.`task_name`, TS.`task_desc`, TS.`
  task_priority`, " +
      "TS1.`repeat_timeStart`, TM.`date`, " +
      "TM.`est_start`, TM.`act_start`, TM.`
  est_end`, TM.`act_end`, TM.`time_diff_start
  `, TM.`time_diff_end` FROM `tasks` TS " +
      "JOIN `tasks_sched` TS1 JOIN `
  task_categories` TC JOIN `tasks_meta` TM "
  +
      "ON TS.`task_id` = TS1.`task_id` AND "
  +
      "TC.`category_id` = TS.`
  task_category_id` AND TM.`task_id` = TS.`
  task_id` WHERE TC.`category_desc` = ? " ;
  SQLiteDatabase db = this.
  getWritableDatabase();
  Cursor cursor = db.rawQuery(sql, new String
  [] {category});
  return cursor;
}

//TASKS OVERVIEW
public Cursor getTaskFromId(int taskId){
  String sql = "SELECT TS.`task_id` AS `_id`,
   TS.`task_name`, TS.`task_desc`, TS.`
  task_priority`, " +
      "TS1.`repeat_timeStart`, TM.`date`, " +
      "TM.`est_start`, TM.`act_start`, TM.`
  est_end`, TM.`act_end`, TM.`time_diff_start
  `, TM.`time_diff_end` FROM `tasks` TS " +
      "JOIN `tasks_sched` TS1 JOIN `
  task_categories` TC JOIN `tasks_meta` TM "
  +
      "ON TS.`task_id` = TS1.`task_id` AND "
  +
      "TC.`category_id` = TS.`
  task_category_id` AND TM.`task_id` = TS.`
  task_id` WHERE TS.`task_id` = ? ";
  SQLiteDatabase db = this.
  getWritableDatabase();
  Cursor cursor = db.rawQuery(sql, new String
  [] {Integer.toString(taskId)});
  DatabaseUtils.dumpCursor(cursor);
  return cursor;
}

//DASHBOARD
  ----------------------------------------------------------------------------------------------------------

public Cursor readNewTasks(String type){
  cal.setMinimalDaysInFirstWeek(1);
  Calendar start = Calendar.getInstance();
  Calendar end = Calendar.getInstance();
  int month = start.get(Calendar.MONTH);
  int day = start.get(Calendar.DAY_OF_MONTH);
  int year = start.get(Calendar.YEAR);
  start.set(year, month, day, 0, 0, 0);
  end.set(year, month, day+5, 23, 59, 59);
  int endDay = day+5;
  int endWeek = end.get(Calendar.
  WEEK_OF_MONTH);
  int week = start.get(Calendar.WEEK_OF_MONTH
  );
  int weekday = start.get(Calendar.
  DAY_OF_WEEK);
  int endWeekday = end.get(Calendar.
  DAY_OF_WEEK);

  String unixStartTime = Long.toString(start.
  getTimeInMillis() / 1000);
  String unixEndTime = Long.toString(end.
  getTimeInMillis() / 1000);

  String sql = "SELECT TS.`task_id` AS `_id`,
   TS.`task_name`, TS.`task_desc`, TS.`
  task_priority`, " +
      "TS1.`repeat_timeStart`, TS1.`
  repeat_start`, TM.`date` " +
      "FROM `tasks` TS JOIN `tasks_sched` TS1
   JOIN `tasks_meta` TM " +
      "ON TS.`task_id` = TS1.`task_id` AND TS
  .`task_id` = TM.`task_id` " +
      "WHERE " +
      "(( ? - repeat_end >= 0 AND
  repeat_start - ? >= 0)" +
      " OR " +
      "( " +
      " ((repeat_year >= ? OR repeat_year =
  '*') AND (repeat_year <= ? OR repeat_year =
   '*')) " +
      " AND ((repeat_month >= ? OR
  repeat_month = '*') AND (repeat_month <= ?
  OR repeat_month = '*')) " +
      " AND ((repeat_day >= ? OR repeat_day =
   '*') AND (repeat_day <= ? OR repeat_day =
  '*')) " +
      " AND ((repeat_week >= ? OR repeat_week
   = '*') AND (repeat_week <= ? OR
  repeat_week = '*')) " +
      " AND ((repeat_weekday >= ? OR
  repeat_weekday = '*') AND (repeat_weekday
  <= ? OR repeat_weekday = '*')) " +
      " AND (repeat_start <= ? OR repeat_end
  >= ?) )";

  if(type.equals("new"))
    sql += ") ORDER BY `repeat_start`,`
  repeat_timeStart` ASC";
  else if(type.equals("highPrio"))
    sql += ") ORDER BY `task_priority` ASC";

  SQLiteDatabase db = this.
  getWritableDatabase();
  Cursor cursor = db.rawQuery(sql, new String
  [] { unixEndTime, unixStartTime,
      Integer.toString(year), Integer.
  toString(year), Integer.toString(month),
  Integer.toString(month),
      Integer.toString(day), Integer.toString
  (endDay), Integer.toString(week),
      Integer.toString(endWeek), Integer.
  toString(weekday), Integer.toString(
  endWeekday), unixStartTime, unixEndTime });
  return cursor;
}
```

```
public Cursor readDelayedTasks(){
  cal.setMinimalDaysInFirstWeek(1);
  Calendar start = Calendar.getInstance();
  int minute = start.get(Calendar.MINUTE);
  int hour = start.get(Calendar.HOUR_OF_DAY);
  String unixStartDate = Long.toString(start.
  getTimeInMillis() / 1000);
  start.setTimeInMillis(0);
  start.set(Calendar.MINUTE, minute);
  start.set(Calendar.HOUR_OF_DAY, hour);
  String unixStartTime = Long.toString(start.
  getTimeInMillis() / 1000);
  System.out.println(unixStartDate + " " +
  unixStartTime);
  String sql = "SELECT TS.`task_id` AS `_id`,
   TS.`task_name`, TS.`task_desc`, TS.`
  task_priority`, " +
      "TS1.`repeat_timeStart`, TS1.`
  repeat_start`, TM.`date` " +
      "FROM `tasks` TS " +
      "JOIN `tasks_sched` TS1 JOIN `
  tasks_meta` TM " +
      "ON TS.`task_id` = TS1.`task_id` AND TS
  .`task_id` = TM.`task_id` " +
      "WHERE " +
      "((TS1.`repeat_timeStart` - ? < 0) AND
  (TM.`date` - ? <= 0) AND (TM.`act_start` is
   null) AND (TM.`act_end` is null)) " +
      "ORDER BY `repeat_start`, `
  repeat_timeStart` ASC";
  SQLiteDatabase db = this.
  getWritableDatabase();
  Cursor cursor = db.rawQuery(sql,new String
  [] {unixStartTime , unixStartDate});
  return cursor;
}

//SEARCH
  ----------------------------------------------------------

public Cursor getWordMatches(String query,
  String nextQuery, String[] columns, String
  searchType) {
  String selection = null;
  String[] selectionArgs = null;
  if(searchType.equals("name")){
    selection = "task_name LIKE ?";
    selectionArgs = new String[] {query};
  }else if(searchType.equals("category")){
    String sql = "SELECT category_id from
  task_categories WHERE category_desc = ?";
    SQLiteDatabase db = this.
  getReadableDatabase();
    Cursor cursor = db.rawQuery(sql, new
  String[] {query});
    String category_id = null;

    if (cursor.moveToFirst()){
      category_id = cursor.getString(cursor.
  getColumnIndex("category_id"));
    }
    selection = "task_category_id LIKE ?";
    selectionArgs = new String[] {category_id
  };

  }else if(searchType.equals("priority")){
    selection = "task_priority = ?";
    selectionArgs = new String[] {query};
  }else if(searchType.equals("date")){
    Date dateStart = null;
    Date dateEnd = null;
    Calendar cal = Calendar.getInstance();
    try {
      dateStart = sdf.parse(query);
      dateEnd = sdf.parse(nextQuery);
    } catch (ParseException e) {
      e.printStackTrace();
    }
    cal.setTime(dateStart);
    query = Long.toString(cal.getTimeInMillis
  ()/1000);
    cal.setTime(dateEnd);
    cal.set(Calendar.HOUR_OF_DAY, 23);
    cal.set(Calendar.MINUTE, 59);
    cal.add(Calendar.SECOND, 59);
```

```
      nextQuery = Long.toString(cal.
    getTimeInMillis()/1000);
      selection = null;
      selectionArgs = new String[] {query,
    nextQuery};
    }
    return query(selection, selectionArgs,
    columns, searchType);
  }

  private Cursor query(String selection, String
    [] selectionArgs, String[] columns, String
    searchType) {
    SQLiteQueryBuilder builder = new
    SQLiteQueryBuilder();
    if(searchType.equals("date")){
      builder.setTables("tasks_meta");
      String sql = "SELECT TS.`task_id` AS `_id
    `, TS.`task_name`, TS.`task_desc` from `
    tasks` TS JOIN `tasks_sched` TC " +
          "WHERE (TC.`repeat_start` >= ? AND TC
    .`repeat_start` <= ?) AND TS.`task_id` = TC
    .`task_id`";
      SQLiteDatabase db = this.
    getReadableDatabase();
      Cursor cursor = db.rawQuery(sql,
    selectionArgs);
      if (cursor == null) {
        return null;
      } else if (!cursor.moveToFirst()) {
        cursor.close();
        return null;
      }
      return cursor;
    }else{
      builder.setTables("tasks");
      Map<String, String> columnMap = new
    HashMap<String, String>();
      columnMap.put("_id", "task_id AS _id");
      columnMap.put("task_name", "task_name");
      columnMap.put("task_desc", "task_desc");

      builder.setProjectionMap(columnMap);

      Cursor cursor = builder.query(this.
    getReadableDatabase(),
          columns, selection, selectionArgs,
    null, null, null);

      if (cursor == null) {
        return null;
      } else if (!cursor.moveToFirst()) {
        cursor.close();
        return null;
      }
      return cursor;
    }
  }

  //TASK
    ------------------------------------------------------------------------------------------------------

  public int[] createTask(ObjectTask objectTask
    ) {

    int[] retValues = new int[2];
    SQLiteDatabase db = this.
    getWritableDatabase();
    String sql = "SELECT TC.`category_id` FROM
    `task_categories` TC WHERE category_desc =
    ?";
    Cursor cursor = db.rawQuery(sql, new String
    []{objectTask.getTask_category()});
    int category_id = 0;
    if (cursor.moveToFirst()) {
      category_id = Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("
    category_id")));
    }
    ContentValues values = new ContentValues();
    values.put("task_name", objectTask.
    getTask_name());
    values.put("task_desc", objectTask.
    getTask_description());
    values.put("task_category_id", category_id)
    ;
    values.put("task_priority", objectTask.
    getTask_priority());
    values.put("task_repeat", objectTask.
    getTask_repeat());
    values.put("task_to_gcal", objectTask.
    getToCal());

    boolean createSuccessfulTasks = db.insert("
    tasks", null, values) > 0;
    boolean createSuccessfulMeta = false;
    int taskID = 0;
    if(createSuccessfulTasks){
      sql = "SELECT last_insert_rowid(); ";
      cursor = db.rawQuery(sql, null);
      if (cursor.moveToFirst()) {
        taskID = Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("
    last_insert_rowid()")));
        retValues[0] = taskID;
        values = new ContentValues();
        values.put("task_id", taskID);
        values.put("alarm_set", objectTask.
    getAlarmSet());
        values.put("motiv_set", objectTask.
    getMotivSet());
        values.put("repeat_start", objectTask.
    getRepeat_start());
        values.put("repeat_end", objectTask.
    getRepeat_end());
        values.put("repeat_timeStart",
    objectTask.getRepeat_timeStart());
        values.put("repeat_timeEnd", objectTask
    .getRepeat_timeEnd());
        values.put("repeat_year", objectTask.
    getTask_repeat_year());
        values.put("repeat_month", objectTask.
    getTask_repeat_month());
        values.put("repeat_day", objectTask.
    getTask_repeat_day());
        values.put("repeat_week", objectTask.
    getTask_repeat_week());
        values.put("repeat_weekday", objectTask
    .getTask_repeat_weekday());

        createSuccessfulMeta = db.insert("
    tasks_sched", null, values) > 0;
      }
    }
    Calendar calStart = Calendar.getInstance();
    Calendar calEnd = Calendar.getInstance();
    calStart.setTimeInMillis(objectTask.
    getRepeat_start()*1000);
    calEnd.setTimeInMillis(objectTask.
    getRepeat_end()*1000);
    boolean populateTasksMeta =
    populateTasksMeta(false, taskID,calStart.
    get(Calendar.YEAR),calStart.get(Calendar.
    MONTH),
        calStart.get(Calendar.DAY_OF_MONTH),
    calEnd.get(Calendar.YEAR),calEnd.get(
    Calendar.MONTH),calEnd.get(Calendar.
    DAY_OF_MONTH),
        objectTask.getAlarmSet(),objectTask.
    getMotivSet(),0,objectTask.getTask_repeat()
    );
    cursor.close();
    if (createSuccessfulTasks &&
    createSuccessfulMeta && populateTasksMeta){
      retValues[1] = 1;
    } else {
      retValues[1] = 0;
    }
    return retValues;
  }

  public boolean populateTasksMeta(boolean
    repopulate, int taskId, int year, int month
    , int day, int endYear, int endMonth,
              int endDay, int alarmSet,
    int motivSet, int toDelete, String repeat){
    SQLiteDatabase db = this.
    getWritableDatabase();
    ContentValues values = new ContentValues();
    Calendar calEnd = Calendar.getInstance();
    calEnd.set(endYear,endMonth,endDay,0,0,0);
    boolean deleteTaskMetaSuccess = true;
    boolean createSuccessfulTaskMeta = false;
    if(repopulate){
      deleteTaskMetaSuccess = db.delete("
    tasks_meta", "task_id = ?", new String[] {
    Integer.toString(taskId)}) > 0;
    }
    if(deleteTaskMetaSuccess) {
      int daily = 0;
      int weekly = 0;
      int monthly = 0;
      long endDate = calEnd.getTimeInMillis() /
     1000;
      long date = 0;
      while (endDate > date) {
        cal.setTimeInMillis(0);
        switch (repeat) {
          case "Once":
            cal.set(year, month, day, 0, 0, 0);
            break;
          case "Daily":
            cal.set(year, month, day, 0, 0, 0);
            cal.add(Calendar.DAY_OF_MONTH,
    daily);
            daily++;
            break;
          case "Weekly":
            cal.set(year, month, day, 0, 0, 0);
            cal.add(Calendar.WEEK_OF_YEAR,
    weekly);
            weekly++;
            break;
          case "Monthly":
            cal.set(year, month, day, 0, 0, 0);
            cal.add(Calendar.MONTH, monthly);
            month++;
            break;
        }
        date = cal.getTimeInMillis() / 1000;
        int requestID = (int) System.
    currentTimeMillis();
        values.put("task_id", taskId);
        values.put("date", date);
        values.put("request_id", requestID);
        values.put("request_motiv_id",
    requestID * -1);
        values.put("alarm_set", alarmSet);
        values.put("motiv_set", motivSet);
        values.put("to_delete", toDelete);

        createSuccessfulTaskMeta = db.insert("
    tasks_meta", null, values) > 0;
      }
    }
    return createSuccessfulTaskMeta;

  }

  public Cursor readTasks(int currMonth, int
    currDay, int currYear) {

    cal.setMinimalDaysInFirstWeek(1);
    String dayStart = currMonth+" "+currDay+" "
    +currYear+ " 00 00 00";
    String dayEnd = currMonth+" "+currDay+" "+
    currYear+ " 23 59 59";

    Date dayStartDate = null;
    Date dayEndDate = null;
    try {
      dayStartDate = dateFormat.parse(dayStart)
    ;
      dayEndDate = dateFormat.parse(dayEnd);
    } catch (ParseException e) {
      e.printStackTrace();
    }

    cal.setTime(dayStartDate);
    int week = cal.get(Calendar.WEEK_OF_MONTH);
    int day = cal.get(Calendar.DAY_OF_WEEK);

    String unixStartTime = Long.toString(
    dayStartDate.getTime() / 1000);
    String unixEndTime = Long.toString(
    dayEndDate.getTime() / 1000);

    String sql = "SELECT TS.`task_id` AS `_id`,
     TS.`task_name`, TS.`task_desc`, TS.`
    task_priority`, " +
        "TS1.`repeat_timeStart`, TS1.`
    repeat_timeEnd`, TS1.`repeat_start`, TS1.`
    repeat_end`, " +
        "TC.`category_desc` FROM `tasks` TS " +
        "JOIN `tasks_sched` TS1 JOIN `
    task_categories` TC " +
        "ON TS.`task_id` = TS1.`task_id` AND "
    +
        "TC.`category_id` = TS.`
    task_category_id` WHERE " +
        "(( ? - repeat_end >= 0 AND
    repeat_start - ? >= 0)" +
        " OR " +
        "( " +
        " (repeat_year = ? OR repeat_year = '*'
     ) " +
        " AND (repeat_month = ? OR repeat_month
     = '*' ) " +
        " AND (repeat_day = ? OR repeat_day =
    '*' ) " +
        " AND (repeat_week = ? OR repeat_week =
     '*' ) " +
        " AND (repeat_weekday = ? OR
    repeat_weekday = '*' ) " +
        " AND repeat_start <= ? AND repeat_end
    >= ? )"
        + ") ORDER BY `task_priority` ASC ";
    SQLiteDatabase db = this.
    getWritableDatabase();
    Cursor cursor = db.rawQuery(sql, new String
    [] { unixEndTime, unixStartTime,
        Integer.toString(currYear), Integer.
    toString(currMonth), Integer.toString(
    currDay),
        Integer.toString(week), Integer.
    toString(day), unixStartTime, unixEndTime
    });
    return cursor;
  }

  public boolean updateTask(ObjectTask
    objectTask) {

    ContentValues values = new ContentValues();
    SQLiteDatabase db = this.
    getWritableDatabase();
    String sql = "SELECT TC.`category_id` FROM
    `task_categories` TC WHERE category_desc =
    ?";
    Cursor cursor = db.rawQuery(sql, new String
    []{objectTask.getTask_category()});
    int category_id = 0;
    if (cursor.moveToFirst()) {
      category_id = Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("
    category_id")));
    }

    values.put("task_name", objectTask.
    getTask_name());
    values.put("task_desc", objectTask.
    getTask_description());
    values.put("task_category_id", category_id)
    ;
    values.put("task_priority", objectTask.
    getTask_priority());
    values.put("task_repeat", objectTask.
    getTask_repeat());

    String where = "task_id = ?";
    String[] whereArgs = { Integer.toString(
    objectTask.getId()) };

    boolean updateSuccessfulTasks = db.update("
    tasks", values, where, whereArgs) > 0;
    boolean deleteTaskSchedSuccess = false;
    boolean updateSuccessfulMeta = false;

    if(updateSuccessfulTasks){
      deleteTaskSchedSuccess = db.delete("
    tasks_sched", "task_id ='" + objectTask.
    getId() + "'", null) > 0;
      if(deleteTaskSchedSuccess){
        values = new ContentValues();
        values.put("task_id", objectTask.getId
    ());
        values.put("alarm_set", objectTask.
    getAlarmSet());
        values.put("motiv_set", objectTask.
    getAlarmSet());
        values.put("repeat_start", objectTask.
    getRepeat_start());
        values.put("repeat_end", objectTask.
    getRepeat_end());
        values.put("repeat_timeStart",
    objectTask.getRepeat_timeStart());
        values.put("repeat_timeEnd", objectTask
    .getRepeat_timeEnd());
        values.put("repeat_year", objectTask.
    getTask_repeat_year());
        values.put("repeat_month", objectTask.
    getTask_repeat_month());
        values.put("repeat_day", objectTask.
    getTask_repeat_day());
        values.put("repeat_week", objectTask.
    getTask_repeat_week());
        values.put("repeat_weekday", objectTask
    .getTask_repeat_weekday());
        updateSuccessfulMeta = db.insert("
    tasks_sched", null, values) > 0;
      }
    }
    Calendar calStart = Calendar.getInstance();
    Calendar calEnd = Calendar.getInstance();
    calStart.setTimeInMillis(objectTask.
    getRepeat_start()*1000);
    calEnd.setTimeInMillis(objectTask.
    getRepeat_end()*1000);
    boolean populateTasksMeta =
    populateTasksMeta(true,objectTask.getId(),
    calStart.get(Calendar.YEAR),calStart.get(
    Calendar.MONTH),
        calStart.get(Calendar.DAY_OF_MONTH),
    calEnd.get(Calendar.YEAR),calEnd.get(
    Calendar.MONTH),calEnd.get(Calendar.
    DAY_OF_MONTH),
        objectTask.getAlarmSet(),objectTask.
    getMotivSet(),0,objectTask.getTask_repeat()
    );
    cursor.close();
    return updateSuccessfulTasks &&
    updateSuccessfulMeta && populateTasksMeta;
  }

  public boolean deleteTask(int id) {
    boolean deleteTaskMetaSuccess = false;
    boolean deleteTaskSuccess = false;
    boolean deleteTaskSchedSuccess = false;

    SQLiteDatabase db = this.
    getWritableDatabase();

    deleteTaskSuccess = db.delete("tasks", "
    task_id = ?", new String[] {Integer.
    toString(id)}) > 0;
    deleteTaskSchedSuccess = db.delete("
    tasks_sched", "task_id = ?", new String[] {
    Integer.toString(id)}) > 0;
    deleteTaskMetaSuccess = db.delete("
    tasks_meta", "task_id = ?", new String[] {
    Integer.toString(id)}) > 0;

    return deleteTaskSuccess &&
    deleteTaskSchedSuccess &&
    deleteTaskMetaSuccess;
  }

  public ObjectTask readSingleRecord(int taskId
    ) {
    ObjectTask objectTask = new ObjectTask();
    String sql = "SELECT TS.* , TC.`
    category_desc` FROM `tasks` TS JOIN `
    task_categories` TC WHERE TS.`task_id` = "
    + taskId +
        " AND TC.`category_id` = TS.`
    task_category_id`";
    SQLiteDatabase db = this.
    getWritableDatabase();
    Cursor cursor = db.rawQuery(sql, null);

    if (cursor.moveToFirst()) {
      objectTask.setId(Integer.parseInt(cursor.
    getString(cursor.getColumnIndex("task_id"))
    ));
      objectTask.setTask_name(cursor.getString(
    cursor.getColumnIndex("task_name")));
      objectTask.setTask_description(cursor.
    getString(cursor.getColumnIndex("task_desc"
    )));
      objectTask.setTask_category(cursor.
    getString(cursor.getColumnIndex("
    category_desc")));
      objectTask.setTask_priority(cursor.getInt
    (cursor.getColumnIndex("task_priority")));
      objectTask.setTask_repeat(cursor.
    getString(cursor.getColumnIndex("
    task_repeat")));
      objectTask.setToCal(Integer.parseInt(
    cursor.getString(cursor.getColumnIndex("
    task_to_gcal"))));
    }

    sql = "SELECT TS.* FROM `tasks_sched` TS
    WHERE TS.`task_id` = ?";
    cursor = db.rawQuery(sql, new String[] {
    Integer.toString(taskId)});

    if (cursor.moveToFirst()) {
      //Alarm Set
      objectTask.setAlarmSet(Integer.parseInt(
    cursor.getString(cursor.getColumnIndex("
    alarm_set"))));
      //Motiv Set
      objectTask.setMotivSet(Integer.parseInt(
    cursor.getString(cursor.getColumnIndex("
    motiv_set"))));
      //Start Date
      objectTask.setRepeat_start(Long.parseLong
    (cursor.getString(cursor.getColumnIndex("
    repeat_start"))));
      //End Date
      objectTask.setRepeat_end(Long.parseLong(
    cursor.getString(cursor.getColumnIndex("
    repeat_end"))));
      //Start Time
      objectTask.setRepeat_timeStart(Long.
    parseLong(cursor.getString(cursor.
    getColumnIndex("repeat_timeStart"))));
      //End Time
      objectTask.setRepeat_timeEnd(Long.
    parseLong(cursor.getString(cursor.
    getColumnIndex("repeat_timeEnd"))));
      //Year
      objectTask.setTask_repeat_year(cursor.
    getString(cursor.getColumnIndex("
    repeat_year")));
      //Month
      objectTask.setTask_repeat_month(cursor.
    getString(cursor.getColumnIndex("
    repeat_month")));
      //Day
      objectTask.setTask_repeat_day(cursor.
    getString(cursor.getColumnIndex("repeat_day
    ")));
      //Week
      objectTask.setTask_repeat_week(cursor.
    getString(cursor.getColumnIndex("
    repeat_week")));
      //Weekday
      objectTask.setTask_repeat_weekday(cursor.
    getString(cursor.getColumnIndex("
    repeat_weekday")));
    }
    cursor.close();
    return objectTask;
  }

  //TASK META
    ----------------------------------------------------
```

```
public boolean startTaskMeta(int taskId, int
  month, int day, int year){

  SQLiteDatabase db = this.
  getWritableDatabase();
  ObjectTask obj = this.readSingleRecord(
  taskId);
  ContentValues values = new ContentValues();
  Calendar calTime = Calendar.getInstance();
  calTime.setTimeZone(TimeZone.getDefault());
  cal.setTimeInMillis(0);
  cal.set(Calendar.YEAR, calTime.get(Calendar
  .YEAR));
  cal.set(Calendar.MONTH, calTime.get(
  Calendar.MONTH));
  cal.set(Calendar.DAY_OF_MONTH, calTime.get(
  Calendar.DAY_OF_MONTH));

  Long startDate = cal.getTimeInMillis()
  /1000;

  cal.setTimeInMillis(0);
  cal.set(Calendar.HOUR_OF_DAY,calTime.get(
  Calendar.HOUR_OF_DAY));
  cal.set(Calendar.MINUTE,calTime.get(
  Calendar.MINUTE));

  Long startTime = cal.getTimeInMillis()
  /1000;

  cal.setTimeInMillis(0);
  cal.set(year,month,day,0,0,0);
  Long date = cal.getTimeInMillis()/1000;

  values.put("est_start", obj.
  getRepeat_timeStart());
  values.put("act_start", startTime);
  values.put("act_start_date", startDate);
  values.put("time_diff_start", obj.
  getRepeat_timeStart() - startTime);

  String where = "task_id = ? AND date = ?";
  String[] whereArgs = { Integer.toString(
  taskId), Long.toString(date) };

  boolean updateSuccessfulTasksMeta = db.
  update("tasks_meta", values, where,
  whereArgs) > 0;
  return updateSuccessfulTasksMeta;
}

public boolean endTaskMeta(int taskId, int
  month, int day, int year){

  SQLiteDatabase db = this.
  getWritableDatabase();

  ObjectTask obj = this.readSingleRecord(
  taskId);
  ContentValues values = new ContentValues();
  Calendar calTime = Calendar.getInstance();
  calTime.setTimeZone(TimeZone.getDefault());
  cal.setTimeInMillis(0);
  cal.set(Calendar.YEAR, calTime.get(Calendar
  .YEAR));
  cal.set(Calendar.MONTH, calTime.get(
  Calendar.MONTH));
  cal.set(Calendar.DAY_OF_MONTH, calTime.get(
  Calendar.DAY_OF_MONTH));

  Long endDate = cal.getTimeInMillis()/1000;

  cal.setTimeInMillis(0);
  cal.set(Calendar.HOUR_OF_DAY,calTime.get(
  Calendar.HOUR_OF_DAY));
  cal.set(Calendar.MINUTE,calTime.get(
  Calendar.MINUTE));

  Long endTime = cal.getTimeInMillis()/1000;

  cal.setTimeInMillis(0);
  cal.set(year,month,day,0,0,0);
  Long date = cal.getTimeInMillis()/1000;
  values.put("task_finished", 1);
  values.put("est_end", obj.getRepeat_timeEnd
  ());
  values.put("act_end", endTime);
  values.put("act_end_date", endDate);
  values.put("time_diff_end", obj.
  getRepeat_timeEnd() - endTime);

  String where = "task_id = ? AND date = ?";
  String[] whereArgs = { Integer.toString(
  taskId), Long.toString(date) };

  boolean updateSuccessfulTasksMeta = db.
  update("tasks_meta", values, where,
  whereArgs) > 0;
  return updateSuccessfulTasksMeta;
}

public ObjectTaskMeta getTaskMetaDetails(int
  taskId, Long currDay){
  ObjectTaskMeta taskMeta = new
  ObjectTaskMeta();
  String sql = "SELECT TM.* FROM `tasks_meta`
   TM WHERE TM.`task_id` = ? AND TM.`date` =
  ?";
  SQLiteDatabase db = this.
  getWritableDatabase();
  Cursor cursor = db.rawQuery(sql, new String
  [] {Integer.toString(taskId), Long.toString
  (currDay)});
  if (cursor.moveToFirst()) {
    if(cursor.getString(cursor.getColumnIndex
  ("alarm_set"))!=null){
      taskMeta.setAlarmSet(Integer.parseInt(
  cursor.getString(cursor.getColumnIndex("
  alarm_set"))));
    }else{
      String sql2 = "SELECT TS.`alarm_set`
  FROM `tasks_sched` TS WHERE TS.`task_id` =
  ?";
      Cursor cursor2 = db.rawQuery(sql2, new
  String[] {Integer.toString(taskId)});
      if(cursor2.moveToFirst()){
        taskMeta.setAlarmSet(Integer.parseInt
  (cursor2.getString(cursor2.getColumnIndex("
  alarm_set"))));
      }
    }
    if(cursor.getString(cursor.getColumnIndex
  ("motiv_set"))!=null){
      taskMeta.setMotivSet(Integer.parseInt(
  cursor.getString(cursor.getColumnIndex("
  motiv_set"))));
    }else{
      String sql2 = "SELECT TS.`motiv_set`
  FROM `tasks_sched` TS WHERE TS.`task_id` =
  ?";
      Cursor cursor2 = db.rawQuery(sql2, new
  String[] {Integer.toString(taskId)});
      if(cursor2.moveToFirst()){
        taskMeta.setMotivSet(Integer.parseInt
  (cursor2.getString(cursor2.getColumnIndex("
  motiv_set"))));
      }
    }
    taskMeta.setRequestID(Integer.parseInt(
  cursor.getString(cursor.getColumnIndex("
  request_id"))));
    taskMeta.setMotivRequestID(Integer.
  parseInt(cursor.getString(cursor.
  getColumnIndex("request_motiv_id"))));
    if(cursor.getString(cursor.getColumnIndex
  ("task_id"))!=null){
      taskMeta.setTaskID(Integer.parseInt(
  cursor.getString(cursor.getColumnIndex("
  task_id"))));
    }else{
      taskMeta.setTaskID(taskId);
    }
    if(cursor.getString(cursor.getColumnIndex
  ("date"))!=null){
      taskMeta.setDate(Long.parseLong(cursor.
  getString(cursor.getColumnIndex("date"))));
    }else{
        taskMeta.setDate(currDay);
      }
      if(cursor.getString(cursor.getColumnIndex
    ("est_start"))!=null){
        taskMeta.setEst_start(Long.parseLong(
    cursor.getString(cursor.getColumnIndex("
    est_start"))));
      }else{
        taskMeta.setEst_start(0L);
      }
      if(cursor.getString(cursor.getColumnIndex
    ("est_end"))!=null){
        taskMeta.setEst_end(Long.parseLong(
    cursor.getString(cursor.getColumnIndex("
    est_end"))));
      }else{
        taskMeta.setEst_end(0L);
      }
      if(cursor.getString(cursor.getColumnIndex
    ("act_start"))!=null){
        taskMeta.setAct_start(Long.parseLong(
    cursor.getString(cursor.getColumnIndex("
    act_start"))));
      }else{
        taskMeta.setAct_start(0L);
      }
      if(cursor.getString(cursor.getColumnIndex
    ("act_end"))!=null){
        taskMeta.setAct_end(Long.parseLong(
    cursor.getString(cursor.getColumnIndex("
    act_end"))));
      }else{
        taskMeta.setAct_end(0L);
      }
      if(cursor.getString(cursor.getColumnIndex
    ("time_diff_start"))!=null){
        taskMeta.setTime_diff_start(Long.
    parseLong(cursor.getString(cursor.
    getColumnIndex("time_diff_start"))));
      }else{
        taskMeta.setTime_diff_start(0L);
      }
      if(cursor.getString(cursor.getColumnIndex
    ("time_diff_end"))!=null){
        taskMeta.setTime_diff_end(Long.
    parseLong(cursor.getString(cursor.
    getColumnIndex("time_diff_end"))));
      }else{
        taskMeta.setTime_diff_end(0L);
      }
    }else{
      String sql2 = "SELECT TS.`alarm_set`, TS
    .`motiv_set` FROM `tasks_sched` TS WHERE TS
    .`task_id` = ?";
      Cursor cursor2 = db.rawQuery(sql2, new
    String[] {Integer.toString(taskId)});
      if(cursor2.moveToFirst()){
        taskMeta.setAlarmSet(Integer.parseInt(
    cursor2.getString(cursor2.getColumnIndex("
    alarm_set"))));
        taskMeta.setMotivSet(Integer.parseInt(
    cursor2.getString(cursor2.getColumnIndex("
    motiv_set"))));
      }
      taskMeta.setTaskID(taskId);
      taskMeta.setDate(currDay);
      taskMeta.setEst_start(0L);
      taskMeta.setEst_end(0L);
      taskMeta.setAct_start(0L);
      taskMeta.setAct_end(0L);
      taskMeta.setTime_diff_start(0L);
      taskMeta.setTime_diff_end(0L);
    }
    cursor.close();
    return taskMeta;
  }

  public boolean isStartedTask(int taskId, int
    month, int day, int year){
    SQLiteDatabase db = this.
    getWritableDatabase();
    cal.setTimeInMillis(0);
    cal.set(year,month,day,0,0,0);
    Long date = cal.getTimeInMillis()/1000;

    String sql = "SELECT TM.`act_start` FROM `
    tasks_meta` TM WHERE TM.`task_id` = ? AND
    TM.`date` = ?";
    Cursor cursor = db.rawQuery(sql, new String
    []{Integer.toString(taskId), Long.toString(
    date)});
    boolean isStarted = false;
    if(cursor.moveToFirst()) {
      if(cursor.getString(cursor.getColumnIndex
    ("act_start"))!=null){
        isStarted = true;
      }
    }
    return isStarted;
  }

}
```

AddToCalendar.java

```
package com.specialproblem.brendz.
    timemanagement.googlecalendar;

import com.google.android.gms.common.
    ConnectionResult;
import com.google.android.gms.common.
    GoogleApiAvailability;
import com.google.android.gms.common.api.
    BooleanResult;
import com.google.api.client.extensions.android
    .http.AndroidHttp;
import com.google.api.client.googleapis.
    extensions.android.gms.auth.
    GoogleAccountCredential;
import com.google.api.client.googleapis.
    extensions.android.gms.auth.
    GooglePlayServicesAvailabilityIOException;
import com.google.api.client.googleapis.
    extensions.android.gms.auth.
    UserRecoverableAuthIOException;

import com.google.api.client.http.HttpTransport
    ;
import com.google.api.client.json.JsonFactory;
import com.google.api.client.json.jackson2.
    JacksonFactory;
import com.google.api.client.util.
    ExponentialBackOff;

import com.google.api.services.calendar.
    CalendarScopes;
import com.google.api.client.util.DateTime;

import com.google.api.services.calendar.model
    .*;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;

import android.Manifest;
import android.accounts.AccountManager;
import android.app.Activity;
import android.app.Dialog;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.widget.Toast;

import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.*;
import java.util.Calendar;

import pub.devrel.easypermissions.
    AfterPermissionGranted;
import pub.devrel.easypermissions.
    EasyPermissions;

/**
 * Activity that adds/deletes tasks in Google
    Calendar
 */
```

```
public class AddToCalendar extends Activity
    implements EasyPermissions.
    PermissionCallbacks {

  GoogleAccountCredential mCredential;
  ProgressDialog mProgress;

  private TableController tblController;
  private int taskId;
  private String task_name;
  private String task_desc;
  private String repeat_type;
  private Long start_date;
  private Long end_date;
  private int start_time;
  private int end_time;
  private int to_gcal;

  static final int REQUEST_ACCOUNT_PICKER =
    1000;
  static final int REQUEST_AUTHORIZATION =
    1001;
  static final int REQUEST_GOOGLE_PLAY_SERVICES
     = 1002;
  static final int
    REQUEST_PERMISSION_GET_ACCOUNTS = 1003;

  private static final String PREF_ACCOUNT_NAME
     = "accountName";
  private static final String[] SCOPES = {
    CalendarScopes.CALENDAR};

  /**
     * Create the main activity.
     * @param savedInstanceState previously
         saved instance data.
     */
  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);

    Bundle extras = getIntent().getExtras();
    if(extras!=null){
      taskId = extras.getInt("TASK_ID");
      task_name = extras.getString("TASK_NAME")
    ;
      task_desc = extras.getString("TASK_DESC")
    ;
      repeat_type = extras.getString("
    REPEAT_TYPE");
      start_date = extras.getLong("START_DATE")
    ;
      end_date = extras.getLong("END_DATE");
      start_time = extras.getInt("START_TIME");
      end_time = extras.getInt("END_TIME");
      to_gcal = extras.getInt("TO_GCAL");
    }
    tblController = TimeManagement.
    tblController;

    mProgress = new ProgressDialog(this);

    // Initialize credentials and service
    object.
    mCredential = GoogleAccountCredential.
    usingOAuth2(
        getApplicationContext(), Arrays.asList(
    SCOPES))
        .setBackOff(new ExponentialBackOff());

    getResultsFromApi();
  }

  /**
     * Attempt to call the API, after verifying
         that all the preconditions are
     * satisfied. The preconditions are: Google
         Play Services installed, an
     * account was selected and the device
         currently has online access. If any
     * of the preconditions are not satisfied,
         the app will prompt the user as
     * appropriate.
     */
  private void getResultsFromApi() {
    if (! isGooglePlayServicesAvailable()) {
      acquireGooglePlayServices();
    } else if (mCredential.
  getSelectedAccountName() == null) {
      chooseAccount();
    } else if (! isDeviceOnline()) {
      Toast.makeText(AddToCalendar.this, "No
  network connection available.", Toast.
  LENGTH_SHORT).show();
      finish();
    } else {
      System.out.println("UMABOT!!");
      new MakeRequestTask(mCredential).execute
  ();
    }
}
/**
   * Attempts to set the account used with
       the API credentials. If an account
   * name was previously saved it will use
       that one; otherwise an account
   * picker dialog will be shown to the user.
       Note that the setting the
   * account to use with the credentials
       object requires the app to have the
   * GET_ACCOUNTS permission, which is
       requested here if it is not already
   * present. The AfterPermissionGranted
       annotation indicates that this
   * function will be rerun automatically
       whenever the GET_ACCOUNTS permission
   * is granted.
   */
@AfterPermissionGranted(
  REQUEST_PERMISSION_GET_ACCOUNTS)
private void chooseAccount() {
  if (EasyPermissions.hasPermissions(
      this, Manifest.permission.GET_ACCOUNTS)
  ) {
    String accountName = getPreferences(
  Context.MODE_PRIVATE)
        .getString(PREF_ACCOUNT_NAME, null);
    if (accountName != null) {
      mCredential.setSelectedAccountName(
  accountName);
      getResultsFromApi();
    } else {
      // Start a dialog from which the user
  can choose an account
      startActivityForResult(
          mCredential.newChooseAccountIntent
  (),
          REQUEST_ACCOUNT_PICKER);
    }
  } else {
    // Request the GET_ACCOUNTS permission
  via a user dialog
    EasyPermissions.requestPermissions(
        this,
        "This app needs to access your Google
   account (via Contacts).",
        REQUEST_PERMISSION_GET_ACCOUNTS,
        Manifest.permission.GET_ACCOUNTS);
  }
}

/**
   * Called when an activity launched here (
       specifically, AccountPicker
   * and authorization) exits, giving you the
       requestCode you started it with,
   * the resultCode it returned, and any
       additional data from it.
   * @param requestCode code indicating which
       activity result is incoming.
   * @param resultCode code indicating the
       result of the incoming
   *     activity result.
   * @param data Intent (containing result
       data) returned by incoming
   *     activity result.
   */
@Override
protected void onActivityResult(
    int requestCode, int resultCode, Intent
  data) {
  super.onActivityResult(requestCode,
  resultCode, data);
  switch(requestCode) {
    case REQUEST_GOOGLE_PLAY_SERVICES:
      if (resultCode != RESULT_OK) {
        Toast.makeText(this, "This app
  requires Google Play Services. Please
  install " +
            "Google Play Services on your
  device and relaunch this app.", Toast.
  LENGTH_SHORT).show();
      } else {
        getResultsFromApi();
      }
      break;
    case REQUEST_ACCOUNT_PICKER:
      if (resultCode == RESULT_OK && data !=
  null &&
          data.getExtras() != null) {
        String accountName =
            data.getStringExtra(
  AccountManager.KEY_ACCOUNT_NAME);
        System.out.println("PUMASOK OH");
        if (accountName != null) {
          SharedPreferences settings =
              getPreferences(Context.
  MODE_PRIVATE);
          SharedPreferences.Editor editor =
  settings.edit();
          editor.putString(PREF_ACCOUNT_NAME,
   accountName);
          editor.apply();
          mCredential.setSelectedAccountName(
  accountName);
          getResultsFromApi();
        }
      }
      break;
    case REQUEST_AUTHORIZATION:
      if (resultCode == RESULT_OK) {
        getResultsFromApi();
      }
      break;
  }
}

/**
   * Respond to requests for permissions at
       runtime for API 23 and above.
   * @param requestCode The request code
       passed in
   *     requestPermissions(android.app.
       Activity, String, int, String[])
   * @param permissions The requested
       permissions. Never null.
   * @param grantResults The grant results
       for the corresponding permissions
   *     which is either PERMISSION_GRANTED
       or PERMISSION_DENIED. Never null.
   */
@Override
public void onRequestPermissionsResult(int
  requestCode,
                  @NonNull String[]
  permissions,
                  @NonNull int[]
  grantResults) {
  super.onRequestPermissionsResult(
  requestCode, permissions, grantResults);
  EasyPermissions.onRequestPermissionsResult(
      requestCode, permissions, grantResults,
   this);
}

/**
   * Callback for when a permission is
       granted using the EasyPermissions
   * library.
   * @param requestCode The request code
       associated with the requested
   *         permission
   * @param list The requested permission
       list. Never null.
   */
@Override
public void onPermissionsGranted(int
  requestCode, List<String> list) {
  // Do nothing.
}

/**
   * Callback for when a permission is denied
       using the EasyPermissions
   * library.
   * @param requestCode The request code
       associated with the requested
   *         permission
   * @param list The requested permission
       list. Never null.
   */
@Override
public void onPermissionsDenied(int
  requestCode, List<String> list) {
  // Do nothing.
}

/**
   * Checks whether the device currently has
       a network connection.
   * @return true if the device has a network
       connection, false otherwise.
   */
private boolean isDeviceOnline() {
  ConnectivityManager connMgr =
      (ConnectivityManager) getSystemService(
  Context.CONNECTIVITY_SERVICE);
  NetworkInfo networkInfo = connMgr.
  getActiveNetworkInfo();
  return (networkInfo != null && networkInfo.
  isConnected());
}

/**
   * Check that Google Play services APK is
       installed and up to date.
   * @return true if Google Play Services is
       available and up to
   *     date on this device; false otherwise
       .
   */
private boolean isGooglePlayServicesAvailable
  () {
  GoogleApiAvailability apiAvailability =
      GoogleApiAvailability.getInstance();
  final int connectionStatusCode =
      apiAvailability.
  isGooglePlayServicesAvailable(this);
  return connectionStatusCode ==
  ConnectionResult.SUCCESS;
}

/**
   * Attempt to resolve a missing, out-of-
       date, invalid or disabled Google
   * Play Services installation via a user
       dialog, if possible.
   */
private void acquireGooglePlayServices() {
  GoogleApiAvailability apiAvailability =
      GoogleApiAvailability.getInstance();
  final int connectionStatusCode =
      apiAvailability.
  isGooglePlayServicesAvailable(this);
  if (apiAvailability.isUserResolvableError(
  connectionStatusCode)) {

  showGooglePlayServicesAvailabilityErrorDialog
  (connectionStatusCode);
  }
}

/**
   * Display an error dialog showing that
       Google Play Services is missing
   * or out of date.
   * @param connectionStatusCode code
       describing the presence (or lack of)
   *     Google Play Services on this device.
   */
void
  showGooglePlayServicesAvailabilityErrorDialog
  (
    final int connectionStatusCode) {
  GoogleApiAvailability apiAvailability =
  GoogleApiAvailability.getInstance();
  Dialog dialog = apiAvailability.
  getErrorDialog(
      AddToCalendar.this,
      connectionStatusCode,
      REQUEST_GOOGLE_PLAY_SERVICES);
  dialog.show();
}

/**
   * An asynchronous task that handles the
       Google Calendar API call.
   * Placing the API calls in their own task
       ensures the UI stays responsive.
   */
private class MakeRequestTask extends
  AsyncTask<Void, Void, String> {
  private com.google.api.services.calendar.
  Calendar mService = null;
  private Exception mLastError = null;

  public MakeRequestTask(
  GoogleAccountCredential credential) {
    HttpTransport transport = AndroidHttp.
  newCompatibleTransport();
    JsonFactory jsonFactory = JacksonFactory.
  getDefaultInstance();
    mService = new com.google.api.services.
  calendar.Calendar.Builder(
        transport, jsonFactory, credential)
        .setApplicationName("TimeMgmt")
        .build();
  }

  /**
       * Background task to call Google
           Calendar API.
       * @param params no parameters needed
           for this task.
       */
  @Override
  protected String doInBackground(Void...
  params) {
    try {
      if(to_gcal==1){
        return createEvent();
      }else{
        return deleteEvent();
      }
    } catch (Exception e) {
      mLastError = e;
      cancel(true);
      return null;
    }
  }

  public String createEvent() {
    mProgress.setMessage("Pushing task to
  Google Calendar ...");
    SimpleDateFormat sdf = new
  SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ssZ",
  Locale.US) {
      public StringBuffer format(Date date,
  StringBuffer toAppendTo, java.text.
  FieldPosition pos) {
        StringBuffer toFix = super.format(
  date, toAppendTo, pos);
        return toFix.insert(toFix.length()-2,
   ':');
      }
    };
    sdf.setTimeZone(TimeZone.getDefault());
    Calendar calendar = Calendar.getInstance
  ();
    Calendar calTime = Calendar.getInstance()
  ;
    calendar.setTimeInMillis(0);
    calTime.setTimeInMillis(0);
    boolean result = false;

    Event event = new Event()
        .setSummary(task_name)
        .setDescription(task_desc);
    calendar.setTimeInMillis(start_date*1000)
  ;
    calTime.setTimeInMillis(start_time*1000);
    calendar.set(Calendar.HOUR_OF_DAY,
  calTime.get(Calendar.HOUR_OF_DAY));
    calendar.set(Calendar.MINUTE, calTime.get
  (Calendar.MINUTE));
    DateTime startDateTime = new DateTime(sdf
  .format(calendar.getTime()));
  EventDateTime start = new EventDateTime()
      .setDateTime(startDateTime)
      .setTimeZone(TimeZone.getDefault().
getID());
  event.setStart(start);
  calendar.setTimeInMillis(start_date*1000)
;
  calTime.setTimeInMillis(end_time*1000);
  calendar.set(Calendar.HOUR_OF_DAY,
calTime.get(Calendar.HOUR_OF_DAY));
  calendar.set(Calendar.MINUTE, calTime.get
(Calendar.MINUTE));
  DateTime endDateTime = new DateTime(sdf.
format(calendar.getTime()));
  EventDateTime end = new EventDateTime()
      .setDateTime(endDateTime)
      .setTimeZone(TimeZone.getDefault().
getID());
  event.setEnd(end);
  System.out.println("Time End: "+ sdf.
format(calendar.getTime()) + " " + calendar
.getTime());
  if(!repeat_type.equals("Once")){
    StringBuilder recur = new StringBuilder
();
    recur.append("RRULE:FREQ=");
    switch(repeat_type){
      case "Daily":
        recur.append("DAILY;");
        break;
      case "Weekly":
        recur.append("WEEKLY;");
        break;
      case "Monthly":
        recur.append("MONTHLY;");
        break;
    }
    calendar.setTimeInMillis(end_date*1000)
;
    calTime.setTimeInMillis(end_time*1000);
    calendar.set(Calendar.HOUR_OF_DAY,
calTime.get(Calendar.HOUR_OF_DAY));
    calendar.set(Calendar.MINUTE, calTime.
get(Calendar.MINUTE));
    SimpleDateFormat sdfRecurrence = new
SimpleDateFormat("yyyyMMdd'T'HHmmss'Z'",
Locale.US);
    recur.append("UNTIL=");
    recur.append(sdfRecurrence.format(
calendar.getTime()));
    String[] recurrence = new String[]{
recur.toString()};
    event.setRecurrence(Arrays.asList(
recurrence));
  }
  String calendarId = "primary";
  try {
    event = mService.events().insert(
calendarId, event).execute();
    result = true;
  } catch (IOException e) {
    e.printStackTrace();
  }
  boolean setToGoogleCalendar =
tblController.setToGoogleCalendar(taskId,
1);
  boolean setEventID = tblController.
setEventID(taskId, event.getId());

  if(result && setEventID &&
setToGoogleCalendar){
    return "add";
  } else {
    return "err";
  }
}

public String deleteEvent() {
  mProgress.setMessage("Deleting task from
Google Calendar...");
  String eventId = tblController.getEventID
(taskId);
  try {
    mService.events().delete("primary",
eventId).execute();
      } catch (IOException e) {
        e.printStackTrace();
      }
      boolean resetToGoogleCalendar =
tblController.setToGoogleCalendar(taskId,
0);
      boolean resetEventID = tblController.
setEventID(taskId, null);
      if(resetToGoogleCalendar && resetEventID)
{
        return "del";
      } else {
        return "err";
      }
    }

    @Override
    protected void onPreExecute() {
      mProgress.show();
    }

    @Override
    protected void onPostExecute(String output)
 {
      mProgress.hide();
      if (output.equals("add")) {
        Toast.makeText(AddToCalendar.this, "
Added to Google Calendar.", Toast.
LENGTH_SHORT).show();
        finish();
      } else if(output.equals("del")){
        Toast.makeText(AddToCalendar.this, "
Task deleted from Google Calendar", Toast.
LENGTH_SHORT).show();
        finish();
      } else {
        Toast.makeText(AddToCalendar.this, "
Operation error!", Toast.LENGTH_SHORT).show
();
        finish();
      }
    }

    @Override
    protected void onCancelled() {
      mProgress.hide();
      if (mLastError != null) {
        if (mLastError instanceof
GooglePlayServicesAvailabilityIOException)
{

showGooglePlayServicesAvailabilityErrorDialog
(
            ((
GooglePlayServicesAvailabilityIOException)
mLastError)
                .getConnectionStatusCode());
        } else if (mLastError instanceof
UserRecoverableAuthIOException) {
          startActivityForResult(
            ((UserRecoverableAuthIOException)
 mLastError).getIntent(),
            AddToCalendar.
REQUEST_AUTHORIZATION);
        } else {
          Toast.makeText(getApplicationContext
(),"The following error occurred:\n"
            + mLastError.getMessage(), Toast.
LENGTH_SHORT);
        }
      } else {
        Toast.makeText(getApplicationContext(),
"Request cancelled.", Toast.LENGTH_SHORT);
      }
    }
  }
}
```

AlarmReceiver.java

```
package com.specialproblem.brendz.
    timemanagement.notification;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;

/**
 * Receiver for AlarmService
 */
public class AlarmReceiver extends
    BroadcastReceiver {
  @Override
  public void onReceive(Context context, Intent
     intent) {
    Intent alarmService = new Intent(context,
    AlarmService.class);
    Bundle extras = intent.getExtras();
    if(extras!=null){
      alarmService.putExtra("TASK_ID",extras.
    getInt("TASK_ID"));
      alarmService.putExtra("TASK_NAME",extras.
    getString("TASK_NAME"));
      alarmService.putExtra("TASK_DESC",extras.
    getString("TASK_DESC"));
      alarmService.putExtra("START_TIME",extras
    .getInt("START_TIME"));
    }
    context.startService(alarmService);
  }
}
```

AlarmService.java

```
package com.specialproblem.brendz.
    timemanagement.notification;

import android.annotation.TargetApi;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.app.Service;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.IBinder;
import android.support.v4.app.
    NotificationCompat;
import android.support.v4.app.TaskStackBuilder;
import android.widget.Toast;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskMain;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskView;

import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Locale;
import java.util.Random;
import java.util.concurrent.atomic.
    AtomicInteger;

/**
 * Displays notification for task
 */
public class AlarmService extends Service {

  public static AtomicInteger
    notificationCounter;

  @Override
  public int onStartCommand(Intent intent, int
    flags, int startId) {
    Bundle extras = intent.getExtras();
    SimpleDateFormat sdf = new SimpleDateFormat
    ("hh:mm", Locale.US);

    int task_id = 0;
    String task_name = null;
    String task_desc = null;
    int start_time = 0;
if(extras!=null){
  task_id = extras.getInt("TASK_ID");
  task_name = extras.getString("TASK_NAME")
;
  task_desc = extras.getString("TASK_DESC")
;
  start_time = extras.getInt("START_TIME");
}

Uri sound = RingtoneManager.getDefaultUri(
RingtoneManager.TYPE_NOTIFICATION);
String title = getString(R.string.app_name)
;
Calendar cal = Calendar.getInstance();
Intent intentI = null;
Intent intentD = null;
if(notificationCounter.get()==0){
  intentI = new Intent(this, TaskView.class
);
  intentD = new Intent(this, TaskView.class
);
  intentI.putExtra("TASK_ID", task_id);
} else {
  intentI = new Intent(this, TaskMain.class
);
  intentD = new Intent(this, TaskMain.class
);
}
intentI.putExtra("CURR_MONTH", cal.get(
Calendar.MONTH)+1);
intentI.putExtra("CURR_DAY", cal.get(
Calendar.DAY_OF_MONTH));
intentI.putExtra("CURR_YEAR", cal.get(
Calendar.YEAR));
intentI.putExtra("DELETE", 0);
intentD.putExtra("DELETE", 1);

int requestID_I = (int) System.
currentTimeMillis();
int requestID_D = 1000;
PendingIntent pIntent = PendingIntent.
getActivity(this, requestID_I, intentI,
PendingIntent.FLAG_UPDATE_CURRENT);
int notificationNumber =
notificationCounter.incrementAndGet();
Notification notification = null;
cal.setTimeInMillis(0);
cal.set(Calendar.MILLISECOND, start_time
*1000);
String am_pm = null;
if(cal.get(Calendar.AM_PM)==0){
  am_pm = " AM";
}else{
  am_pm = " PM";
}
NotificationCompat.Builder notifBuilder =
new NotificationCompat.Builder(this)
     .setSmallIcon(R.drawable.ic_add_btn)
     .setContentTitle(title)
     .setContentText(task_name + " - " +
task_desc + " at " + sdf.format(cal.
getTimeInMillis()) + am_pm)
     .setTicker("New Imminent Task: " +
task_name)
     .setContentIntent(pIntent)
     .setDeleteIntent(PendingIntent.
getActivity(this, requestID_D, intentD,
PendingIntent.FLAG_CANCEL_CURRENT))
     .setSound(sound)
     .setVibrate(new long[] { 1000, 1000,
1000, 1000, 1000 })
     .setNumber(notificationNumber);

if (Build.VERSION.SDK_INT > 15) {
  notification = buildSDK15(notifBuilder);
} else {
  notification = notifBuilder.build();
}
// hide the notification after its selected
notification.flags |= Notification.
FLAG_AUTO_CANCEL;

NotificationManager notificationManager = (
NotificationManager) getSystemService(
    NOTIFICATION_SERVICE);
    // cancel previous notification to clean up
     garbage in the status bar
    notificationManager.cancel(
    notificationNumber - 1);
    // add new notification
    notificationManager.notify(
    notificationNumber, notification);
    return 0;
  }

  @Override
  public void onCreate(){
    notificationCounter = new AtomicInteger();
    notificationCounter.set(0);
  }

  @Override
  public IBinder onBind(Intent intent) {
    return null;
  }

  @TargetApi(Build.VERSION_CODES.JELLY_BEAN)
  private Notification buildSDK15(
    NotificationCompat.Builder notifBuilder){
    notifBuilder.setPriority(Notification.
    PRIORITY_HIGH);
    return notifBuilder.build();
  }
}
```

AlarmSetter.java

```
package com.specialproblem.brendz.
    timemanagement.notification;

import android.app.AlarmManager;
import android.app.PendingIntent;
import android.app.Service;
import android.content.Intent;
import android.database.Cursor;
import android.os.IBinder;
import android.widget.Toast;

import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;

import java.util.Calendar;

/**
 * Service that will set all alarms in the
     Alarm Manager
 */
public class AlarmSetter extends Service {

  private TableController tblController;
  private Intent intentAR;
  private Intent intentMotiv;
  private PendingIntent pIntent;
  private PendingIntent pMotivIntent;
  private AlarmManager alarmManager;

  @Override
  public IBinder onBind(Intent intent) {
    return null;
  }

  @Override
  public void onCreate() {
    tblController = TimeManagement.
    tblController;
    alarmManager = TimeManagement.alarmManager;

    Calendar cal = Calendar.getInstance();
    Calendar calDate = Calendar.getInstance();

    Cursor cursor = tblController.setAlarm(cal.
    get(Calendar.MONTH)+1, cal.get(Calendar.
    DATE), cal.get(Calendar.YEAR));
    if (cursor.moveToFirst()) {
      do {
        intentAR = new Intent(this,
    AlarmReceiver.class);
        intentMotiv = new Intent(this,
    MotivationReceiver.class);
    int alarmSet;
    int motivSet;
    int toDelete = 0;
    int requestId = (int) System.
currentTimeMillis();
    int requestMotivId = requestId*-1;
    int month = calDate.get(Calendar.MONTH)
;
    int day = calDate.get(Calendar.
DAY_OF_MONTH);
    int year = calDate.get(Calendar.YEAR);
    int taskId = Integer.parseInt(cursor.
getString(cursor.getColumnIndex("_id")));
    String taskName = cursor.getString(
cursor.getColumnIndex("task_name"));
    String taskDesc = cursor.getString(
cursor.getColumnIndex("task_desc"));
    int timeStart = Integer.parseInt(cursor
.getString(cursor.getColumnIndex("
repeat_timeStart")));
    int categoryId = Integer.parseInt(
cursor.getString(cursor.getColumnIndex("
task_category_id")));
    Cursor realAlarmCursor = tblController.
setRealAlarm(taskId, month+1, day, year);
    if(realAlarmCursor.moveToFirst()){
      alarmSet = Integer.parseInt(
realAlarmCursor.getString(realAlarmCursor.
getColumnIndex("_real_alarm")));
      requestId = Integer.parseInt(
realAlarmCursor.getString(realAlarmCursor.
getColumnIndex("request_id")));
      motivSet = Integer.parseInt(
realAlarmCursor.getString(realAlarmCursor.
getColumnIndex("motiv_set")));
      requestMotivId = Integer.parseInt(
realAlarmCursor.getString(realAlarmCursor.
getColumnIndex("request_motiv_id")));
      toDelete = Integer.parseInt(
realAlarmCursor.getString(realAlarmCursor.
getColumnIndex("to_delete")));
    }else{
      alarmSet = Integer.parseInt(cursor.
getString(cursor.getColumnIndex("alarm_set"
)));
      motivSet = Integer.parseInt(cursor.
getString(cursor.getColumnIndex("motiv_set"
)));
    }
    intentAR.putExtra("TASK_ID", taskId);
    intentAR.putExtra("TASK_NAME", taskName
);
    intentAR.putExtra("TASK_DESC", taskDesc
);
    intentAR.putExtra("START_TIME",
timeStart);
    intentMotiv.putExtra("CATEGORY_ID",
categoryId);
    intentMotiv.putExtra("CATEGORY", 1);
    if (alarmSet == 1) {
      tblController.updateRequestID(taskId,
 month, day, year, requestId, 1, 0);
      cal.setTimeInMillis(0);
      cal.set(Calendar.YEAR, year);
      cal.set(Calendar.MONTH, month);
      cal.set(Calendar.DAY_OF_MONTH, day);
      cal.add(Calendar.MILLISECOND,
timeStart * 1000);
      cal.add(Calendar.MINUTE, -1);
    } else {
      tblController.updateRequestID(taskId,
 month, day, year, requestId, 0, 0);
    }
    if(motivSet==1){
      tblController.updateRequestMotivID(
taskId, month, day, year, requestMotivId,
1);
    }else{
      tblController.updateRequestMotivID(
taskId, month, day, year, requestMotivId,
0);
    }
    pIntent = PendingIntent.getBroadcast(
this, requestId, intentAR, PendingIntent.
FLAG_UPDATE_CURRENT);
    pMotivIntent = PendingIntent.
getBroadcast(this, requestMotivId,
intentMotiv, PendingIntent.
FLAG_UPDATE_CURRENT);
    if(alarmSet==1){
      if(cal.getTimeInMillis()>System.
currentTimeMillis()){
        alarmManager.set(AlarmManager.
RTC_WAKEUP, cal.getTimeInMillis(), pIntent)
;
      }
      if(motivSet==1){
        cal.add(Calendar.MINUTE, -4);
        if(cal.getTimeInMillis()>System.
currentTimeMillis()){
          alarmManager.set(AlarmManager.
RTC_WAKEUP, cal.getTimeInMillis(),
pMotivIntent);
        }
      }else{
        alarmManager.cancel(pMotivIntent);
        pMotivIntent.cancel();
      }
    } else {
      alarmManager.cancel(pIntent);
      pIntent.cancel();
      if(motivSet==1){
        cal.add(Calendar.MINUTE, -29);
        if(cal.getTimeInMillis()>System.
currentTimeMillis()){
          alarmManager.set(AlarmManager.
RTC_WAKEUP, cal.getTimeInMillis(),
pMotivIntent);
        }
      }
      if(toDelete==1){
        alarmManager.cancel(pMotivIntent);
        pMotivIntent.cancel();
        boolean deleteSuccessful =
tblController.deleteTask(taskId);
        if (deleteSuccessful){
          Toast.makeText(this, "Task record
 was deleted.", Toast.LENGTH_SHORT).show();
        }else{
          Toast.makeText(this, "Unable to
delete task record.", Toast.LENGTH_SHORT).
show();
        }
      }
    }
  } while (cursor.moveToNext());
}
int[] res = tblController.getMotivationMeta
();
Intent motivIntent = new Intent(this,
MotivationReceiver.class);
motivIntent.putExtra("CATEGORY", 0);
PendingIntent motivPIntent = PendingIntent.
getBroadcast(this, 1000, motivIntent,
PendingIntent.FLAG_UPDATE_CURRENT);
if(res[0]==1){
  Calendar motivCal = Calendar.getInstance
();
  motivCal.setTimeInMillis(0);
  motivCal.set(calDate.get(Calendar.YEAR),
calDate.get(Calendar.MONTH), calDate.get(
Calendar.DAY_OF_MONTH),0,0,0);
  motivCal.add(Calendar.MILLISECOND, res
[1]*1000);
  motivCal.add(Calendar.HOUR, 8);
  if(motivCal.getTimeInMillis()>System.
currentTimeMillis()){
    alarmManager.set(AlarmManager.
RTC_WAKEUP, motivCal.getTimeInMillis(),
motivPIntent);
  }
}else{
  alarmManager.cancel(motivPIntent);
  motivPIntent.cancel();
}
stopSelf();
```

```
  }
}
```

AlarmSetterReceiver.java

```
package com.specialproblem.brendz.
    timemanagement.notification;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;

/**
 * Receiver for Alarm Setter
 */
public class AlarmSetterReceiver extends
    BroadcastReceiver {
  @Override
  public void onReceive(Context context, Intent
      intent) {
    Intent alarmSetter = new Intent(context,
    AlarmSetter.class);
    context.startService(alarmSetter);
  }
}
```

MotivationReceiver.java

```
package com.specialproblem.brendz.
    timemanagement.notification;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;

/**
 * Receiver for Motivation Service
 */
public class MotivationReceiver extends
    BroadcastReceiver {
  @Override
  public void onReceive(Context context, Intent
      intent) {
    Bundle extras = intent.getExtras();
    Intent motivService = new Intent(context,
    MotivationService.class);
    if(extras!=null){
      int category = extras.getInt("CATEGORY");
      motivService.putExtra("CATEGORY",
    category);
      if(category==1) {
        motivService.putExtra("CATEGORY_ID",
    extras.getInt("CATEGORY_ID"));
      }
    }
    context.startService(motivService);
  }
}
```

MotivationService.java

```
package com.specialproblem.brendz.
    timemanagement.notification;

import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.app.Service;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;

import android.os.Bundle;
import android.os.IBinder;
import android.support.v4.app.
    NotificationCompat;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .task.ObjectMotivation;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskDashboard;

import java.util.List;
import java.util.Random;

/**
 * Displays notification for motivation
 */
public class MotivationService extends Service
    {

  private TableController tblController;
  @Override
  public IBinder onBind(Intent intent) {
    return null;
  }

  @Override
  public int onStartCommand(Intent intent, int
    flags, int startId) {
    Bundle extras = intent.getExtras();
    int category = 0;
    int category_id = 0;
    if(extras!=null){
      category = extras.getInt("CATEGORY");
      if(category==1){
        category_id = extras.getInt("
    CATEGORY_ID");
      }
    }
    tblController = TimeManagement.
    tblController;
    Uri sound = RingtoneManager.getDefaultUri(
    RingtoneManager.TYPE_NOTIFICATION);
    String title = getString(R.string.app_name)
    ;
    Intent intentI = new Intent(this,
    TaskDashboard.class);
    int requestID_I = (int) System.
    currentTimeMillis();
    List<ObjectMotivation> objectMotivationList
     = null;
    if(category==1){
      objectMotivationList = tblController.
    getMotivationsCategory(category_id);
    }else{
      objectMotivationList = tblController.
    getAllMotivations();
    }
    int count = objectMotivationList.size();
    Random rand = new Random();
    int index = rand.nextInt(count);
    String motivationText =
    objectMotivationList.get(index).
    getMotivation_text();

    PendingIntent pIntent = PendingIntent.
    getActivity(this, requestID_I, intentI,
    PendingIntent.FLAG_UPDATE_CURRENT);
    Notification notification = null;
    NotificationCompat.Builder notifBuilder =
    new NotificationCompat.Builder(this)
        .setSmallIcon(R.drawable.ic_add_btn)
        .setContentTitle(title)
        .setContentText(motivationText)
        .setTicker("Motivational message: ")
        .setContentIntent(pIntent)
        .setSound(sound)
        .setVibrate(new long[] { 1000, 1000,
    1000, 1000, 1000 });

    notification = notifBuilder.build();
    // hide the notification after its selected
    notification.flags |= Notification.
    FLAG_AUTO_CANCEL;

    NotificationManager notificationManager = (
    NotificationManager) getSystemService(
    NOTIFICATION_SERVICE);
    // cancel previous notification to clean up
     garbage in the status bar
    notificationManager.cancel(0);
    // add new notification
    notificationManager.notify(0, notification)
    ;
    return 0;
  }
}
```

ServiceStartOnBootReceiver.java

```
package com.specialproblem.brendz.
    timemanagement.notification;

import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;

import com.specialproblem.brendz.timemanagement
    .TimeManagement;

import java.util.Calendar;

/**
 * Restarts service on device boot
 */
public class ServiceStartOnBootReceiver extends
     BroadcastReceiver{

  private AlarmManager alarmManager;

  @Override
  public void onReceive(Context context, Intent
     intent) {
    //Restart setting of notifs when rebooted
    intent = new Intent(context,
    AlarmSetterReceiver.class);
    alarmManager = TimeManagement.alarmManager;
    PendingIntent pendingIntent = PendingIntent
    .getBroadcast(context, 1, intent,
    PendingIntent.FLAG_UPDATE_CURRENT);
    Calendar calendar = Calendar.getInstance();
    calendar.setTimeInMillis(System.
    currentTimeMillis());
    calendar.set(Calendar.SECOND, 0);
    calendar.set(Calendar.MINUTE, 0);
    calendar.set(Calendar.HOUR, 0);
    calendar.set(Calendar.AM_PM, Calendar.AM);
    alarmManager.setInexactRepeating(
    AlarmManager.RTC_WAKEUP, calendar.
    getTimeInMillis(), AlarmManager.
    INTERVAL_DAY, pendingIntent);
  }
}
```

ObjectCategory.java

```
package com.specialproblem.brendz.
    timemanagement.task;

/*
 * Category
 */
public class ObjectCategory {
  private int id;
  private String category_desc;

  public ObjectCategory(){}

  public int getId() {
    return id;
  }

  public void setId(int id) {
    this.id = id;
  }

  public String getCategory_desc() {
    return category_desc;
  }

  public void setCategory_desc(String
    category_desc) {
    this.category_desc = category_desc;
  }
}
```

ObjectMotivation.java

```
package com.specialproblem.brendz.
    timemanagement.task;

/**
 * Motivation
 */
public class ObjectMotivation {

  private int id;
  private String motivation_text;
  private int category_id;
  private String category_desc;

  public ObjectMotivation(){}

  public int getId() {
    return id;
  }

  public void setId(int id) {
    this.id = id;
  }

  public String getMotivation_text() {
    return motivation_text;
  }

  public void setMotivation_text(String
    motivation_text) {
    this.motivation_text = motivation_text;
  }

  public int getCategory_id() {
    return category_id;
  }

  public void setCategory_id(int category_id) {
    this.category_id = category_id;
  }

  public String getCategory_desc() {
    return category_desc;
  }

  public void setCategory_desc(String
    category_desc) {
    this.category_desc = category_desc;
  }
}
```

ObjectTask.java

```
package com.specialproblem.brendz.
    timemanagement.task;

/*
 * Task
 */
public class ObjectTask {

  private int id;
  private String task_name;
  private String task_description;
  private String task_category;
  private int task_priority;
  private String task_repeat;

  private int toCal;

  private int alarmSet;

  private int motivSet;

  private long repeat_start;
  private long repeat_end;
  private long repeat_timeStart;
  private long repeat_timeEnd;

  //For display purposes
  private String dateStart;
  private String dateEnd;
private String timeStart;
private String timeEnd;

private String task_repeat_month;
private String task_repeat_day;
private String task_repeat_year;
private String task_repeat_week;
private String task_repeat_weekday;

public ObjectTask(){}

public int getId() {
  return id;
}

public void setId(int id) {
  this.id = id;
}

public String getTask_name() {
  return task_name;
}

public void setTask_name(String task_name) {
  this.task_name = task_name;
}

public String getTask_description() {
  return task_description;
}

public void setTask_description(String
  task_description) {
  this.task_description = task_description;
}

public String getTask_repeat_month() {
  return task_repeat_month;
}

public void setTask_repeat_month(String
  task_repeat_month) {
  this.task_repeat_month = task_repeat_month;
}

public String getTask_repeat_day() {
  return task_repeat_day;
}

public void setTask_repeat_day(String
  task_repeat_day) {
  this.task_repeat_day = task_repeat_day;
}

public String getTask_repeat_year() {
  return task_repeat_year;
}

public void setTask_repeat_year(String
  task_repeat_year) {
  this.task_repeat_year = task_repeat_year;
}

public String getTask_category() {
  return task_category;
}

public void setTask_category(String
  task_category) {
  this.task_category = task_category;
}

public int getTask_priority() {
  return task_priority;
}

public void setTask_priority(int
  task_priority) {
  this.task_priority = task_priority;
}

public String getTask_repeat_week() {
  return task_repeat_week;
}

public void setTask_repeat_week(String
  task_repeat_week) {
  this.task_repeat_week = task_repeat_week;
}

public String getTask_repeat_weekday() {
  return task_repeat_weekday;
}

public void setTask_repeat_weekday(String
  task_repeat_weekday) {
  this.task_repeat_weekday =
  task_repeat_weekday;
}

public long getRepeat_start() {
  return repeat_start;
}

public void setRepeat_start(long repeat_start
  ) {
  this.repeat_start = repeat_start;
}

public long getRepeat_end() {
  return repeat_end;
}

public void setRepeat_end(long repeat_end) {
  this.repeat_end = repeat_end;
}

public long getRepeat_timeStart() {
  return repeat_timeStart;
}

public void setRepeat_timeStart(long
  repeat_timeStart) {
  this.repeat_timeStart = repeat_timeStart;
}

public long getRepeat_timeEnd() {
  return repeat_timeEnd;
}

public void setRepeat_timeEnd(long
  repeat_timeEnd) {
  this.repeat_timeEnd = repeat_timeEnd;
}

public String getDateStart() {
  return dateStart;
}

public void setDateStart(String dateStart) {
  this.dateStart = dateStart;
}

public String getDateEnd() {
  return dateEnd;
}

public void setDateEnd(String dateEnd) {
  this.dateEnd = dateEnd;
}

public String getTimeStart() {
  return timeStart;
}

public void setTimeStart(String timeStart) {
  this.timeStart = timeStart;
}

public String getTimeEnd() {
  return timeEnd;
}

public void setTimeEnd(String timeEnd) {
  this.timeEnd = timeEnd;
}

public String getTask_repeat() {
  return task_repeat;
}

public void setTask_repeat(String task_repeat
  ) {
  this.task_repeat = task_repeat;
}
```

```
  public int getAlarmSet() {
    return alarmSet;
  }

  public void setAlarmSet(int alarmSet) {
    this.alarmSet = alarmSet;
  }

  public int getToCal() {
    return toCal;
  }

  public void setToCal(int toCal) {
    this.toCal = toCal;
  }

  public int getMotivSet() {
    return motivSet;
  }

  public void setMotivSet(int motivSet) {
    this.motivSet = motivSet;
  }

}
```

ObjectTaskMeta.java

```
package com.specialproblem.brendz.
    timemanagement.task;

/**
 * Task Meta
 */
public class ObjectTaskMeta {

  private int taskID;
  private int alarmSet;
  private int motivSet;
  private int requestID;
  private int motivRequestID;
  private Long date;
  private Long est_start;
  private Long act_start;
  private Long est_end;
  private Long act_end;
  private Long time_diff_start;
  private Long time_diff_end;

  //For display purposes
  private String strdate;
  private String str_est_start;
  private String str_act_start;
  private String str_est_end;
  private String str_act_end;
  private String str_time_diff_start;
  private String str_time_diff_end;

  public ObjectTaskMeta(){}

  public String getStrdate() {
    return strdate;
  }

  public void setStrdate(String strdate) {
    this.strdate = strdate;
  }

  public String getStr_est_start() {
    return str_est_start;
  }

  public void setStr_est_start(String
    str_est_start) {
    this.str_est_start = str_est_start;
  }

  public String getStr_act_start() {
    return str_act_start;
  }

  public void setStr_act_start(String
    str_act_start) {
    this.str_act_start = str_act_start;
  }

  public String getStr_est_end() {
    return str_est_end;
  }

  public void setStr_est_end(String str_est_end
    ) {
    this.str_est_end = str_est_end;
  }

  public String getStr_act_end() {
    return str_act_end;
  }

  public void setStr_act_end(String str_act_end
    ) {
    this.str_act_end = str_act_end;
  }

  public String getStr_time_diff_start() {
    return str_time_diff_start;
  }

  public void setStr_time_diff_start(String
    str_time_diff_start) {
    this.str_time_diff_start =
    str_time_diff_start;
  }

  public String getStr_time_diff_end() {
    return str_time_diff_end;
  }

  public void setStr_time_diff_end(String
    str_time_diff_end) {
    this.str_time_diff_end = str_time_diff_end;
  }

  public int getTaskID() {
    return taskID;
  }

  public void setTaskID(int taskID) {
    this.taskID = taskID;
  }

  public Long getDate() {
    return date;
  }

  public void setDate(Long date) {
    this.date = date;
  }

  public Long getEst_start() {
    return est_start;
  }

  public void setEst_start(Long est_start) {
    this.est_start = est_start;
  }

  public Long getAct_start() {
    return act_start;
  }

  public void setAct_start(Long act_start) {
    this.act_start = act_start;
  }

  public Long getEst_end() {
    return est_end;
  }

  public void setEst_end(Long est_end) {
    this.est_end = est_end;
  }

  public Long getAct_end() {
    return act_end;
  }

  public void setAct_end(Long act_end) {
    this.act_end = act_end;
  }

  public Long getTime_diff_start() {
    return time_diff_start;
  }

  public void setTime_diff_start(Long
    time_diff_start) {
    this.time_diff_start = time_diff_start;
  }

  public Long getTime_diff_end() {
    return time_diff_end;
  }

  public void setTime_diff_end(Long
    time_diff_end) {
    this.time_diff_end = time_diff_end;
  }

  public int getAlarmSet() {
    return alarmSet;
  }

  public void setAlarmSet(int alarmSet) {
    this.alarmSet = alarmSet;
  }

  public int getRequestID() {
    return requestID;
  }

  public void setRequestID(int requestID) {
    this.requestID = requestID;
  }

  public int getMotivSet() {
    return motivSet;
  }

  public void setMotivSet(int motivSet) {
    this.motivSet = motivSet;
  }

  public int getMotivRequestID() {
    return motivRequestID;
  }

  public void setMotivRequestID(int
    motivRequestID) {
    this.motivRequestID = motivRequestID;
  }

}
```

OnClickListenerTask.java

```
package com.specialproblem.brendz.
    timemanagement.task;

import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Toast;

import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .view.calendar.CalendarMain;
import com.specialproblem.brendz.timemanagement
    .view.tasks.CategoryInput;
import com.specialproblem.brendz.timemanagement
    .view.tasks.CategoryStat;
import com.specialproblem.brendz.timemanagement
    .view.motivation.MotivationMain;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskInput;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskMain;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskView;

import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;

/*
 * OnClickListener Class for several OnClick
     events
 */
public class OnClickListenerTask implements
    View.OnClickListener {

  private String clickType;
  private Context context;
  private String id;
  private TableController tblController;
  private int month;
  private int day;
  private int year;

  public OnClickListenerTask(String clickType){
    this.clickType = clickType;
    tblController = TimeManagement.
    tblController;
  }

  public OnClickListenerTask(String clickType,
    int month, int day, int year){
    this.clickType = clickType;
    tblController = TimeManagement.
    tblController;
    this.month = month;
    this.day = day;
    this.year = year;
  }

  @Override
  public void onClick(View view){
    context = view.getContext();
    switch(clickType){
      case "createTask":
        //OnClickListener for Create Task
    button
        Intent createTask = new Intent(context,
     TaskInput.class);
        createTask.putExtra("INPUT_TYPE", 1);
        context.startActivity(createTask);
        break;
      case "createCategory":
        //OnClickListener for Create Category
    button
        LayoutInflater inflater = (
    LayoutInflater) context.getSystemService(
    Context.LAYOUT_INFLATER_SERVICE);
        final View formElementsView = inflater.
    inflate(R.layout.
    display_category_input_form, null, false);
        final EditText editTextCategoryDesc = (
    EditText) formElementsView.findViewById(R.
    id.editTextCategoryName);
        new AlertDialog.Builder(context).
    setView(formElementsView).setTitle("Add
    Category")
            .setPositiveButton("Add", new
    DialogInterface.OnClickListener() {
              @Override
              public void onClick(
    DialogInterface dialog, int id) {
                String categoryName =
    editTextCategoryDesc.getText().toString();
                boolean createSuccessful =
    tblController.insertCategory(categoryName);
                if (createSuccessful) {
                  Toast.makeText(context, "
    Category was saved.", Toast.LENGTH_SHORT).
    show();
                  ((CategoryInput) context).
    onResume();
                } else {
                  Toast.makeText(context, "
    Unable to save category.", Toast.
    LENGTH_SHORT).show();
                  ((CategoryInput) context).
    onResume();
                }
              }
            }).show();
        break;
      case "taskItem":
        id = view.getTag().toString();
        Intent viewTask = new Intent(context,
    TaskView.class);
    viewTask.putExtra("TASK_ID",Integer.
parseInt(id));
    viewTask.putExtra("CURR_MONTH",month);
    viewTask.putExtra("CURR_DAY",day);
    viewTask.putExtra("CURR_YEAR",year);
    context.startActivity(viewTask);
    break;
  case "startActual":
    id = view.getTag().toString();
    boolean insertActual = tblController.
startTaskMeta(Integer.parseInt(id), month,
day, year);
    if(insertActual) Toast.makeText(context
, "Task started." , Toast.LENGTH_SHORT).
show();
    break;
  case "endActual":
    id = view.getTag().toString();
    boolean isStarted = tblController.
isStartedTask(Integer.parseInt(id),month,
day,year);
    if(isStarted){
      boolean endActual = tblController.
endTaskMeta(Integer.parseInt(id), month,
day, year);
      if(endActual) Toast.makeText(context,
 "Task ended.", Toast.LENGTH_SHORT).show();
    }else{
      Toast.makeText(context, "Task not yet
 started.", Toast.LENGTH_SHORT).show();
    }
    break;
  case "calendar":
    Intent calendar = new Intent(context,
CalendarMain.class);
    context.startActivity(calendar);
    break;
  case "currentDay":
    Calendar cal = Calendar.getInstance();
    Intent taskMain = new Intent(context,
TaskMain.class);
    taskMain.putExtra("CURR_MONTH", cal.get
(Calendar.MONTH)+1);
    taskMain.putExtra("CURR_DAY", cal.get(
Calendar.DAY_OF_MONTH));
    taskMain.putExtra("CURR_YEAR", cal.get(
Calendar.YEAR));
    context.startActivity(taskMain);
    break;
  case "categoryStat":
    Intent categoryStat = new Intent(
context, CategoryStat.class);
    context.startActivity(categoryStat);
    break;
  case "motivMain":
    Intent motivMain = new Intent(context,
MotivationMain.class);
    context.startActivity(motivMain);
    break;
  case "motivInput":
    LayoutInflater inflaterMotiv = (
LayoutInflater) context.getSystemService(
Context.LAYOUT_INFLATER_SERVICE);
    final View formMotivationView =
inflaterMotiv.inflate(R.layout.
display_motivation_input_form, null, false)
;
    final EditText editTextMotivationName =
 (EditText) formMotivationView.findViewById
(R.id.editTextMotivation);
    final Spinner editCategorySpinner = (
Spinner) formMotivationView.findViewById(R.
id.spinnerCategory);

    //Set categories
    List<ObjectCategory> categories =
tblController.getAllCategories();
    List<String> category_descs = new
ArrayList<String>();
    for(ObjectCategory obj : categories){
      category_descs.add(obj.
getCategory_desc());
    }
    ArrayAdapter categoryAdapter = new
ArrayAdapter<String>(context, android.R.
layout.simple_spinner_dropdown_item,
category_descs);
    categoryAdapter.setDropDownViewResource
(android.R.layout.
simple_spinner_dropdown_item);
    editCategorySpinner.setAdapter(
categoryAdapter);
    new android.support.v7.app.AlertDialog.
Builder(context).setView(formMotivationView
).setTitle("Add Motivation")
        .setPositiveButton("Save Changes",
new DialogInterface.OnClickListener() {
          public void onClick(
DialogInterface dialog, int id) {
            ObjectMotivation
objectMotivation = new ObjectMotivation();
            objectMotivation.
setMotivation_text(editTextMotivationName.
getText().toString());
            objectMotivation.
setCategory_desc(editCategorySpinner.
getSelectedItem().toString());
            boolean insertSuccessful =
tblController.insertMotivation(
objectMotivation);
            if (insertSuccessful) {
              Toast.makeText(context, "
Motivation text was inserted.", Toast.
LENGTH_SHORT).show();
              ((MotivationMain) context).
onResume();
            }else{
              Toast.makeText(context, "
Unable to insert motivation record.", Toast
.LENGTH_SHORT).show();
              ((MotivationMain) context).
onResume();
            }
          }
        }).show();
    break;
  default: break;
  }
 }
}
```

CalendarMain.java

```
package com.specialproblem.brendz.
    timemanagement.view.calendar;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TableLayout;

import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .R;

/*
 * Class for displaying calendar
 */
public class CalendarMain extends
    AppCompatActivity {

  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    //sets the main layout of the activity
    setContentView(R.layout.
    activity_calendar_main);
    initToolbar();
    MonthView mv = new MonthView(this);
    TableLayout v = (TableLayout) findViewById(
    R.id.calendar);
    v.addView(mv);
  }

  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_task_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem
     item) {
    ToolBarListener action = new
    ToolBarListener();
    return action.actionbar_action(item, this);
  }
}
```

MonthView.java

```
package com.specialproblem.brendz.
    timemanagement.view.calendar;

import java.util.Calendar;
import java.util.Date;

import android.content.Context;
import android.content.Intent;
import android.content.res.Resources;
import android.graphics.Color;
import android.text.Html;
import android.util.AttributeSet;
import android.util.TypedValue;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.animation.
    TranslateAnimation;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .view.tasks.TaskMain;

/*
 *
 */
public class MonthView extends TableLayout{

  public static final int[] resDaysSun = {R.
    string.sunday,R.string.monday,R.string.
    tuesday,R.string.wednesday,
      R.string.thursday,R.string.friday,R.
    string.saturday};
  // private int[] resDaysMon = {R.string.
    monday,R.string.tuesday,R.string.wednesday,
  //          R.string.thursday,R.string.friday,
    R.string.saturday,R.string.sunday};

  public static final int[] monthIds = {R.
    string.january,R.string.february,R.string.
    march,R.string.april,R.string.may,R.string.
    june,
      R.string.july,R.string.august,R.string.
    september,R.string.october,R.string.
    november,R.string.december};

  int day=0,month=0,year=0;
  public int firstDay=Calendar.SUNDAY;
  private TextView btn;
  private TranslateAnimation animSet1,animSet2;
  private Context context;
  private TableRow tr;
  private Boolean[] isEvent = new Boolean[32];
  private String[] days;
  private String[] months = new String[12];

  Calendar cal,prevCal,today; //today will be
    used for setting a box around today's date
  //prevCal will be used to display last few
    dates of previous month in the calendar
  public MonthView(Context context,
    AttributeSet attrs) {
    super(context, attrs);
    init(context);
  }
  public MonthView(Context context){
    super(context);
    init(context);
  }

  private void init(Context context_arg)
  {
    context = context_arg; //initializing the
    context variable
    Resources res = getResources();
    for(int i=0;i<12;i++) {
      months[i] = res.getString(monthIds[i]);
    }
    days = new String[7];
    setStretchAllColumns(true); //stretch all
    columns so that calendar's width fits the
    View
    today = Calendar.getInstance();//get
    current date and time's instance

    today.clear(Calendar.HOUR);//remove the
    hour,minute,second and millisecond from the
     today variable
    today.clear(Calendar.MINUTE);
    today.clear(Calendar.SECOND);
    today.clear(Calendar.MILLISECOND);

    //if(firstDay==Calendar.MONDAY) {
      // today.setFirstDayOfWeek(Calendar.
    MONDAY);
    //}
    cal = (Calendar) today.clone(); //create
    exact copy as today for dates display
    purpose.

    DisplayMonth(true);//uses cal and prevCal
    to display the month
  }
  public void GoToDate(Date date)
  {
    cal.setTime(date);
    DisplayMonth(true);
  }

  private boolean animFlag=false;
  //Change month listener called when the user
    clicks to show next or prev month.
  private OnClickListener ChangeMonthListener =
     new OnClickListener(){
    @Override
    public void onClick(View v) {
      ImageView tv = (ImageView) v;
      //If previous month is to be displayed
    subtract one from current month.
      if(tv.getTag().equals("<"))
      {
        cal.add(Calendar.MONTH, -1);
        animFlag = false;
      }
      //If next month is to be displayed add
    one to the current month
      else
      {
        cal.add(Calendar.MONTH, 1);
        animFlag = true;
      }
      selected_day = 0;
      DisplayMonth(true);
    }};

  //Main function for displaying the current
    selected month
  int selected_day = 0;
  void DisplayMonth(boolean animationEnabled) {

    if(animationEnabled)
    {
      animSet1 = new TranslateAnimation(0,
    getWidth(),1,1);
  animSet1.setDuration(300);

  animSet2 = new TranslateAnimation(0,-
getWidth(),1,1);
  animSet2.setDuration(300);
}
Resources r = getResources();
String tempDay;
for(int i=0;i<7;i++)
{
  // if(firstDay == Calendar.MONDAY) {
 //       tempDay = r.getString(resDaysMon[i
]);
 //  }
 //  else {
    tempDay = r.getString(resDaysSun[i]);
  // }
  days[i] = tempDay.substring(0,3);
}

removeAllViews();//Clears the calendar so
that a new month can be displayed, removes
all child elements (days,week numbers, day
labels)

int firstDayOfWeek,prevMonthDay,
nextMonthDay,week;
cal.set(Calendar.DAY_OF_MONTH, 1); //Set
date = 1st of current month so that we can
know in next step which day is the first
day of the week.
firstDayOfWeek = cal.get(Calendar.
DAY_OF_WEEK)-1; //get which day is on the
first date of the month
if(firstDay==Calendar.MONDAY)
{
  firstDayOfWeek--;
  if(firstDayOfWeek==-1)
    firstDayOfWeek=6;
}
week = cal.get(Calendar.WEEK_OF_YEAR)-1; //
get which week is the current week.
if(firstDayOfWeek==0 && cal.get(Calendar.
MONTH)==Calendar.JANUARY) //adjustment for
week number when january starts with first
day of month as sunday
  week = 1;
if(week==0)
  week = 52;

prevCal = (Calendar) cal.clone(); //create
a calendar item for the previous month by
subtracting
prevCal.add(Calendar.MONTH, -1); //1 from
the current month

//get the number of days in the previous
month to display last few days of previous
month
prevMonthDay = prevCal.getActualMaximum(
Calendar.DAY_OF_MONTH)-firstDayOfWeek+1;
nextMonthDay = 1; //set the next month
counter to date 1
android.widget.TableRow.LayoutParams lp;

RelativeLayout rl = (RelativeLayout)
LayoutInflater.from(context).inflate(R.
layout.content_monthview, null);

//create the left arrow button for
displaying the previous month
ImageView btn1 = (ImageView) rl.
findViewById(R.id.imgLeft);
btn1.setTag("<");
btn1.setOnClickListener(ChangeMonthListener
);

btn = (TextView) rl.findViewById(R.id.
txtDay);
btn.setText(months[cal.get(Calendar.MONTH)
]);

((TextView)rl.findViewById(R.id.txtYear)).
setText(""+cal.get(Calendar.YEAR));

//create the right arrow button for
```

```
displaying the next month
btn1 = (ImageView) rl.findViewById(R.id.
imgRight);
btn1.setTag(">");
btn1.setOnClickListener(ChangeMonthListener
);
//add the tablerow containing the next and
prev views to the calendar
addView(rl);

tr = new TableRow(context); //create a new
row to add to the tablelayout
tr.setWeightSum(0.7f);
lp = new TableRow.LayoutParams();
lp.weight = 0.1f;
//Create the day labels on top of the
calendar
for(int i=0;i<7;i++)
{
  btn = new TextView(context);
  btn.setBackgroundResource(R.drawable.
calheader);
  btn.setPadding(10, 3,10, 3);
  btn.setLayoutParams(lp);
  btn.setTextColor(Color.parseColor("#9
C9A9D"));
  btn.setText(days[i]);
  btn.setTextSize(TypedValue.
COMPLEX_UNIT_SP,13);
  btn.setGravity(Gravity.CENTER);
  tr.addView(btn); //add the day label to
the tablerow
}
if(animationEnabled)
{
  if(animFlag)
    tr.startAnimation(animSet2);
  else
    tr.startAnimation(animSet1);
}
addView(tr); //add the tablerow to the
tablelayout (first row of the calendar)

tr.setLayoutParams(new LayoutParams(
LayoutParams.FILL_PARENT, LayoutParams.
FILL_PARENT));
/*initialize the day counter to 1, it will be
    used to display the dates of the month*/
int day=1;
lp = new TableRow.LayoutParams();
lp.weight = 0.1f;
for(int i=0;i<6;i++)
{
  if(day>cal.getActualMaximum(Calendar.
DAY_OF_MONTH))
    break;
  tr = new TableRow(context);
  tr.setWeightSum(0.7f);
  //this loop is used to fill out the days
in the i-th row in the calendar
  for(int j=0;j<7;j++)
  {
    btn = new TextView(context);
    btn.setLayoutParams(lp);
    btn.setBackgroundResource(R.drawable.
rectgrad);
    btn.setGravity(Gravity.CENTER);
    btn.setTextSize(TypedValue.
COMPLEX_UNIT_SP, 20);
    btn.setTextColor(Color.GRAY);
    if(j<firstDayOfWeek && day==1) //checks
 if the first day of the week has arrived
or previous month's date should be printed
      btn.setText(Html.fromHtml(String.
valueOf("<b>"+prevMonthDay+++"</b>")));
    else if(day>cal.getActualMaximum(
Calendar.DAY_OF_MONTH)) //checks to see
whether to print next month's date
    {
      btn.setText(Html.fromHtml("<b>"+
nextMonthDay+++"</b>"));
    }
    else //day counter is in the current
month
```

```
      {
        cal.set(Calendar.DAY_OF_MONTH, day);
        btn.setTag(day); //tag to be used
  when closing the calendar view
        btn.setOnClickListener(
  dayClickedListener);
        if(cal.equals(today))//if the day is
  today then set different background and
  text color
        {
          tv = btn;
          btn.setBackgroundResource(R.
  drawable.current_day);
          btn.setTextColor(Color.BLACK);
        }
        else if(selected_day==day)
        {
          tv = btn;
          btn.setBackgroundResource(R.
  drawable.selectedgrad);
          btn.setTextColor(Color.BLACK);
        }
        else
          btn.setTextColor(Color.WHITE);

        //set the text of the day
        btn.setText(Html.fromHtml("<b>"+
  String.valueOf(day++)+"</b>"));
        if(j==0)
          btn.setTextColor(Color.parseColor("
  #D73C10"));
        else if(j==6)
          btn.setTextColor(Color.parseColor("
  #009EF7"));

        if((day==this.day+1)&&(this.month==
  cal.get(Calendar.MONTH)+1)&&(this.year==cal
  .get(Calendar.YEAR)))
          btn.setBackgroundColor(Color.GRAY);
      }
      btn.setPadding(8,8,8,8); //maintains
  proper distance between two adjacent days
      tr.addView(btn);
    }
    if(animationEnabled)
    {
      if(animFlag)
        tr.startAnimation(animSet2);
      else
        tr.startAnimation(animSet1);
    }
    //this adds a table row for six times for
   six different rows in the calendar
    addView(tr);
  }
}
private TextView tv;

//Called when a day is clicked.
private OnClickListener dayClickedListener =
  new OnClickListener(){
  @Override
  public void onClick(View v) {
    if(tv.getText().toString().trim().equals(
  String.valueOf(today.get(Calendar.DATE))))
    {
      tv.setBackgroundResource(R.drawable.
  selectedgrad);
    }
    day = Integer.parseInt(v.getTag().
  toString());
    selected_day = day;
    tv = (TextView)v;
    tv.setBackgroundResource(R.drawable.
  selectedgrad);
    DisplayMonth(false);
  /*save the day,month and year in the public
      int variables day,month and year
   so that they can be used when the calendar
      is closed */

    cal.set(Calendar.DAY_OF_MONTH, day);
    final Context context = v.getContext();
    Intent taskMain = new Intent(context,
      TaskMain.class);
        taskMain.putExtra("CURR_YEAR", cal.get(
      Calendar.YEAR));
        taskMain.putExtra("CURR_MONTH", cal.get(
      Calendar.MONTH)+1);
        taskMain.putExtra("CURR_DAY", day);
        context.startActivity(taskMain);
      }
    };
}
```

MotivationMain.java

```
package com.specialproblem.brendz.
    timemanagement.view.motivation;

import android.app.TimePickerDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.Spinner;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.TimePicker;
import android.widget.Toast;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .notification.AlarmSetter;
import com.specialproblem.brendz.timemanagement
    .task.ObjectCategory;
import com.specialproblem.brendz.timemanagement
    .task.ObjectMotivation;
import com.specialproblem.brendz.timemanagement
    .task.OnClickListenerTask;

import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;
import java.util.Locale;

/**
 * Activity displaying motivations
 */
public class MotivationMain extends
    AppCompatActivity {

  private TableController tblController;
  private Switch motivSwitch;
  private TimePickerDialog
    timeStartPickerDialog;
  private TextView textViewTimeSet;

  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.
    activity_motivation_input);
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
    tblController = TimeManagement.
    tblController;

    Button buttonCreateLocation = (Button)
    findViewById(R.id.buttonCreateMotivation);
    buttonCreateLocation.setOnClickListener(new
     OnClickListenerTask("motivInput"));
    motivSwitch = (Switch) findViewById(R.id.
    motivSwitch);
    textViewTimeSet = (TextView) findViewById(R
    .id.textViewTimeSet);
    initToolbar();
    onResume();
  }

  @Override
  public void onResume(){
    setInput();
    readRecords();
    super.onResume();
  }

  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_task_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem
     item) {
    ToolBarListener action = new
    ToolBarListener();
    return action.actionbar_action(item, this);
  }

  public void readRecords() {
    //Display records on the task screen
    LinearLayout linearLayoutRecords = (
    LinearLayout) findViewById(R.id.
    linearLayoutRecordsMotivation);
    linearLayoutRecords.removeAllViews();
    List<ObjectMotivation> motivations =
    tblController.getAllMotivations();
    if (motivations.size() > 0) {
      for (ObjectMotivation obj : motivations)
    {
        String textViewContents = obj.
    getMotivation_text() + " - " + obj.
    getCategory_desc();
        TextView textViewTaskItem= new TextView
    (this);
        textViewTaskItem.setPadding(0, 10, 0,
    10);
        textViewTaskItem.setText(
    textViewContents);
        textViewTaskItem.setTag(obj.getId());
        textViewTaskItem.setOnLongClickListener
    (new OnLongClickListenerMotivation());
        linearLayoutRecords.addView(
    textViewTaskItem);
      }
    }
    else {
      TextView locationItem = new TextView(this
    );
      locationItem.setPadding(8, 8, 8, 8);
      locationItem.setText(R.string.
    no_records_yet);
      linearLayoutRecords.addView(locationItem)
    ;
    }
  }

  private void setInput(){
    final SimpleDateFormat timeFormatter = new
    SimpleDateFormat(TimeManagement.
    DEFAULT_TIME_FORMAT, Locale.US);
    int[] res = tblController.getMotivationMeta
    ();
    Calendar cal = Calendar.getInstance();
    cal.setTimeInMillis(res[1]);

    textViewTimeSet.setText("Set time is: " +
    timeFormatter.format(cal.getTime()));
    Calendar newCalendar = Calendar.getInstance
    ();
    int hour = newCalendar.get(Calendar.
    HOUR_OF_DAY);
    int minute = newCalendar.get(Calendar.
    MINUTE);

    if(res[0]==1){
      motivSwitch.setChecked(true);
    } else {
      motivSwitch.setChecked(false);
    }

    timeStartPickerDialog = new
    TimePickerDialog(MotivationMain.this, new
    TimePickerDialog.OnTimeSetListener(){
      @Override
      public void onTimeSet(TimePicker view,
    int hourOfDay, int minute) {
        Calendar newTime = Calendar.getInstance
    ();
        newTime.set(0, 0, 0, hourOfDay, minute)
    ;
        String date = timeFormatter.format(
    newTime.getTime());
        try {
          newTime.setTime(timeFormatter.parse(
    date));
        } catch (ParseException e) {
          e.printStackTrace();
        }
        boolean timeSuccess = tblController.
    setMotivationMeta(1,newTime.getTimeInMillis
    ()/1000);
        if(timeSuccess){
          Toast.makeText(MotivationMain.this, "
    Time has been set.", Toast.LENGTH_SHORT).
    show();
          textViewTimeSet.setText("Set time is:
     " + date);
          setAlarm();
        }
      }
    }, hour, minute, false);
    timeStartPickerDialog.setTitle("Set
    motivation notification time");

    motivSwitch.setOnCheckedChangeListener(new
    CompoundButton.OnCheckedChangeListener() {
      @Override
      public void onCheckedChanged(
    CompoundButton buttonView, boolean
    isChecked) {
        if(isChecked){
          timeStartPickerDialog.show();
        }else{
          boolean timeSuccess = tblController.
    setMotivationMeta(0, 0);
          if(timeSuccess){
            Toast.makeText(MotivationMain.this,
     "Time has been unset.", Toast.LENGTH_SHORT
    ).show();
            textViewTimeSet.setText("Motivation
     notification is off.");
          }
        }
      }
    });
  }

  private void setAlarm(){
    Intent intent = new Intent(this,
    AlarmSetter.class);
    startService(intent);
  }

  private class OnLongClickListenerMotivation
    implements View.OnLongClickListener {
    @Override
    public boolean onLongClick(View view) {
      final View viewFin = view;
      Context context = view.getContext();
      final String id = view.getTag().toString
    ();
      final CharSequence[] items = { "Edit", "
    Delete" };
      new AlertDialog.Builder(context).setTitle
    ("Motivation Record")
          .setItems(items, new DialogInterface.
    OnClickListener() {
            public void onClick(DialogInterface
     dialog, int item) {
              if (item == 0) {
                editMotivationRecord(Integer.
    parseInt(id));
              } else if (item == 1) {
                if(Integer.parseInt(id) > 15){
                  boolean deleteSuccessful =
    tblController.deleteMotivation(Integer.
    parseInt(id));
                  deleteAction(deleteSuccessful
    );
                } else {
                  deleteAction(false);
                }
              }
              dialog.dismiss();
            }
          }).show();
      return false;
    }
  }

  private void deleteAction(boolean
    deleteSuccessful){
    if (deleteSuccessful){
      Toast.makeText(this, "Motivation text was
     deleted.", Toast.LENGTH_SHORT).show();
      onResume();
    }else{
      Toast.makeText(this, "Cannot delete
    default motivation text.", Toast.
    LENGTH_SHORT).show();
      onResume();
    }
  }

  public void editMotivationRecord(final int
    motivationID) {
    ObjectMotivation objectMotivation =
    tblController.getSingleMotivation(
    motivationID);
    LayoutInflater inflater = (LayoutInflater)
    getSystemService(Context.
    LAYOUT_INFLATER_SERVICE);
    final View formMotivationView = inflater.
    inflate(R.layout.
    display_motivation_input_form, null, false)
    ;
    final EditText editTextMotivationName = (
    EditText) formMotivationView.findViewById(R
    .id.editTextMotivation);
    final Spinner editCategorySpinner = (
    Spinner) formMotivationView.findViewById(R.
    id.spinnerCategory);
    editTextMotivationName.setText(
    objectMotivation.getMotivation_text());

    //Set categories
    List<ObjectCategory> categories =
    tblController.getAllCategories();
    List<String> category_descs = new ArrayList
    <String>();
    for(ObjectCategory obj : categories){
      category_descs.add(obj.getCategory_desc()
    );
    }
    ArrayAdapter categoryAdapter = new
    ArrayAdapter<String>(this, android.R.layout
    .simple_spinner_dropdown_item,
    category_descs);
    categoryAdapter.setDropDownViewResource(
    android.R.layout.
    simple_spinner_dropdown_item);
    editCategorySpinner.setAdapter(
    categoryAdapter);

    new AlertDialog.Builder(this).setView(
    formMotivationView).setTitle("Edit
    Motivation")
        .setPositiveButton("Save Changes", new
    DialogInterface.OnClickListener() {
          public void onClick(DialogInterface
    dialog, int id) {
            ObjectMotivation objectMotivation =
     new ObjectMotivation();
            objectMotivation.setId(motivationID
    );
            objectMotivation.setMotivation_text
    (editTextMotivationName.getText().toString
    ());
            objectMotivation.setCategory_desc(
    editCategorySpinner.getSelectedItem().
    toString());
            boolean updateSuccessful =
    tblController.updateMotivation(
    objectMotivation);
            if (updateSuccessful) {
              Toast.makeText(
    getApplicationContext(), "Motivation text
    was updated", Toast.LENGTH_SHORT).show();
              onResume();
            }else{
              Toast.makeText(
    getApplicationContext(), "Unable to update
    motivation record.", Toast.LENGTH_SHORT).
    show();
              onResume();
            }
          }
        }).show();
  }
}
```

CategoryInput.java

```
package com.specialproblem.brendz.
    timemanagement.view.tasks;

import android.content.Context;
import android.content.DialogInterface;
import android.os.Bundle;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .task.ObjectCategory;
import com.specialproblem.brendz.timemanagement
    .task.OnClickListenerTask;
import com.specialproblem.brendz.timemanagement
    .R;

import java.util.List;

/**
 * Activity displaying categories
 */
public class CategoryInput extends
    AppCompatActivity {

  private TableController tblController;
  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
```

```
  super.onCreate(savedInstanceState);
  setContentView(R.layout.
  activity_category_input);
  Toolbar toolbar = (Toolbar) findViewById(R.
  id.toolbar);
  setSupportActionBar(toolbar);
  tblController = TimeManagement.
  tblController;

  Button buttonCreateLocation = (Button)
  findViewById(R.id.buttonCreateCategory);
  buttonCreateLocation.setOnClickListener(new
   OnClickListenerTask("createCategory"));

  initToolbar();
}

@Override
public void onResume(){
  readRecords();
  super.onResume();
}

public void initToolbar(){
  Toolbar toolbar = (Toolbar) findViewById(R.
  id.toolbar);
  setSupportActionBar(toolbar);
}

@Override
public boolean onCreateOptionsMenu(Menu menu)
   {
  getMenuInflater().inflate(R.menu.
  menu_task_main, menu);
  return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem
   item) {
  ToolBarListener action = new
  ToolBarListener();
  return action.actionbar_action(item, this);
}

public void readRecords() {
  //Display records on the task screen
  LinearLayout linearLayoutRecords = (
  LinearLayout) findViewById(R.id.
  linearLayoutRecordsCategory);
  linearLayoutRecords.removeAllViews();
  List<ObjectCategory> categories =
  tblController.getAllCategories();
  if (categories.size() > 0) {
    for (ObjectCategory obj : categories) {
      String textViewContents = obj.
  getCategory_desc();
      TextView textViewTaskItem= new TextView
  (this);
      textViewTaskItem.setPadding(0, 10, 0,
  10);
      textViewTaskItem.setText(
  textViewContents);
      textViewTaskItem.setTag(obj.getId());
      textViewTaskItem.setOnLongClickListener
  (new OnLongClickListenerCategory());
      linearLayoutRecords.addView(
  textViewTaskItem);
    }
  }
  else {
    TextView locationItem = new TextView(this
  );
    locationItem.setPadding(8, 8, 8, 8);
    locationItem.setText(R.string.
  no_records_yet);
    linearLayoutRecords.addView(locationItem)
  ;
  }
}

private class OnLongClickListenerCategory
  implements View.OnLongClickListener {
  @Override
  public boolean onLongClick(View view) {
    final View viewFin = view;
    Context context = view.getContext();
    final String id = view.getTag().toString
  ();
    final CharSequence[] items = { "Edit", "
  Delete" };
    new AlertDialog.Builder(context).setTitle
  ("Category Record")
        .setItems(items, new DialogInterface.
  OnClickListener() {
          public void onClick(DialogInterface
   dialog, int item) {
            if (item == 0) {
              editCategoryRecord(Integer.
  parseInt(id));
            } else if (item == 1) {
              if(Integer.parseInt(id) > 4){
                boolean deleteSuccessful =
  tblController.deleteCategory(Integer.
  parseInt(id));
                deleteAction(deleteSuccessful
  );
              } else {
                deleteAction(false);
              }
            }
            dialog.dismiss();
          }
        }).show();
    return false;
  }
}

private void deleteAction(boolean
  deleteSuccessful){
  if (deleteSuccessful){
    Toast.makeText(this, "Category record was
   deleted.", Toast.LENGTH_SHORT).show();
  }else{
    Toast.makeText(this, "Cannot delete
  default category.", Toast.LENGTH_SHORT).
  show();
  }
}

public void editCategoryRecord (final int
  categoryID) {
  ObjectCategory objectCategory =
  tblController.getSingleCategory(categoryID)
  ;
  LayoutInflater inflater = (LayoutInflater)
  getSystemService(Context.
  LAYOUT_INFLATER_SERVICE);
  final View formCategoryView = inflater.
  inflate(R.layout.
  display_category_input_form, null, false);
  final EditText editTextCategoryName = (
  EditText) formCategoryView.findViewById(R.
  id.editTextCategoryName);
  editTextCategoryName.setText(objectCategory
  .getCategory_desc());
  new AlertDialog.Builder(this).setView(
  formCategoryView).setTitle("Edit Category")
      .setPositiveButton("Save Changes", new
  DialogInterface.OnClickListener() {
        public void onClick(DialogInterface
  dialog, int id) {
          ObjectCategory objectCategory = new
   ObjectCategory();
          objectCategory.setId(categoryID);
          objectCategory.setCategory_desc(
  editTextCategoryName.getText().toString());
          boolean updateSuccessful =
  tblController.updateCategory(objectCategory
  );
          if (updateSuccessful) {
            Toast.makeText(
  getApplicationContext(), "Category record
  was updated", Toast.LENGTH_SHORT).show();
            onResume();
          }else{
            Toast.makeText(
  getApplicationContext(), "Unable to update
```

```
    category record.", Toast.LENGTH_SHORT).show
    ();
              onResume();
            }
          }
        }).show();
  }
}
```

CategoryStat.java

```
package com.specialproblem.brendz.
    timemanagement.view.tasks;

import android.content.Intent;
import android.database.Cursor;
import android.graphics.Color;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.text.TextUtils;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.SimpleCursorAdapter;
import android.widget.Spinner;
import android.widget.TextView;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .task.ObjectCategory;
import com.specialproblem.brendz.timemanagement
    .view.calendar.MonthView;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;
import java.util.Locale;

/**
 * Activity displaying statistics related to
     category
 */
public class CategoryStat extends
    AppCompatActivity {

  private TableController tblController;

  private int currMonth;
  private int currDay;
  private int currYear;
  private String category;
  private int countStart;
  private int countEnd;

  private ListView listview;
  private TextView tasksUnder;
  private TextView timeDiffStart;
  private TextView timeDiffEnd;
  private TextView lblStart;
  private TextView lblEnd;

  private SimpleDateFormat sdf;

  private int time_diff_start;
  private int time_diff_end;

  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.
    activity_category_stat);
    initToolbar();
    listview = (ListView) findViewById(R.id.
    listView);
    tasksUnder = (TextView) findViewById(R.id.
    lblTasksUnder);
    timeDiffStart = (TextView) findViewById(R.
    id.txtStartTime);
    timeDiffEnd = (TextView) findViewById(R.id.
    txtEndTime);
    lblStart = (TextView) findViewById(R.id.
    lblStart);
    lblEnd = (TextView) findViewById(R.id.
    lblEnd);

    countStart = 0;
    countEnd = 0;
    sdf = new SimpleDateFormat("HH:mm", Locale.
    US);
    onReturnToActivity();
  }

  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
  }

  private void setFields(){
    tblController = TimeManagement.
    tblController;
    final Spinner categorySpinner = (Spinner)
    findViewById(R.id.categorySpinner);
    List<ObjectCategory> categories =
    tblController.getAllCategories();
    List<String> category_descs = new ArrayList
    <String>();
    for(ObjectCategory obj : categories){
      category_descs.add(obj.getCategory_desc()
    );
    }
    ArrayAdapter categoryAdapter = new
    ArrayAdapter<String>(this, android.R.layout
    .simple_spinner_dropdown_item,
    category_descs);
    categoryAdapter.setDropDownViewResource(
    android.R.layout.
    simple_spinner_dropdown_item);
    categorySpinner.setAdapter(categoryAdapter)
    ;
    categorySpinner.setOnItemSelectedListener(
    new AdapterView.OnItemSelectedListener() {
      @Override
      public void onItemSelected(AdapterView<?>
     parent, View view, int position, long id)
    {
        category = categorySpinner.
    getSelectedItem().toString();
        readCategoryTasks(category);
      }
      @Override
      public void onNothingSelected(AdapterView
    <?> parent) {

      }
    });
  }

  private void onReturnToActivity(){
    setFields();
    readCategoryTasks("Work");
  }

  @Override
  public void onResume(){
    onReturnToActivity();
    super.onResume();
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_task_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem
    item) {
    ToolBarListener action = new
    ToolBarListener();
    return action.actionbar_action(item, this);
  }

  private void readCategoryTasks(String
    category){
    countStart = 0;
    countEnd = 0;
    time_diff_start = 0;
    time_diff_end = 0;
    tasksUnder.setText("Displaying tasks under:
     " + category);
    Cursor cursor = tblController.
    getTasksUnderCategories(category);
    SimpleCursorAdapter adapter = new
    SimpleCursorAdapter(
        this,
        R.layout.display_stat,
        cursor,
        new String[]{"task_name", "date", "date
    ", "date", "repeat_timeStart", "
    task_priority",
        "est_start", "act_start", "est_end", "
    act_end", "time_diff_start", "time_diff_end
    "},
        new int[]{R.id.msg_tv, R.id.year_tv, R.
    id.month_tv, R.id.date_tv, R.id.time_tv, R.
    id.priority_tv,
        R.id.est_start_tv, R.id.act_start_tv, R
    .id.est_end_tv, R.id.act_end_tv, R.id.
    time_diff_start_tv, R.id.time_diff_end_tv})
    ;
    adapter.setViewBinder(new
    SimpleCursorAdapter.ViewBinder() {
      @Override
      public boolean setViewValue(View view,
    Cursor cursor, int columnIndex) {
        if (view.getId() == R.id.msg_tv) return
     false;
        TextView tv = (TextView)view;
        RelativeLayout parent = (RelativeLayout
    ) tv.getParent();
        Calendar cal = Calendar.getInstance();
        Long time = 0L;
        int hour = 0;
        int minute = 0;
        String txtTime = null;
        String txtHour = null;
        String txtMinute = null;
        if(cursor.getString(columnIndex)!=null)
    {
          time = cursor.getLong(columnIndex)
    *1000;
          cal.setTimeInMillis(time);
          hour = cal.get(Calendar.HOUR);
          minute = cal.get(Calendar.MINUTE);
          txtTime = (cal.get(Calendar.
    HOUR_OF_DAY) < 12) ? " AM":" PM";
          txtHour = (hour < 9) ? "0"+Integer.
    toString(hour)+":":Integer.toString(hour)+"
    :";
          txtMinute = (minute < 9) ? "0"+
    Integer.toString(minute):Integer.toString(
    minute);
        } else {
          txtTime = "-";
          txtHour = "";
          txtMinute = "";
        }
        switch(view.getId()) {
          case R.id.est_start_tv:
            tv.setText(TextUtils.concat("Est
    start: ",txtHour,txtMinute,txtTime));
            break;
          case R.id.est_end_tv:
            tv.setText(TextUtils.concat("Est
    end: ",txtHour,txtMinute,txtTime));
            break;
          case R.id.act_start_tv:
            tv.setText(TextUtils.concat("Act
    start: ",txtHour,txtMinute,txtTime));
          case R.id.act_end_tv:
            tv.setText(TextUtils.concat("Act
    end: ",txtHour,txtMinute,txtTime));
            break;
          case R.id.time_diff_start_tv:
            if(cursor.getString(cursor.
getColumnIndex("time_diff_start"))!=null){
              int start = Integer.parseInt(
cursor.getString(cursor.getColumnIndex("
time_diff_start")));
              if(start > 0){
                cal.setTimeInMillis(start*1000)
;
                cal.add(Calendar.HOUR_OF_DAY,
-8);
                tv.setText("Difference Start:+"
 + sdf.format(cal.getTimeInMillis()));
              }else{
                cal.setTimeInMillis(start
*-1000);
                cal.add(Calendar.HOUR_OF_DAY,
-8);
                tv.setText("Difference Start:-"
 + sdf.format(cal.getTimeInMillis()));
              }
            }else{
              tv.setText("Difference Start
:+00:00");
            }
            break;
          case R.id.time_diff_end_tv:
            if(cursor.getString(cursor.
getColumnIndex("time_diff_end"))!=null) {
              int end = Integer.parseInt(cursor
.getString(cursor.getColumnIndex("
time_diff_end")));
              if (end > 0) {
                cal.setTimeInMillis(end * 1000)
;
                cal.add(Calendar.HOUR_OF_DAY,
-8);
                tv.setText("Difference End:+" +
 sdf.format(cal.getTimeInMillis()));
              } else {
                cal.setTimeInMillis(end *
-1000);
                cal.add(Calendar.HOUR_OF_DAY,
-8);
                tv.setText("Difference End:-" +
 sdf.format(cal.getTimeInMillis()));
              }
            } else {
              tv.setText("Difference End:-00:00
");
            }
            break;
          case R.id.priority_tv:
            tv.setText("Priority Level: " +
cursor.getString(cursor.getColumnIndex("
task_priority")));
            break;
          case R.id.year_tv:
            currMonth = cal.get(Calendar.MONTH)
+1;
            currDay = cal.get(Calendar.
DAY_OF_MONTH);
            currYear = cal.get(Calendar.YEAR);
            String[] infoTask = {cursor.
getString(cursor.getColumnIndex("_id")),
Integer.toString(currMonth),
              Integer.toString(currDay),
Integer.toString(currYear)};
            parent.setTag(infoTask);
            tv.setText(Integer.toString(cal.get
(Calendar.YEAR)));
            break;
          case R.id.month_tv:
            int monthID = cal.get(Calendar.
MONTH);
            tv.setText(getString(MonthView.
monthIds[monthID]));
            break;
          case R.id.date_tv:
            tv.setText(Integer.toString(cal.get
(Calendar.DAY_OF_MONTH)));
            break;
          case R.id.time_tv:
            long now = System.currentTimeMillis
();
        tv.setText(TextUtils.concat(txtHour
,txtMinute,txtTime));
        cal.set(currYear,currMonth-1,
currDay);
        time = cal.getTimeInMillis();
        TextView tv2 = (TextView) parent.
findViewById(R.id.msg_tv);
        if (time < now) tv2.setTextColor(
Color.parseColor("#B70000"));
        else tv2.setTextColor(Color.
parseColor("#587498"));
        break;
    }
    return true;
  }
});
listview.setAdapter(adapter);
listview.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
  @Override
  public void onItemClick(AdapterView<?>
parent, View view, int position, long id) {
    String[] infoTask = (String[]) view.
getTag();
    Intent viewTask = new Intent(view.
getContext(), TaskView.class);
    viewTask.putExtra("TASK_ID",Integer.
parseInt(infoTask[0]));
    viewTask.putExtra("CURR_MONTH",Integer.
parseInt(infoTask[1]));
    viewTask.putExtra("CURR_DAY",Integer.
parseInt(infoTask[2]));
    viewTask.putExtra("CURR_YEAR",Integer.
parseInt(infoTask[3]));
    view.getContext().startActivity(
viewTask);
  }
});
if(cursor.moveToFirst()){
  do {
    //Set average time
    if(cursor.getString(cursor.
getColumnIndex("time_diff_start"))!=null){
      time_diff_start += Integer.parseInt(
cursor.getString(cursor.getColumnIndex("
time_diff_start")));
      countStart++;
    }
    if(cursor.getString(cursor.
getColumnIndex("time_diff_end"))!=null){
      time_diff_end += Integer.parseInt(
cursor.getString(cursor.getColumnIndex("
time_diff_end")));
      countEnd++;
    }
  } while (cursor.moveToNext());
}
setAverageValues();
}

private void setAverageValues(){
  SimpleDateFormat sdf = new SimpleDateFormat
  ("HH:mm");
  Calendar cal = Calendar.getInstance();
  if(time_diff_start > 0){
    if(countStart > 0){
      cal.setTimeInMillis(time_diff_start
*1000/countStart);
    }else{
      cal.setTimeInMillis(time_diff_start
*1000);
    }
    cal.add(Calendar.HOUR_OF_DAY, -8);
    timeDiffStart.setText("+" + sdf.format(
cal.getTimeInMillis()));
    lblStart.setText("Advanced by: (Hrs:Mins)
");
  }else{
    if(countStart > 0){
      cal.setTimeInMillis(time_diff_start
*-1000/countStart);
    }else{
      cal.setTimeInMillis(time_diff_start
*-1000);
    }
    cal.add(Calendar.HOUR_OF_DAY, -8);
    timeDiffStart.setText("-" + sdf.format(
cal.getTimeInMillis()));
    lblStart.setText("Delayed by: (Hrs:Mins)"
);
  }
  if(time_diff_end > 0){
    if(countEnd > 0){
      cal.setTimeInMillis(time_diff_end*1000/
countEnd);
    }else{
      cal.setTimeInMillis(time_diff_end*1000)
;
    }
    cal.add(Calendar.HOUR_OF_DAY, -8);
    timeDiffEnd.setText("+" + sdf.format(cal.
getTimeInMillis()));
    lblEnd.setText("Advanced by: (Hrs:Mins)")
;
  }else{
    if(countEnd > 0){
      cal.setTimeInMillis(time_diff_start
*-1000/countEnd);
    }else{
      cal.setTimeInMillis(time_diff_start
*-1000);
    }
    cal.add(Calendar.HOUR_OF_DAY, -8);
    timeDiffEnd.setText("-" + sdf.format(cal.
getTimeInMillis()));
    lblEnd.setText("Delayed by: (Hrs:Mins)");
  }
 }
}
```

TaskDashboard.java

```
package com.specialproblem.brendz.
    timemanagement.view.tasks;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.graphics.Color;
import android.os.Bundle;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.text.TextUtils;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.SimpleCursorAdapter;
import android.widget.Spinner;
import android.widget.TextView;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .notification.AlarmSetter;
import com.specialproblem.brendz.timemanagement
    .task.OnClickListenerTask;
import com.specialproblem.brendz.timemanagement
    .view.calendar.MonthView;

import java.util.Calendar;

/**
 * Activity displaying task dashboard
 */
```

```
public class TaskDashboard extends
    AppCompatActivity {

  private ListView listview;
  private TableController tblController;
  private int currMonth = 0;
  private int currDay = 0;
  private int currYear = 0;

  @Override
  public void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.
    activity_task_dashboard);
    initToolbar();
    tblController = TimeManagement.
    tblController;
    onReturnToActivity();
  }

  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
  }

  private void setFields(){
    Button calendarShow = (Button) findViewById
    (R.id.calendarShow);
    Button currentDayShow = (Button)
    findViewById(R.id.currentDayShow);
    Button categoryStatShow = (Button)
    findViewById(R.id.categoryStatShow);
    Button motivationShow = (Button)
    findViewById(R.id.motivationShow);

    listview = (ListView) findViewById(R.id.
    listView);
    calendarShow.setOnClickListener(new
    OnClickListenerTask("calendar"));
    currentDayShow.setOnClickListener(new
    OnClickListenerTask("currentDay"));
    categoryStatShow.setOnClickListener(new
    OnClickListenerTask("categoryStat"));
    motivationShow.setOnClickListener(new
    OnClickListenerTask("motivMain"));

    Spinner dashboardSpinner = (Spinner)
    findViewById(R.id.dashboardSpinner);
    String[] array = getResources().
    getStringArray(R.array.dashboardArray);
    ArrayAdapter dashboardString = new
    ArrayAdapter<String>(this, android.R.layout
    .simple_spinner_dropdown_item, array);
    dashboardString.setDropDownViewResource(
    android.R.layout.
    simple_spinner_dropdown_item);
    dashboardSpinner.setAdapter(dashboardString
    );
    dashboardSpinner.setOnItemSelectedListener(
    new AdapterView.OnItemSelectedListener() {
      @Override
      public void onItemSelected(AdapterView<?>
     parent, View view, int position, long id)
    {
          switch(position){
            case 0: readTasks("new");
                break;
            case 1: readTasks("delayed");
                break;
            case 2: readTasks("highPrio");
                break;
            default: readTasks("new");
                 break;
          }
      }
      @Override
      public void onNothingSelected(AdapterView
    <?> parent) {

      }
    });
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_task_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem
     item) {
    ToolBarListener action = new
    ToolBarListener();
    return action.actionbar_action(item, this);
  }

  @Override
  public void onResume(){
    onReturnToActivity();
    super.onResume();
  }

  private void onReturnToActivity(){
    setFields();
    readTasks("new");
  }

  private void readTasks(String type){
    Cursor cursor = null;
    switch(type){
      case "new": cursor = tblController.
    readNewTasks("new");
            break;
      case "delayed": cursor = tblController.
    readDelayedTasks();
              break;
      case "highPrio": cursor = tblController.
    readNewTasks("highPrio");
              break;
    }
    SimpleCursorAdapter adapter = new
    SimpleCursorAdapter(
        this,
        R.layout.display_taskmain,
        cursor,
        new String[]{"task_name", "date", "date
    ", "date", "repeat_timeStart", "
    task_priority"},
        new int[]{R.id.msg_tv, R.id.year_tv, R.
    id.month_tv, R.id.date_tv, R.id.time_tv, R.
    id.priority_tv});
    adapter.setViewBinder(new
    SimpleCursorAdapter.ViewBinder() {
      @Override
      public boolean setViewValue(View view,
    Cursor cursor, int columnIndex) {
        if (view.getId() == R.id.msg_tv) return
     false;
        TextView tv = (TextView)view;
        RelativeLayout parent = (RelativeLayout
    ) tv.getParent();
        Calendar cal = Calendar.getInstance();
        Long time = cursor.getLong(columnIndex)
    *1000;
        cal.setTimeInMillis(time);
        switch(view.getId()) {
          case R.id.priority_tv:
            tv.setText("Priority Level: " +
    cursor.getString(cursor.getColumnIndex("
    task_priority")));
            break;
          case R.id.year_tv:
            currMonth = cal.get(Calendar.MONTH)
    +1;
            currDay = cal.get(Calendar.
    DAY_OF_MONTH);
            currYear = cal.get(Calendar.YEAR);
            String[] infoTask = {cursor.
    getString(cursor.getColumnIndex("_id")),
    Integer.toString(currMonth),
                Integer.toString(currDay),
    Integer.toString(currYear)};
            parent.setTag(infoTask);
            tv.setText(Integer.toString(cal.get
    (Calendar.YEAR)));
            break;
        case R.id.month_tv:
          int monthID = cal.get(Calendar.
MONTH);
          tv.setText(getString(MonthView.
monthIds[monthID]));
          break;
        case R.id.date_tv:
          tv.setText(Integer.toString(cal.get
(Calendar.DAY_OF_MONTH)));
          break;
        case R.id.time_tv:
          long now = System.currentTimeMillis
();
          int hour = cal.get(Calendar.HOUR);
          int minute = cal.get(Calendar.
MINUTE);
          String txtTime = (cal.get(Calendar.
HOUR_OF_DAY) < 12) ? " AM":" PM";
          String txtHour = (hour < 9) ? "0"+
Integer.toString(hour):Integer.toString(
hour);
          String txtMinute = (minute < 9) ? "
0"+Integer.toString(minute):Integer.
toString(minute);
          tv.setText(TextUtils.concat(txtHour
,":",txtMinute,txtTime));
          cal.set(currYear,currMonth-1,
currDay);
          time = cal.getTimeInMillis();
          TextView tv2 = (TextView) parent.
findViewById(R.id.msg_tv);
          if (time < now) tv2.setTextColor(
Color.parseColor("#B70000"));
          else tv2.setTextColor(Color.
parseColor("#587498"));
          break;
      }
      return true;
    }
});
listview.setAdapter(adapter);
listview.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
  @Override
  public void onItemClick(AdapterView<?>
parent, View view, int position, long id) {
    String[] infoTask = (String[]) view.
getTag();
    Intent viewTask = new Intent(view.
getContext(), TaskView.class);
    viewTask.putExtra("TASK_ID",Integer.
parseInt(infoTask[0]));
    viewTask.putExtra("CURR_MONTH",Integer.
parseInt(infoTask[1]));
    viewTask.putExtra("CURR_DAY",Integer.
parseInt(infoTask[2]));
    viewTask.putExtra("CURR_YEAR",Integer.
parseInt(infoTask[3]));
    view.getContext().startActivity(
viewTask);
  }
});
listview.setOnItemLongClickListener(new
AdapterView.OnItemLongClickListener() {
  @Override
  public boolean onItemLongClick(
AdapterView<?> parent, View view, int
position, long id) {
    final View viewFin = view;
    final Context context = view.getContext
();
    final String[] infoTask = (String[])
view.getTag();
    final CharSequence[] items = { "Task
Overview", "Edit", "Delete" };
    new AlertDialog.Builder(context).
setTitle("Task record")
          .setItems(items, new
DialogInterface.OnClickListener() {
            public void onClick(
DialogInterface dialog, int item) {
              if (item == 0){
                Intent tOverview = new Intent
(context, TaskOverview.class);
                tOverview.putExtra("TASK_ID",
  infoTask[0]);
                startActivity(tOverview);
              }else if (item == 1) {
                editTaskRecord(Integer.
  parseInt(infoTask[0]), viewFin);
              }
              else if (item == 2) {
                int requestId = tblController
  .getRequestID(Integer.parseInt(infoTask[0])
  ,Integer.parseInt(infoTask[1])-1,Integer.
  parseInt(infoTask[2]),Integer.parseInt(
  infoTask[3]));
                tblController.updateRequestID
  (Integer.parseInt(infoTask[0]),Integer.
  parseInt(infoTask[1])-1,Integer.parseInt(
  infoTask[2]),Integer.parseInt(infoTask[3]),
  requestId,0,1);
                setAlarm();
                onResume();
              }
              dialog.dismiss();
            }
          }).show();
      return true;
    }
  });
}

private void setAlarm(){
  Intent alarmSetter = new Intent(this,
  AlarmSetter.class);
  startService(alarmSetter);
}

private void editTaskRecord (final int taskId
  , final View viewFin) {
  final Context context = viewFin.getContext
  ();
  Intent editTask = new Intent(context,
  TaskInput.class);
  editTask.putExtra("INPUT_TYPE",2);
  editTask.putExtra("ID",taskId);
  context.startActivity(editTask);
}
}
```

TaskInput.java

```
package com.specialproblem.brendz.
    timemanagement.view.tasks;

import android.app.DatePickerDialog;
import android.app.TimePickerDialog;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.text.InputType;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Switch;
import android.widget.TextView;
import android.widget.TimePicker;
import android.widget.Toast;

import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .database_controller.InputProcessing;
import com.specialproblem.brendz.timemanagement
    .googlecalendar.AddToCalendar;
import com.specialproblem.brendz.timemanagement
    .notification.AlarmSetter;
```

```
import com.specialproblem.brendz.timemanagement
    .task.ObjectCategory;
import com.specialproblem.brendz.timemanagement
    .task.ObjectTask;
import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;
import java.util.Locale;

/*
 * Activity for input/edit of saved tasks
 */
public class TaskInput extends
    AppCompatActivity {

  private int taskId;
  private int inputType;

  private TableController tblController;

  private Spinner categoryDropdown,
    priorityDropdown, repeatDropdown;

  private DatePickerDialog
    dateStartPickerDialog;
  private DatePickerDialog dateEndPickerDialog;

  private TimePickerDialog
    timeStartPickerDialog;
  private TimePickerDialog timeEndPickerDialog;

  private ArrayAdapter<String> categoryAdapter;
  private ArrayAdapter<String> repeatAdapter;
  private ArrayAdapter<Integer> priorityAdapter
    ;

  private EditText datePickerFrom;
  private EditText datePickerTo;

  private EditText timePickerFrom;
  private EditText timePickerTo;

  private SimpleDateFormat dateFormatter;
  private SimpleDateFormat timeFormatter;

  private Switch alarmSwitch;
  private Switch motivSwitch;
  private CheckBox addToCalendar;

  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_task_input
    );
    initToolbar();

    tblController = TimeManagement.
    tblController;

    datePickerFrom = (EditText) findViewById(R.
    id.datePickerStart);
    datePickerFrom.setInputType(InputType.
    TYPE_NULL);

    datePickerTo = (EditText) findViewById(R.id
    .datePickerEnd);
    datePickerTo.setInputType(InputType.
    TYPE_NULL);

    timePickerFrom = (EditText) findViewById(R.
    id.timePickerStart);
    timePickerFrom.setInputType(InputType.
    TYPE_NULL);

    timePickerTo = (EditText) findViewById(R.id
    .timePickerEnd);
    timePickerTo.setInputType(InputType.
    TYPE_NULL);

    categoryDropdown = (Spinner) findViewById(R
    .id.spinnerCategory);
    priorityDropdown = (Spinner) findViewById(R
    .id.spinnerPriority);
    repeatDropdown = (Spinner) findViewById(R.
    id.spinnerRepeat);

    alarmSwitch = (Switch) findViewById(R.id.
    alarmSwitch);
    motivSwitch = (Switch) findViewById(R.id.
    motivSwitch);
    addToCalendar = (CheckBox) findViewById(R.
    id.toCalendarSwitch);
    setInputType();
  }

  // Toolbar Functions
  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_task_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem
     item) {
    ToolBarListener action = new
    ToolBarListener();
    return action.actionbar_action(item, this);
  }

  private void setInputType(){
    Bundle extras = getIntent().getExtras();
    Button inputButton = (Button) findViewById(
    R.id.inputButton);
    if (extras != null) {
      inputType = extras.getInt("INPUT_TYPE");
      loadSpinnerInputs();
      setDateTimeField();
      if(inputType==1){
        inputButton.setText("Add");
        alarmSwitch.setChecked(false);
        motivSwitch.setChecked(false);
      }else{
        inputButton.setText("Edit");
        taskId = extras.getInt("ID");
        ObjectTask objectTask = tblController.
    readSingleRecord(taskId);
        InputProcessing inputProcessing = new
    InputProcessing();
        objectTask = inputProcessing.
    editInputProcessing(objectTask);

        EditText editTaskName = (EditText)
    findViewById(R.id.editTaskName);
        EditText editTaskDesc = (EditText)
    findViewById(R.id.editTaskDesc);
        EditText editTimeStart = (EditText)
    findViewById(R.id.timePickerStart);
        EditText editTimeEnd = (EditText)
    findViewById(R.id.timePickerEnd);
        EditText editDateStart = (EditText)
    findViewById(R.id.datePickerStart);
        EditText editDateEnd = (EditText)
    findViewById(R.id.datePickerEnd);

        editTaskName.setText(objectTask.
    getTask_name());
        editTaskDesc.setText(objectTask.
    getTask_description());
        String category = objectTask.
    getTask_category();

        int spinnerPosition = categoryAdapter.
    getPosition(category);
        categoryDropdown.setSelection(
    spinnerPosition);
        int priority = objectTask.
    getTask_priority();
        spinnerPosition = priorityAdapter.
  getPosition(priority);
      priorityDropdown.setSelection(
  spinnerPosition);

      String timeStart = objectTask.
  getTimeStart();
      String timeEnd = objectTask.getTimeEnd
  ();
      editTimeStart.setText(timeStart);
      editTimeEnd.setText(timeEnd);

      String repeatType = objectTask.
  getTask_repeat();
      spinnerPosition = repeatAdapter.
  getPosition(repeatType);
      repeatDropdown.setSelection(
  spinnerPosition);

      int alarmSet = objectTask.getAlarmSet()
  ;
      int motivSet = objectTask.getMotivSet()
  ;

      if(alarmSet==1){
        alarmSwitch.setChecked(true);
      }else{
        alarmSwitch.setChecked(false);
      }

      if(motivSet==1){
        motivSwitch.setChecked(true);
      }else{
        motivSwitch.setChecked(false);
      }

      String dateStart = objectTask.
  getDateStart();
      editDateStart.setText(dateStart);
      if(!repeatType.equals("Once")){
        String dateEnd = objectTask.
  getDateEnd();
        editDateEnd.setText(dateEnd);
      }

    }
  }
  inputButton.setOnClickListener(new
  OnClickListenerAddTaskToDB());
}

private void loadSpinnerInputs() {
  List<ObjectCategory> categories =
  tblController.getAllCategories();
  List<String> category_descs = new ArrayList
  <String>();
  for(ObjectCategory obj : categories){
    category_descs.add(obj.getCategory_desc()
  );
  }
  categoryAdapter = new ArrayAdapter<String>(
  this, android.R.layout.
  simple_spinner_dropdown_item,
  category_descs);
  categoryAdapter.setDropDownViewResource(
  android.R.layout.
  simple_spinner_dropdown_item);
  categoryDropdown.setAdapter(categoryAdapter
  );

  String[] array = getResources().
  getStringArray(R.array.priorityArray);
  Integer[] intArray = new Integer[array.
  length];
  for(int i = 0; i < array.length; i++) {
    intArray[i] = Integer.parseInt(array[i]);
  }
  priorityAdapter = new ArrayAdapter<Integer
  >(this, android.R.layout.
  simple_spinner_dropdown_item, intArray);
  priorityAdapter.setDropDownViewResource(
  android.R.layout.
  simple_spinner_dropdown_item);
  priorityDropdown.setAdapter(priorityAdapter
  );
  array = getResources().getStringArray(R.
  array.repeatArray);
  repeatAdapter = new ArrayAdapter<String>(
  this, android.R.layout.
  simple_spinner_dropdown_item, array);
  repeatAdapter.setDropDownViewResource(
  android.R.layout.
  simple_spinner_dropdown_item);
  repeatDropdown.setAdapter(repeatAdapter);
  repeatDropdown.setOnItemSelectedListener(
  new OnItemSelectedListenerDisplay());
}

private void setDateTimeField() {
  dateFormatter = new SimpleDateFormat(
  TimeManagement.DEFAULT_DATE_FORMAT, Locale.
  US);
  timeFormatter = new SimpleDateFormat(
  TimeManagement.DEFAULT_TIME_FORMAT, Locale.
  US);

  if(inputType==1){
    datePickerFrom.setOnClickListener(new
  OnClickListenerDatePicker());
    datePickerTo.setOnClickListener(new
  OnClickListenerDatePicker());
    timePickerFrom.setOnClickListener(new
  OnClickListenerTimePicker());
    timePickerTo.setOnClickListener(new
  OnClickListenerTimePicker());
  }

  Calendar newCalendar = Calendar.getInstance
  ();
  dateStartPickerDialog = new
  DatePickerDialog(this, new DatePickerDialog
  .OnDateSetListener() {
    public void onDateSet(DatePicker view,
  int year, int monthOfYear, int dayOfMonth)
  {
      Calendar newDate = Calendar.getInstance
  ();
      newDate.set(year, monthOfYear,
  dayOfMonth);
      datePickerFrom.setText(dateFormatter.
  format(newDate.getTime()));
    }
  },newCalendar.get(Calendar.YEAR),
  newCalendar.get(Calendar.MONTH),
  newCalendar.get(Calendar.DAY_OF_MONTH));

  dateEndPickerDialog = new DatePickerDialog(
  this, new DatePickerDialog.
  OnDateSetListener() {

    public void onDateSet(DatePicker view,
  int year, int monthOfYear, int dayOfMonth)
  {
      Calendar newDate = Calendar.getInstance
  ();
      newDate.set(year, monthOfYear,
  dayOfMonth);
      datePickerTo.setText(dateFormatter.
  format(newDate.getTime()));
    }

  },newCalendar.get(Calendar.YEAR),
  newCalendar.get(Calendar.MONTH),
  newCalendar.get(Calendar.DAY_OF_MONTH));

  int hour = newCalendar.get(Calendar.
  HOUR_OF_DAY);
  int minute = newCalendar.get(Calendar.
  MINUTE);
  timeStartPickerDialog = new
  TimePickerDialog(this, new TimePickerDialog
  .OnTimeSetListener(){
    @Override
    public void onTimeSet(TimePicker view,
  int hourOfDay, int minute) {
      Calendar newTime = Calendar.getInstance
  ();
      newTime.set(0, 0, 0, hourOfDay, minute)
  ;
```

```
      timePickerFrom.setText(timeFormatter.
  format(newTime.getTime()));
    }
  }, hour, minute, false);

  timeEndPickerDialog = new TimePickerDialog(
  this, new TimePickerDialog.
  OnTimeSetListener(){
    @Override
    public void onTimeSet(TimePicker view,
  int hourOfDay, int minute) {
      Calendar newTime = Calendar.getInstance
  ();
      newTime.set(0, 0, 0, hourOfDay, minute)
  ;
      timePickerTo.setText(timeFormatter.
  format(newTime.getTime()));
    }
  }, hour, minute, false);
}

//Listener For DatePicker
private class OnClickListenerDatePicker
  implements View.OnClickListener{
  @Override
  public void onClick(View view) {
    if(view == datePickerFrom) {
      dateStartPickerDialog.show();
    } else if(view == datePickerTo) {
      dateEndPickerDialog.show();
    }
  }
}

//Listener For TimePicker
private class OnClickListenerTimePicker
  implements View.OnClickListener{
  @Override
  public void onClick(View view) {
    if(view == timePickerFrom) {
      timeStartPickerDialog.show();
    } else if(view == timePickerTo) {
      timeEndPickerDialog.show();
    }
  }
}

private class OnItemSelectedListenerDisplay
  implements AdapterView.
  OnItemSelectedListener{
  TextView txtToDate = (TextView)
  findViewById(R.id.txtToDate);

  @Override
  public void onItemSelected(AdapterView<?>
  parent, View view, int position, long id) {
    if(!parent.getItemAtPosition(position).
  toString().equals("Once")){
      txtToDate.setVisibility(View.VISIBLE);
      datePickerTo.setVisibility(View.VISIBLE
  );
    }
  }

  @Override
  public void onNothingSelected(AdapterView
  <?> parent) {
  }
}

private class OnClickListenerAddTaskToDB
  implements View.OnClickListener{
  @Override
  public void onClick(View view) {
    final Context context = view.getContext()
  ;
    final EditText editTaskName = (EditText)
  findViewById(R.id.editTaskName);
    final EditText editTaskDesc = (EditText)
  findViewById(R.id.editTaskDesc);
    final EditText editFromDate = (EditText)
  findViewById(R.id.datePickerStart);
    final EditText editToDate = (EditText)
  findViewById(R.id.datePickerEnd);
    final EditText editFromTime = (EditText)
findViewById(R.id.timePickerStart);
  final EditText editToTime = (EditText)
findViewById(R.id.timePickerEnd);
  final Switch alarmSet = (Switch)
findViewById(R.id.alarmSwitch);
  final Switch motivSet = (Switch)
findViewById(R.id.motivSwitch);

  String taskName = editTaskName.getText().
toString();
  if(taskName.equalsIgnoreCase(""))
  {
    editTaskName.setHint("Please enter task
 name");
    editTaskName.setError("Please enter
task name");
    Toast.makeText(context, "Task name is
required!", Toast.LENGTH_SHORT).show();
  }
  else
  {
    String taskDesc = editTaskDesc.getText
().toString();
    String taskCategory = categoryDropdown.
getSelectedItem().toString();
    int taskPriority = Integer.parseInt(
priorityDropdown.getSelectedItem().toString
());
    String repeatType = repeatDropdown.
getSelectedItem().toString();
    String startDate = editFromDate.getText
().toString();
    String endDate = editToDate.getText().
toString();
    String startTime = editFromTime.getText
().toString();
    String endTime = editToTime.getText().
toString();
    ObjectTask objectTask = new ObjectTask
();
    if(alarmSet.isChecked()){
      objectTask.setAlarmSet(1);
    }else{
      objectTask.setAlarmSet(0);
    }
    if(motivSet.isChecked()){
      objectTask.setMotivSet(1);
    }else{
      objectTask.setMotivSet(0);
    }
    objectTask.setTask_name(taskName);
    objectTask.setTask_description(taskDesc
);
    objectTask.setTask_category(
taskCategory);
    objectTask.setTask_priority(
taskPriority);
    objectTask.setTask_repeat(repeatType);
    InputProcessing inputProcessing = new
InputProcessing();

    if(startTime.equalsIgnoreCase("") ||
endTime.equalsIgnoreCase("")){
      editFromTime.setHint("Please enter
start time");
      editFromTime.setError("Please enter
start time");
      editToTime.setHint("Please enter end
time");
      editToTime.setError("Please enter end
 time");
      Toast.makeText(context, "Start and
end time are required!", Toast.LENGTH_SHORT
).show();
    }else{
      if(!repeatType.equals("Once")&&
endDate.equalsIgnoreCase("")){
        editToDate.setHint("Please enter
end date");
        editToDate.setError("Please enter
end date");
        Toast.makeText(context, "End date
is required!", Toast.LENGTH_SHORT).show();
```

```
      }else{
        objectTask = inputProcessing.
processingInputDates(objectTask, startDate,
 endDate, startTime, endTime, repeatType);
        if(objectTask.getRepeat_timeStart()
 > objectTask.getRepeat_timeEnd()){
          editFromTime.setHint("Start time
should be less than or equal the end time."
);
          editFromTime.setError("Enter
correct time");
          editToTime.setHint("End time
should be greater than or equal the start
time.");
          editToTime.setError("Enter
correct time");
          Toast.makeText(context, "Correct
time parameters are required!", Toast.
LENGTH_SHORT).show();
        }else {
          if(objectTask.getRepeat_start() >
 objectTask.getRepeat_end()){
            editToDate.setHint("End date
should be greater than start date.");
            editToDate.setError("Enter
appropriate date");
            Toast.makeText(context, "
Correct end date is required!", Toast.
LENGTH_SHORT).show();
          }else {
            if(inputType==1){
              int[] createSuccessful =
tblController.createTask(objectTask);
              if (createSuccessful[1]==1) {
                if(addToCalendar.isChecked
()){
                  Intent addToCalendar =
new Intent(getApplicationContext(),
AddToCalendar.class);
                  addToCalendar.putExtra("
TASK_ID", createSuccessful[0]);
                  addToCalendar.putExtra("
TASK_NAME", objectTask.getTask_name());
                  addToCalendar.putExtra("
TASK_DESC", objectTask.getTask_description
());
                  addToCalendar.putExtra("
REPEAT_TYPE", objectTask.getTask_repeat());
                  addToCalendar.putExtra("
START_DATE", objectTask.getRepeat_start());
                  addToCalendar.putExtra("
END_DATE", objectTask.getRepeat_end());
                  addToCalendar.putExtra("
START_TIME", (int) objectTask.
getRepeat_timeStart());
                  addToCalendar.putExtra("
END_TIME", (int) objectTask.
getRepeat_timeEnd());
                  addToCalendar.putExtra("
TO_GCAL", 1);
                  startActivity(
addToCalendar);
                }
                Toast.makeText(context, "
Task information was saved.", Toast.
LENGTH_SHORT).show();
                setAlarm();
              } else {
                Toast.makeText(context, "
Unable to save task information.", Toast.
LENGTH_SHORT).show();
              }
            }else{
              objectTask.setId(taskId);
              boolean updateSuccessful =
tblController.updateTask(objectTask);
              if(updateSuccessful){
                if(addToCalendar.isChecked
()){
                  Intent addToCalendar =
new Intent(getApplicationContext(),
AddToCalendar.class);
                  addToCalendar.putExtra("
    TASK_ID", objectTask.getId());
                  addToCalendar.putExtra("
    TASK_NAME", objectTask.getTask_name());
                  addToCalendar.putExtra("
    TASK_DESC", objectTask.getTask_description
    ());
                  addToCalendar.putExtra("
    REPEAT_TYPE", objectTask.getTask_repeat());
                  addToCalendar.putExtra("
    START_DATE", objectTask.getRepeat_start());
                  addToCalendar.putExtra("
    END_DATE", objectTask.getRepeat_end());
                  addToCalendar.putExtra("
    START_TIME", (int) objectTask.
    getRepeat_timeStart());
                  addToCalendar.putExtra("
    END_TIME", (int) objectTask.
    getRepeat_timeEnd());
                  addToCalendar.putExtra("
    TO_GCAL", 1);
                  startActivity(
    addToCalendar);
                }
                Toast.makeText(context, "
    Task record was updated.", Toast.
    LENGTH_SHORT).show();
                setAlarm();
              }else{
                Toast.makeText(context, "
    Unable to update task record.", Toast.
    LENGTH_SHORT).show();
              }
            }
            finish();
          }
        }
      }
    }
  }
}

  private void setAlarm(){
    Intent intent = new Intent(this,
    AlarmSetter.class);
    startService(intent);
  }
}
```

TaskMain.java

```
package com.specialproblem.brendz.
    timemanagement.view.tasks;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.graphics.Color;
import android.os.Bundle;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.text.TextUtils;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.SimpleCursorAdapter;
import android.widget.TextView;

import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .notification.AlarmService;
import com.specialproblem.brendz.timemanagement
    .notification.AlarmSetter;
```

```
import com.specialproblem.brendz.timemanagement
    .task.OnClickListenerTask;
import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .view.calendar.MonthView;

import java.util.Calendar;

/*
 * Activity displaying tasks for the selected
     day
 */
public class TaskMain extends AppCompatActivity
     {

  private int month, day, year;
  private TableController tblController;
  private ListView listView;

  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_task_main)
    ;
    initToolbar();
    tblController = TimeManagement.
    tblController;
    Button buttonCreateLocation = (Button)
    findViewById(R.id.buttonCreateTask);
    buttonCreateLocation.setOnClickListener(new
     OnClickListenerTask("createTask"));
    listView = (ListView) findViewById(R.id.
    listView);

    onReturnToActivity();
  }

  @Override
  public void onResume(){
    //When returning from previous Intent (Task
     Input)
    onReturnToActivity();
    super.onResume();
  }

  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_task_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem
     item) {
    ToolBarListener action = new
    ToolBarListener();
    return action.actionbar_action(item, this);
  }

  private void readRecords(){
    //Display records on the task screen
    Cursor cursor = tblController.readTasks(
    month, day, year);
    SimpleCursorAdapter adapter = new
    SimpleCursorAdapter(
        this,
        R.layout.display_taskmain,
        cursor,
        new String[]{"task_name", "repeat_start
    ", "repeat_start", "repeat_start", "
    repeat_timeStart", "task_priority"},
        new int[]{R.id.msg_tv, R.id.year_tv, R.
    id.month_tv, R.id.date_tv, R.id.time_tv, R.
    id.priority_tv});

    adapter.setViewBinder(new
    SimpleCursorAdapter.ViewBinder() {
  @Override
  public boolean setViewValue(View view,
Cursor cursor, int columnIndex) {

    if (view.getId() == R.id.msg_tv) return
 false;
    TextView tv = (TextView)view;
    RelativeLayout parent = (RelativeLayout
) tv.getParent();
    String[] infoTask = {cursor.getString(
cursor.getColumnIndex("_id")),Integer.
toString(month),
        Integer.toString(day),Integer.
toString(year)};
    parent.setTag(infoTask);
    Calendar cal = Calendar.getInstance();
    Long time = cursor.getLong(columnIndex)
*1000;
    cal.setTimeInMillis(time);
    switch(view.getId()) {
      case R.id.priority_tv:
        tv.setText("Priority Level: " +
cursor.getString(cursor.getColumnIndex("
task_priority")));
        break;
      case R.id.year_tv:
        tv.setText(Integer.toString(cal.get
(Calendar.YEAR)));
        break;
      case R.id.month_tv:
        int monthID = cal.get(Calendar.
MONTH);
        tv.setText(getString(MonthView.
monthIds[monthID]));
        break;
      case R.id.date_tv:
        tv.setText(Integer.toString(cal.get
(Calendar.DAY_OF_MONTH)));
        break;
      case R.id.time_tv:
        long now = System.currentTimeMillis
();
        int hour = cal.get(Calendar.HOUR);
        int minute = cal.get(Calendar.
MINUTE);
        String txtTime = (cal.get(Calendar.
HOUR_OF_DAY) < 12) ? " AM":" PM";
        String txtHour = (hour < 9) ? "0"+
Integer.toString(hour):Integer.toString(
hour);
        String txtMinute = (minute < 9) ? "
0"+Integer.toString(minute):Integer.
toString(minute);
        tv.setText(TextUtils.concat(txtHour
,":",txtMinute,txtTime));
        cal.set(year,month-1,day);
        time = cal.getTimeInMillis();
        TextView tv2 = (TextView) parent.
findViewById(R.id.msg_tv);
        if (time < now) tv2.setTextColor(
Color.parseColor("#B70000"));
        else tv2.setTextColor(Color.
parseColor("#587498"));
        break;
    }
    return true;
  }
});
listView.setAdapter(adapter);
listView.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
  @Override
  public void onItemClick(AdapterView<?>
parent, View view, int position, long id) {
    String[] infoTask = (String[]) view.
getTag();
    Intent viewTask = new Intent(view.
getContext(), TaskView.class);
    viewTask.putExtra("TASK_ID",Integer.
parseInt(infoTask[0]));
    viewTask.putExtra("CURR_MONTH",Integer.
parseInt(infoTask[1]));
    viewTask.putExtra("CURR_DAY",Integer.
parseInt(infoTask[2]));
```

```
        viewTask.putExtra("CURR_YEAR",Integer.
  parseInt(infoTask[3]));
        view.getContext().startActivity(
  viewTask);
      }
    });
    listView.setOnItemLongClickListener(new
  AdapterView.OnItemLongClickListener() {
      @Override
      public boolean onItemLongClick(
  AdapterView<?> parent, View view, int
  position, long id) {
        final View viewFin = view;
        final Context context = view.getContext
  ();
        final String[] infoTask = (String[])
  view.getTag();
        final CharSequence[] items = { "Task
  Overview", "Edit", "Delete" };
        new AlertDialog.Builder(context).
  setTitle("Task record")
             .setItems(items, new
  DialogInterface.OnClickListener() {
               public void onClick(
  DialogInterface dialog, int item) {
                 if (item == 0) {
                   Intent tOverview = new Intent
  (context, TaskOverview.class);
                   tOverview.putExtra("TASK_ID",
   infoTask[0]);
                   startActivity(tOverview);
                 }
                 else if (item == 1) {
                   editTaskRecord(Integer.
  parseInt(infoTask[0]), viewFin);
                 }
                 else if (item == 2) {
                   tblController.updateRequestID
  (Integer.parseInt(infoTask[0]),Integer.
  parseInt(infoTask[1])-1,Integer.parseInt(
  infoTask[2]),Integer.parseInt(infoTask[3])
  ,-1,0,1);
                   setAlarm();
                   onResume();
                 }
                 dialog.dismiss();
               }
             }).show();
        return false;
      }
    });
  }

  private void setAlarm(){
    Intent intent = new Intent(this,
  AlarmSetter.class);
    startService(intent);
  }

  public void editTaskRecord (final int taskId,
     final View viewFin) {
    final Context context = viewFin.getContext
  ();
    Intent editTask = new Intent(context,
  TaskInput.class);
    editTask.putExtra("INPUT_TYPE",2);
    editTask.putExtra("ID",taskId);
    context.startActivity(editTask);
  }

  private void setDate(){
    TextView textViewCurrDate = (TextView)
  findViewById(R.id.textViewCurrDate);
    String dispText = getString(MonthView.
  monthIds[month-1]) + " - " + day + " - " +
  year;
    textViewCurrDate.setText(dispText);
  }

  private void onReturnToActivity(){
    Bundle extras = getIntent().getExtras();
    if(extras!=null){
      int delete = extras.getInt("DELETE");
      if(AlarmService.notificationCounter!=null
  )
        AlarmService.notificationCounter.set(0)
  ;
      if(delete==1){
        finish();
        return;
      }else{
        year = extras.getInt("CURR_YEAR");
        month = extras.getInt("CURR_MONTH");
        day = extras.getInt("CURR_DAY");
      }
    }
    setDate();
    readRecords();
  }
}
```

TaskOverview.java

```
package com.specialproblem.brendz.
    timemanagement.view.tasks;

import android.content.Intent;
import android.database.Cursor;
import android.graphics.Color;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.text.TextUtils;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.SimpleCursorAdapter;
import android.widget.TextView;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .view.calendar.MonthView;

import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Locale;

/**
 * Acitivty showing specific task statistic
 */
public class TaskOverview extends
    AppCompatActivity {

  private TableController tblController;

  private TextView timeDiffStart;
  private TextView timeDiffEnd;
  private TextView lblStart;
  private TextView lblEnd;

  private int taskId;

  private int currMonth;
  private int currDay;
  private int currYear;

  private ListView listview;
  private int countStart;
  private int countEnd;

  private SimpleDateFormat sdf;

  private int time_diff_start;
  private int time_diff_end;

  @Override
  public void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.
    activity_task_overview);
    initToolbar();
    tblController = TimeManagement.
    tblController;
    Bundle extras = getIntent().getExtras();
    if(extras!=null) {
      taskId = Integer.parseInt(extras.
    getString("TASK_ID"));
    }
    listview = (ListView) findViewById(R.id.
    listView);
    timeDiffStart = (TextView) findViewById(R.
    id.txtStartTime);
    timeDiffEnd = (TextView) findViewById(R.id.
    txtEndTime);
    lblStart = (TextView) findViewById(R.id.
    lblStart);
    lblEnd = (TextView) findViewById(R.id.
    lblEnd);

    countStart = 0;
    countEnd = 0;
    sdf = new SimpleDateFormat("HH:mm", Locale.
    US);
    onReturnToActivity();
  }

  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
  }

  @Override
  public void onResume(){
    onReturnToActivity();
    super.onResume();
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_task_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem
     item) {
    ToolBarListener action = new
    ToolBarListener();
    return action.actionbar_action(item, this);
  }

  private void onReturnToActivity(){
    readTasks();
  }

  private void readTasks(){
    countStart = 0;
    countEnd = 0;
    time_diff_start = 0;
    time_diff_end = 0;
    Cursor cursor = tblController.getTaskFromId
    (taskId);
    SimpleCursorAdapter adapter = new
    SimpleCursorAdapter(
        this,
        R.layout.display_stat,
        cursor,
        new String[]{"task_name", "date", "date
    ", "date", "repeat_timeStart", "
    task_priority",
            "est_start", "act_start", "est_end"
    , "act_end", "time_diff_start", "
    time_diff_end"},
        new int[]{R.id.msg_tv, R.id.year_tv, R.
    id.month_tv, R.id.date_tv, R.id.time_tv, R.
    id.priority_tv,
            R.id.est_start_tv, R.id.
    act_start_tv, R.id.est_end_tv, R.id.
    act_end_tv, R.id.time_diff_start_tv, R.id.
    time_diff_end_tv});
    adapter.setViewBinder(new
    SimpleCursorAdapter.ViewBinder() {
      @Override
      public boolean setViewValue(View view,
Cursor cursor, int columnIndex) {
        if (view.getId() == R.id.msg_tv) return
 false;
        TextView tv = (TextView)view;
        RelativeLayout parent = (RelativeLayout
) tv.getParent();
        Calendar cal = Calendar.getInstance();
        Long time = 0L;
        int hour = 0;
        int minute = 0;
        String txtTime = null;
        String txtHour = null;
        String txtMinute = null;
        if(cursor.getString(columnIndex)!=null)
{
          time = cursor.getLong(columnIndex)
*1000;
          cal.setTimeInMillis(time);
          hour = cal.get(Calendar.HOUR);
          minute = cal.get(Calendar.MINUTE);
          txtTime = (cal.get(Calendar.
HOUR_OF_DAY) < 12) ? " AM":" PM";
          txtHour = (hour < 9) ? "0"+Integer.
toString(hour)+":":Integer.toString(hour)+"
:";
          txtMinute = (minute < 9) ? "0"+
Integer.toString(minute):Integer.toString(
minute);
        } else {
          txtTime = "-";
          txtHour = "";
          txtMinute = "";
        }
        switch(view.getId()) {
          case R.id.est_start_tv:
            tv.setText(TextUtils.concat("Est
start: ",txtHour,txtMinute,txtTime));
            break;
          case R.id.est_end_tv:
            tv.setText(TextUtils.concat("Est
end: ",txtHour,txtMinute,txtTime));
            break;
          case R.id.act_start_tv:
            tv.setText(TextUtils.concat("Act
start: ",txtHour,txtMinute,txtTime));
          case R.id.act_end_tv:
            tv.setText(TextUtils.concat("Act
end: ",txtHour,txtMinute,txtTime));
            break;
          case R.id.time_diff_start_tv:
            if(cursor.getString(cursor.
getColumnIndex("time_diff_start"))!=null){
              int start = Integer.parseInt(
cursor.getString(cursor.getColumnIndex("
time_diff_start")));
              if(start > 0){
                cal.setTimeInMillis(start*1000)
;
                cal.add(Calendar.HOUR_OF_DAY,
-8);
                tv.setText("Difference Start:+"
 + sdf.format(cal.getTimeInMillis()));
              }else{
                cal.setTimeInMillis(start
*-1000);
                cal.add(Calendar.HOUR_OF_DAY,
-8);
                tv.setText("Difference Start:-"
 + sdf.format(cal.getTimeInMillis()));
              }
            }else{
              tv.setText("Difference Start
:+00:00");
            }
            break;
          case R.id.time_diff_end_tv:
            if(cursor.getString(cursor.
getColumnIndex("time_diff_start"))!=null) {
              int end = Integer.parseInt(cursor
.getString(cursor.getColumnIndex("
time_diff_end")));
              if (end > 0) {
                cal.setTimeInMillis(end * 1000)
;
                cal.add(Calendar.HOUR_OF_DAY,
-8);
            tv.setText("Difference End:+" +
 sdf.format(cal.getTimeInMillis()));
          } else {
            cal.setTimeInMillis(end *
-1000);
            cal.add(Calendar.HOUR_OF_DAY,
-8);
            tv.setText("Difference End:-" +
 sdf.format(cal.getTimeInMillis()));
          }
        } else {
          tv.setText("Difference End:-00:00
");
        }
        break;
      case R.id.priority_tv:
        tv.setText("Priority Level: " +
cursor.getString(cursor.getColumnIndex("
task_priority")));
        break;
      case R.id.year_tv:
        currMonth = cal.get(Calendar.MONTH)
+1;
        currDay = cal.get(Calendar.
DAY_OF_MONTH);
        currYear = cal.get(Calendar.YEAR);
        String[] infoTask = {cursor.
getString(cursor.getColumnIndex("_id")),
Integer.toString(currMonth),
            Integer.toString(currDay),
Integer.toString(currYear)};
        parent.setTag(infoTask);
        tv.setText(Integer.toString(cal.get
(Calendar.YEAR)));
        break;
      case R.id.month_tv:
        int monthID = cal.get(Calendar.
MONTH);
        tv.setText(getString(MonthView.
monthIds[monthID]));
        break;
      case R.id.date_tv:
        tv.setText(Integer.toString(cal.get
(Calendar.DAY_OF_MONTH)));
        break;
      case R.id.time_tv:
        long now = System.currentTimeMillis
();
        tv.setText(TextUtils.concat(txtHour
,txtMinute,txtTime));
        cal.set(currYear,currMonth-1,
currDay);
        time = cal.getTimeInMillis();
        TextView tv2 = (TextView) parent.
findViewById(R.id.msg_tv);
        if (time < now) tv2.setTextColor(
Color.parseColor("#B70000"));
        else tv2.setTextColor(Color.
parseColor("#587498"));
        break;
    }
    return true;
  }
});
listview.setAdapter(adapter);
listview.setOnItemClickListener(new
AdapterView.OnItemClickListener() {
  @Override
  public void onItemClick(AdapterView<?>
parent, View view, int position, long id) {
    String[] infoTask = (String[]) view.
getTag();
    Intent viewTask = new Intent(view.
getContext(), TaskView.class);
    viewTask.putExtra("TASK_ID",Integer.
parseInt(infoTask[0]));
    viewTask.putExtra("CURR_MONTH",Integer.
parseInt(infoTask[1]));
    viewTask.putExtra("CURR_DAY",Integer.
parseInt(infoTask[2]));
    viewTask.putExtra("CURR_YEAR",Integer.
parseInt(infoTask[3]));
    view.getContext().startActivity(
  viewTask);
    }
  });
  if(cursor.moveToFirst()){
    do {
      //Set average time
      if(cursor.getString(cursor.
  getColumnIndex("time_diff_start"))!=null){
        time_diff_start += Integer.parseInt(
  cursor.getString(cursor.getColumnIndex("
  time_diff_start")));
        countStart++;
      }
      if(cursor.getString(cursor.
  getColumnIndex("time_diff_end"))!=null){
        time_diff_end += Integer.parseInt(
  cursor.getString(cursor.getColumnIndex("
  time_diff_end")));
        countEnd++;
      }
    } while (cursor.moveToNext());
  }
  setAverageValues();
}

private void setAverageValues(){
  SimpleDateFormat sdf = new SimpleDateFormat
  ("HH:mm");
  Calendar cal = Calendar.getInstance();
  if(time_diff_start > 0){
    if(countStart > 0){
      cal.setTimeInMillis(time_diff_start
  *1000/countStart);
    }else{
      cal.setTimeInMillis(time_diff_start
  *1000);
    }
    cal.add(Calendar.HOUR_OF_DAY, -8);
    timeDiffStart.setText("+" + sdf.format(
  cal.getTimeInMillis()));
    lblStart.setText("Advanced by: (Hrs:Mins)
  ");
  }else{
    if(countStart > 0){
      cal.setTimeInMillis(time_diff_start
  *-1000/countStart);
    }else{
      cal.setTimeInMillis(time_diff_start
  *-1000);
    }
    cal.add(Calendar.HOUR_OF_DAY, -8);
    timeDiffStart.setText("-" + sdf.format(
  cal.getTimeInMillis()));
    lblStart.setText("Delayed by: (Hrs:Mins)"
  );
  }
  if(time_diff_end > 0){
    if(countEnd > 0){
      cal.setTimeInMillis(time_diff_end*1000/
  countEnd);
    }else{
      cal.setTimeInMillis(time_diff_end*1000)
  ;
    }
    cal.add(Calendar.HOUR_OF_DAY, -8);
    timeDiffEnd.setText("+" + sdf.format(cal.
  getTimeInMillis()));
    lblEnd.setText("Advanced by: (Hrs:Mins)")
  ;
  }else{
    if(countEnd > 0){
      cal.setTimeInMillis(time_diff_start
  *-1000/countEnd);
    }else{
      cal.setTimeInMillis(time_diff_start
  *-1000);
    }
    cal.add(Calendar.HOUR_OF_DAY, -8);
    timeDiffEnd.setText("-" + sdf.format(cal.
  getTimeInMillis()));
    lblEnd.setText("Delayed by: (Hrs:Mins)");
  }
}
```

```
}
```

TaskSearchChoice.java

```
package com.specialproblem.brendz.
    timemanagement.view.tasks;

import android.app.DatePickerDialog;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.text.InputType;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.RadioButton;
import android.widget.Spinner;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .database_controller.SearchTask;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .task.ObjectCategory;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.List;
import java.util.Locale;

/**
 * Activity for searching
 */
public class TaskSearchChoice extends
    AppCompatActivity {

  private String searchType;
  private TableController tblController;

  private EditText nameTask;
  private Spinner spinnerCategory;
  private Spinner priorityLevel;
  private LinearLayout linearLayoutDateRange;

  private EditText datePickerFrom;
  private EditText datePickerTo;
  private SimpleDateFormat dateFormatter;

  private DatePickerDialog
    dateStartPickerDialog;
  private DatePickerDialog dateEndPickerDialog;

  private ArrayAdapter<String> categoryAdapter;
  private ArrayAdapter<Integer> priorityAdapter
    ;

  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);

    setContentView(R.layout.
    activity_task_search_choice);
    initToolbar();

    tblController = TimeManagement.
    tblController;

    Button buttonCreateLocation = (Button)
    findViewById(R.id.nextSearchButton);
    nameTask = (EditText) findViewById(R.id.
    taskName);
    spinnerCategory = (Spinner) findViewById(R.
    id.spinnerCategory);
    priorityLevel = (Spinner) findViewById(R.id
    .spinnerPriority);
    linearLayoutDateRange = (LinearLayout)
    findViewById(R.id.linearLayoutDateRange);
    datePickerFrom = (EditText) findViewById(R.
    id.datePickerStart);
    datePickerFrom.setInputType(InputType.
    TYPE_NULL);

    datePickerTo = (EditText) findViewById(R.id
    .datePickerEnd);
    datePickerTo.setInputType(InputType.
    TYPE_NULL);

    loadSpinnerInputs();
    setDateTimeField();

    buttonCreateLocation.setOnClickListener(new
     OnClickSearchListener());
  }

  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);
  }

  private void loadSpinnerInputs() {
    List<ObjectCategory> categories =
    tblController.getAllCategories();
    List<String> category_descs = new ArrayList
    <String>();
    for (ObjectCategory obj : categories) {
      category_descs.add(obj.getCategory_desc()
    );
    }

    categoryAdapter = new ArrayAdapter<String>(
    this, android.R.layout.
    simple_spinner_dropdown_item,
    category_descs);
    categoryAdapter.setDropDownViewResource(
    android.R.layout.
    simple_spinner_dropdown_item);
    spinnerCategory.setAdapter(categoryAdapter)
    ;

    String[] array = getResources().
    getStringArray(R.array.priorityArray);
    Integer[] intArray = new Integer[array.
    length];
    for (int i = 0; i < array.length; i++) {
      intArray[i] = Integer.parseInt(array[i]);
    }
    priorityAdapter = new ArrayAdapter<Integer
    >(this, android.R.layout.
    simple_spinner_dropdown_item, intArray);
    priorityAdapter.setDropDownViewResource(
    android.R.layout.
    simple_spinner_dropdown_item);
    priorityLevel.setAdapter(priorityAdapter);
  }

  private void setDateTimeField() {
    dateFormatter = new SimpleDateFormat(
    TimeManagement.DEFAULT_DATE_FORMAT, Locale.
    US);

    datePickerFrom.setOnClickListener(new
    OnClickListenerDatePicker());
    datePickerTo.setOnClickListener(new
    OnClickListenerDatePicker());

    Calendar newCalendar = Calendar.getInstance
    ();
    dateStartPickerDialog = new
    DatePickerDialog(this, new DatePickerDialog
    .OnDateSetListener() {
      public void onDateSet(DatePicker view,
    int year, int monthOfYear, int dayOfMonth)
    {
        Calendar newDate = Calendar.getInstance
    ();
        newDate.set(year, monthOfYear,
    dayOfMonth);
        datePickerFrom.setText(dateFormatter.
    format(newDate.getTime()));
      }
    }, newCalendar.get(Calendar.YEAR),
  newCalendar.get(Calendar.MONTH),
  newCalendar.get(Calendar.DAY_OF_MONTH));

  dateEndPickerDialog = new DatePickerDialog(
  this, new DatePickerDialog.
  OnDateSetListener() {
    public void onDateSet(DatePicker view,
  int year, int monthOfYear, int dayOfMonth)
  {
      Calendar newDate = Calendar.getInstance
  ();
      newDate.set(year, monthOfYear,
  dayOfMonth);
      datePickerTo.setText(dateFormatter.
  format(newDate.getTime()));
    }
  }, newCalendar.get(Calendar.YEAR),
  newCalendar.get(Calendar.MONTH),
  newCalendar.get(Calendar.DAY_OF_MONTH));
}

private class OnClickListenerDatePicker
  implements View.OnClickListener{
  @Override
  public void onClick(View view) {
    if(view == datePickerFrom) {
      dateStartPickerDialog.show();
    } else if(view == datePickerTo) {
      dateEndPickerDialog.show();
    }
  }
}

public void onRadioButtonClicked(View view) {
  boolean checked = ((RadioButton) view).
  isChecked();
  switch(view.getId()) {
    case R.id.nameTask:
      if (checked) {
        nameTask.setVisibility(View.VISIBLE);
        spinnerCategory.setVisibility(View.
  GONE);
        priorityLevel.setVisibility(View.GONE
  );
        linearLayoutDateRange.setVisibility(
  View.GONE);
        searchType = "name";
      }
      break;
    case R.id.categoryTask:
      if (checked){
        nameTask.setVisibility(View.GONE);
        spinnerCategory.setVisibility(View.
  VISIBLE);
        priorityLevel.setVisibility(View.GONE
  );
        linearLayoutDateRange.setVisibility(
  View.GONE);
        searchType = "category";
      }
      break;
    case R.id.priorityLevel:
      if (checked){
        nameTask.setVisibility(View.GONE);
        spinnerCategory.setVisibility(View.
  GONE);
        priorityLevel.setVisibility(View.
  VISIBLE);
        linearLayoutDateRange.setVisibility(
  View.GONE);
        searchType = "priority";
      }
      break;
    case R.id.dateRange:
      if (checked){
        nameTask.setVisibility(View.GONE);
        spinnerCategory.setVisibility(View.
  GONE);
        priorityLevel.setVisibility(View.GONE
  );
        linearLayoutDateRange.setVisibility(
  View.VISIBLE);
        searchType = "date";
      }
      break;
    }
  }

  private class OnClickSearchListener
    implements View.OnClickListener {
    @Override
    public void onClick(View view) {
      final EditText taskName = (EditText)
    findViewById(R.id.taskName);
      final EditText fromDate = (EditText)
    findViewById(R.id.datePickerStart);
      final EditText toDate = (EditText)
    findViewById(R.id.datePickerEnd);
      Intent searchTask = new Intent(view.
    getContext(), SearchTask.class);
      searchTask.putExtra("SEARCH_TYPE",
    searchType);
      switch(searchType){
        case "name":
          searchTask.putExtra("NAME", taskName.
    getText().toString());
          break;
        case "category":
          searchTask.putExtra("CATEGORY",
    spinnerCategory.getSelectedItem().toString
    ());
          break;
        case "priority":
          searchTask.putExtra("PRIORITY",
    priorityLevel.getSelectedItem().toString())
    ;
          break;
        case "date":
          searchTask.putExtra("DATE_START",
    fromDate.getText().toString());
          searchTask.putExtra("DATE_END",
    toDate.getText().toString());
          break;
      }
      view.getContext().startActivity(
    searchTask);
    }
  }
}
```

TaskView.java

```
package com.specialproblem.brendz.
    timemanagement.view.tasks;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity
    ;
import android.support.v7.widget.Toolbar;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Button;
import android.widget.CompoundButton;
import android.widget.Switch;
import android.widget.TextView;

import com.specialproblem.brendz.timemanagement
    .R;
import com.specialproblem.brendz.timemanagement
    .TimeManagement;
import com.specialproblem.brendz.timemanagement
    .actionbar.ToolBarListener;
import com.specialproblem.brendz.timemanagement
    .database_controller.InputProcessing;
import com.specialproblem.brendz.timemanagement
    .database_controller.TableController;
import com.specialproblem.brendz.timemanagement
    .googlecalendar.AddToCalendar;
import com.specialproblem.brendz.timemanagement
    .notification.AlarmService;
import com.specialproblem.brendz.timemanagement
    .notification.AlarmSetter;
import com.specialproblem.brendz.timemanagement
    .task.ObjectTask;
import com.specialproblem.brendz.timemanagement
    .task.ObjectTaskMeta;
import com.specialproblem.brendz.timemanagement
    .task.OnClickListenerTask;
import com.specialproblem.brendz.timemanagement
    .view.calendar.MonthView;
```

```
import java.util.Calendar;

/*
 *  Activity for viewing specific task for the
     specific day
 */
public class TaskView extends AppCompatActivity
     {

  private int taskID;
  private int alarmSet;
  private int motivSet;
  private int toGcal;
  private TableController tblController;
  private int month, day, year;

  private Button startTask;
  private Button endTask;

  private Switch alarmSwitch;
  private Switch motivSwitch;
  private Switch toGcalSwitch;

  private ObjectTask objectTask;
  private ObjectTaskMeta objectTaskMeta;

  @Override
  protected void onCreate(Bundle
    savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_task_view)
    ;
    initToolbar();

    tblController = TimeManagement.
    tblController;

    Bundle extras = getIntent().getExtras();
    if(extras!=null){
      int delete = extras.getInt("DELETE");
      if(AlarmService.notificationCounter!=null
    )
        AlarmService.notificationCounter.set(0)
    ;
      if(delete==1){
        finish();
        return;
      }else {
        taskID = extras.getInt("TASK_ID");
        month = extras.getInt("CURR_MONTH") -
    1;
        day = extras.getInt("CURR_DAY");
        year = extras.getInt("CURR_YEAR");
      }
    }

    startTask = (Button) findViewById(R.id.
    startActual);
    endTask = (Button) findViewById(R.id.
    endActual);
    alarmSwitch = (Switch) findViewById(R.id.
    alarmSwitch);
    motivSwitch = (Switch) findViewById(R.id.
    motivSwitch);
    toGcalSwitch = (Switch) findViewById(R.id.
    toCalendarSwitch);
    startTask.setTag(taskID);
    endTask.setTag(taskID);

    startTask.setOnClickListener(new
    OnClickListenerTask("startActual",month,day
    ,year));
    endTask.setOnClickListener(new
    OnClickListenerTask("endActual",month,day,
    year));

    setDetails();
    setDate();
  }

  // Toolbar Functions
  public void initToolbar(){
    Toolbar toolbar = (Toolbar) findViewById(R.
    id.toolbar);
    setSupportActionBar(toolbar);  }

  @Override
  public void onResume() {
    setDetails();
    setDate();
    super.onResume();
  }

  @Override
  public boolean onCreateOptionsMenu(Menu menu)
     {
    getMenuInflater().inflate(R.menu.
    menu_task_main, menu);
    return true;
  }

  @Override
  public boolean onOptionsItemSelected(MenuItem
     item) {
    ToolBarListener action = new
    ToolBarListener();
    return action.actionbar_action(item, this);
  }

  private void setDetails(){
    Calendar cal = Calendar.getInstance();
    cal.set(Calendar.MONTH, month);
    cal.set(Calendar.DAY_OF_MONTH, day);
    cal.set(Calendar.YEAR, year);
    cal.set(Calendar.HOUR_OF_DAY,0);
    cal.set(Calendar.MINUTE,0);
    cal.set(Calendar.SECOND,0);
    objectTask = tblController.readSingleRecord
    (taskID);
    objectTaskMeta = tblController.
    getTaskMetaDetails(taskID, cal.
    getTimeInMillis()/1000);

    InputProcessing inputProcessing = new
    InputProcessing();
    objectTask = inputProcessing.
    editInputProcessing(objectTask);
    objectTaskMeta = inputProcessing.
    editInputMetaProcessing(objectTaskMeta);

    alarmSet = objectTaskMeta.getAlarmSet();
    motivSet = objectTaskMeta.getMotivSet();
    toGcal = objectTask.getToCal();

    if(alarmSet==1){
      alarmSwitch.setChecked(true);
    }else{
      alarmSwitch.setChecked(false);
    }
    if(motivSet==1){
      motivSwitch.setChecked(true);
    }else{
      motivSwitch.setChecked(false);
    }
    if(toGcal==1){
      toGcalSwitch.setChecked(true);
    }else{
      toGcalSwitch.setChecked(false);
    }

    alarmSwitch.setOnCheckedChangeListener(new
    OnCheckedChangeAlarm());
    motivSwitch.setOnCheckedChangeListener(new
    OnCheckedChangeMotiv());
    toGcalSwitch.setOnCheckedChangeListener(new
     OnCheckedChangeToCalendar());
    //Task Details
    TextView taskName = (TextView) findViewById
    (R.id.txtTaskName);
    TextView taskDesc = (TextView) findViewById
    (R.id.txtTaskDesc);
    TextView taskCategory = (TextView)
    findViewById(R.id.txtCategory);
    TextView taskPriority = (TextView)
    findViewById(R.id.txtPriority);
    TextView taskTimeStart = (TextView)
    findViewById(R.id.txtTimeStart);
    TextView taskTimeEnd = (TextView)
    findViewById(R.id.txtTimeEnd);
    TextView taskRepeat = (TextView)
    findViewById(R.id.txtRepeat);
  TextView taskDateStart = (TextView)
  findViewById(R.id.txtDateStart);
  TextView taskDateEnd = (TextView)
  findViewById(R.id.txtDateEnd);

  //Task meta Details
  TextView metaEstStart = (TextView)
  findViewById(R.id.estStart);
  TextView metaEstEnd = (TextView)
  findViewById(R.id.estEnd);
  TextView metaActStart = (TextView)
  findViewById(R.id.actStart);
  TextView metaActEnd = (TextView)
  findViewById(R.id.actEnd);

  taskName.setText(objectTask.getTask_name())
  ;
  taskDesc.setText(objectTask.
  getTask_description());
  taskCategory.setText(objectTask.
  getTask_category());
  taskPriority.setText(Integer.toString(
  objectTask.getTask_priority()));
  taskTimeStart.setText(objectTask.
  getTimeStart());
  taskTimeEnd.setText(objectTask.getTimeEnd()
  );
  taskRepeat.setText(objectTask.
  getTask_repeat());
  taskDateStart.setText(objectTask.
  getDateStart());
  taskDateEnd.setText(objectTask.getDateEnd()
  );

  metaEstStart.setText(objectTaskMeta.
  getStr_est_start());
  metaEstEnd.setText(objectTaskMeta.
  getStr_est_end());
  metaActStart.setText(objectTaskMeta.
  getStr_act_start());
  metaActEnd.setText(objectTaskMeta.
  getStr_act_end());
}

private void setAlarm(){
  Intent intent = new Intent(this,
  AlarmSetter.class);
  startService(intent);
}

private void setDate(){
  TextView textViewCurrDate = (TextView)
  findViewById(R.id.textViewCurrDate);
  String dispText = getString(MonthView.
  monthIds[month]) + " - " + day + " - " +
  year;
  textViewCurrDate.setText(dispText);
}

private class OnCheckedChangeAlarm implements
   CompoundButton.OnCheckedChangeListener {
  @Override
  public void onCheckedChanged(CompoundButton
   buttonView, boolean isChecked) {
    if(isChecked){
      if(!tblController.alarmChange(taskID,
  month,day,year,1)){
        int requestID = (int) System.
  currentTimeMillis();
        tblController.updateRequestID(taskID,
  month,day,year,requestID,1,0);
      }
    } else {
      if(!tblController.alarmChange(taskID,
  month,day,year,0)){
        int requestID = tblController.
  getRequestID(taskID, month, day, year);
        tblController.updateRequestID(taskID,
  month,day,year, requestID,0,0);
      }
    }
    setAlarm();
  }
}

private class OnCheckedChangeMotiv implements
   CompoundButton.OnCheckedChangeListener {
  @Override
  public void onCheckedChanged(CompoundButton
   buttonView, boolean isChecked) {
    if(isChecked){
      if(!tblController.motivChange(taskID,
  month,day,year,1)){
        int requestID = (int) System.
  currentTimeMillis();
        tblController.updateRequestMotivID(
  taskID,month,day,year,requestID,1);
      }
    } else {
      if(!tblController.alarmChange(taskID,
  month,day,year,0)){
        int requestID = tblController.
  getRequestMotivID(taskID, month, day, year)
  ;
        tblController.updateRequestMotivID(
  taskID,month,day,year,requestID,0);
      }
    }
    setAlarm();
  }
}

private class OnCheckedChangeToCalendar
  implements CompoundButton.
  OnCheckedChangeListener {
  @Override
  public void onCheckedChanged(CompoundButton
   buttonView, boolean isChecked) {
    if(isChecked){
      Intent addToCalendar = new Intent(
  getApplicationContext(), AddToCalendar.
  class);
      addToCalendar.putExtra("TASK_ID",
  taskID);
      addToCalendar.putExtra("TASK_NAME",
  objectTask.getTask_name());
      addToCalendar.putExtra("TASK_DESC",
  objectTask.getTask_description());
      addToCalendar.putExtra("REPEAT_TYPE",
  objectTask.getTask_repeat());
      addToCalendar.putExtra("START_DATE",
  objectTask.getRepeat_start());
      addToCalendar.putExtra("END_DATE",
  objectTask.getRepeat_end());
      addToCalendar.putExtra("START_TIME", (
  int) objectTask.getRepeat_timeStart());
      addToCalendar.putExtra("END_TIME", (int
  ) objectTask.getRepeat_timeEnd());
      addToCalendar.putExtra("TO_GCAL", 1);
      startActivity(addToCalendar);
    } else {
      Intent addToCalendar = new Intent(
  getApplicationContext(), AddToCalendar.
  class);
      addToCalendar.putExtra("TASK_ID",
  taskID);
      addToCalendar.putExtra("TO_GCAL", 0);
      startActivity(addToCalendar);
    }
  }
}
}
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android
    .com/apk/res/android"
  package="com.specialproblem.brendz.
    timemanagement" >

  <uses-permission android:name="android.
    permission.INTERNET" />
  <uses-permission android:name="android.
    permission.ACCESS_NETWORK_STATE" />
  <uses-permission android:name="android.
    permission.GET_ACCOUNTS" />
  <uses-permission android:name="android.
    permission.READ_EXTERNAL_STORAGE"/>
  <uses-permission android:name="android.
    permission.WRITE_EXTERNAL_STORAGE"/>
  <uses-permission android:name="android.
    permission.WAKE_LOCK" />
  <uses-permission android:name="android.
    permission.RECEIVE_BOOT_COMPLETED"/>
  <uses-permission android:name="android.
    permission.VIBRATE" />

  <!-- Application -->
  <application
    android:name=".TimeManagement"
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/Theme.AppCompat" >

    <!-- Calendar Activity -->
    <activity
      android:name=".view.calendar.CalendarMain
    "
      android:label="@string/app_name"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
    </activity>

    <!-- Task Activities -->
    <activity
      android:name=".view.tasks.TaskDashboard"
      android:label="@string/app_name"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
      <intent-filter>
        <action android:name="android.intent.
    action.MAIN" />
        <category android:name="android.intent.
    category.LAUNCHER" />
      </intent-filter>
    </activity>
    <activity
      android:name=".view.tasks.TaskMain"
      android:label="@string/app_name"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
    </activity>
    <activity
      android:name=".view.tasks.TaskInput"
      android:label="@string/
    title_activity_task_input"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
      <meta-data
        android:name="android.support.
    PARENT_ACTIVITY"
        android:value=".view.tasks.TaskMain" />
    </activity>
    <activity
      android:name=".view.tasks.TaskView"
      android:label="@string/
    title_activity_task_view"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
      <meta-data
        android:name="android.support.
    PARENT_ACTIVITY"
        android:value=".view.tasks.TaskMain" />
    </activity>
    <activity
      android:name=".view.tasks.TaskOverview"
      android:label="@string/
    title_activity_task_view"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
    </activity>

    <!-- Category Activities-->
    <activity
      android:name=".view.tasks.CategoryInput"
      android:label="@string/app_name"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
    </activity>
    <activity
      android:name=".view.tasks.CategoryStat"
      android:label="@string/app_name"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
    </activity>
    <!-- Motivation Activity -->
    <activity
      android:name=".view.motivation.
    MotivationMain"
      android:label="@string/app_name"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
    </activity>

    <!-- Search Activity -->
    <activity
      android:name=".view.tasks.
    TaskSearchChoice"
      android:label="Search"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
    </activity>
    <activity android:name=".
    database_controller.SearchTask"
      android:launchMode="singleTop"
      android:theme="@style/Theme.AppCompat.
    NoActionBar">
    </activity>

    <!-- Add to Google Calendar -->
    <activity
      android:name=".googlecalendar.
    AddToCalendar"
      android:label="Add to Google Calendar"
      android:theme="@style/Theme.AppCompat.
    NoActionBar" >
    </activity>
    <!-- Notification Services -->
    <service android:enabled="true"
    android:name=".notification.AlarmService"/>
    <service android:enabled="true"
    android:name=".notification.AlarmSetter"/>
    <service android:enabled="true"
    android:name=".notification.
    MotivationService"/>

    <!-- Notification Receivers -->
    <receiver android:name=".notification.
    AlarmReceiver" />
    <receiver android:name=".notification.
    AlarmSetterReceiver" />
    <receiver android:name=".notification.
    MotivationReceiver" />
    <receiver
      android:name=".notification.
    ServiceStartOnBootReceiver"
      android:enabled="true" >
      <intent-filter>
        <action android:name="android.intent.
    action.BOOT_COMPLETED" />
      </intent-filter>
    </receiver>

    <!-- REMOVE; For Database Viewing only -->

    <meta-data
      android:name="com.google.android.gms.
    version"
      android:value="@integer/
    google_play_services_version" />
  </application>

</manifest>
```

calheader.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.
```

```
    com/apk/res/android"
  android:shape="rectangle"
  android:useLevel="false">
  <solid
    android:color="#435259"/>
  <stroke
    android:width="1dp"
    android:color="#999"/>
</shape>
```

current_day.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.
    com/apk/res/android"
  android:shape="rectangle"
  android:useLevel="false">
  <stroke
    android:width="3dp"
    android:color="#84BEE6"/>
  <solid
    android:color="#eee"/>
</shape>
```

dayinmonth.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<layer-list xmlns:android="http://schemas.
    android.com/apk/res/android">
  <item android:drawable="@drawable/rectgrad"/>
  <item>
    <bitmap android:src="@drawable/event"
      android:gravity="bottom|right" />
  </item>
</layer-list>
```

layout_border.xml

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list xmlns:android="http://schemas.
    android.com/apk/res/android">
  <item>
    <shape android:shape="rectangle">
      <solid android:color="#FFFFFF" />
    </shape>
  </item>
  <item android:left="1dp" android:right="1dp"
    android:top="1dp" android:bottom="1dp">
    <shape android:shape="rectangle">

    </shape>
  </item>
</layer-list>
```

rectgrad.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.
    com/apk/res/android"
  android:shape="rectangle"
  android:useLevel="false">
  <solid
    android:color="#555"/>
  <stroke
    android:width="0.3dp"
    android:color="#999999"/>
</shape>
```

selectedgrad.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.
    com/apk/res/android"
  android:shape="rectangle"
  android:useLevel="false">
  <stroke
    android:width="3dp"
    android:color="#84BEE6"/>
</shape>
```

menu_search.xml

```
<menu xmlns:android="http://schemas.android.com
    /apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " tools:context=".TaskMain">
  <item android:id="@+id/action_search"
    android:title="Search"
    android:icon="@drawable/ic_prev_btn"
    app:showAsAction="always"
    app:actionViewClass="android.support.v7.
    widget.SearchView" />
</menu>
```

menu_task_main.xml

```
<menu xmlns:android="http://schemas.android.com
    /apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " tools:context=".TaskMain">

  <!-- Add button -->
  <item android:id="@+id/action_add"
    android:icon="@drawable/ic_add_btn"
    android:title="@string/action_add"
    app:showAsAction="ifRoom"/>
  <item android:id="@+id/action_categories"
    android:title="@string/action_categories"
    android:orderInCategory="100"
    app:showAsAction="never" />
  <item android:id="@+id/action_search"
    android:title="Search"
    android:icon="@drawable/ic_prev_btn"
    app:showAsAction="collapseActionView"
    android:actionViewClass="android.support.v7
    .widget.ActionMenuView" />
</menu>
```

arrays.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string-array name="priorityArray">
    <item>1</item>
    <item>2</item>
    <item>3</item>
    <item>4</item>
    <item>5</item>
  </string-array>

  <string-array name="repeatArray">
    <item>Once</item>
    <item>Daily</item>
    <item>Weekly</item>
    <item>Monthly</item>
  </string-array>

  <string-array name="dashboardArray">
    <item>New/Upcoming tasks</item>
    <item>Delayed tasks</item>
    <item>High Priority tasks</item>
  </string-array>


  <string-array name="time_option_arr">
    <item>Actual</item>
    <item>Remaining</item>
  </string-array>

  <string-array name="date_range_arr">
    <item>Daily</item>
    <item>Weekly</item>
    <item>Monthly</item>
    <!--item>Yearly</item-->
  </string-array>

  <string-array name="val_arr">
```

```
    <item>0</item>
    <item>1</item>
    <item>2</item>
    <item>3</item>
  </string-array>

  <string-array name="date_format_arr">
    <item>yyyy-MM-dd</item>
    <item>yyyy-M-d</item>
    <item>MMMM dd yyyy</item>
    <item>MM/dd/yyyy</item>
    <item>dd MMMM yyyy</item>
    <item>dd MMM yyyy</item>
    <item>d MMM yyyy</item>
    <item>dd-MM-yyyy</item>
    <item>dd/MM/yy</item>
  </string-array>

</resources>
```

colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <item name="transparent" type="color">#00
    FFFFFF</item>
  <item name="grey" type="color">#555555</item>
  <item name="green" type="color">#4499CC00</
    item>
  <item name="darkgreen" type="color">#FF669900
    </item>
  <color name="colorView">#587498</color>
  <color name="colorPrimary">#3F51B5</color>
  <color name="colorPrimaryDark">#303F9F</color
    >
  <color name="colorAccent">#FFAB40</color>
</resources>
```

dimens.xml

```
<resources>
  <!-- Default screen margins, per the Android
    Design guidelines. -->
  <dimen name="activity_horizontal_margin">16dp
    </dimen>
  <dimen name="activity_vertical_margin">16dp</
    dimen>
  <dimen name="fab_margin">16dp</dimen>
</resources>
```

strings.xml

```
<resources>
  <string name="app_name">Time Management</
    string>
  <!-- Toolbar -->
  <string name="action_add">Add Task</string>
  <string name="action_settings">Settings</
    string>
  <string name="action_categories">Categories</
    string>
  <string name="action_search">Search</string>
  <!-- Activity Titles -->
  <string name="title_activity_calendar_main">
    Calendar</string>
  <string name="title_activity_task_input">
    Input Task</string>
  <string name="title_activity_task_view">View
    Task</string>
  <!-- Strings for Task Main -->
  <string name="no_records_yet">No records yet.
    </string>
  <string name="rec_found">" records found."</
    string>
  <!-- Day strings -->
  <string name="sunday">Sunday</string>
  <string name="monday">Monday</string>
  <string name="tuesday">Tuesday</string>
  <string name="wednesday">Wednesday</string>
  <string name="thursday">Thursday</string>
  <string name="friday">Friday</string>
  <string name="saturday">Saturday</string>
  <!-- Month strings -->
  <string name="january">January</string>
  <string name="february">February</string>
  <string name="march">March</string>
  <string name="april">April</string>
  <string name="may">May</string>
  <string name="june">June</string>
  <string name="july">July</string>
  <string name="august">August</string>
  <string name="september">September</string>
  <string name="october">October</string>
  <string name="november">November</string>
  <string name="december">December</string>

  <!-- Search -->
  <string name="search_hint">Search Task</
    string>
  <!-- Shown above search results when we
    receive a search request. -->
  <plurals name="search_results">
    <item quantity="one">%1$d result for \"%2$s
    \": </item>
    <item quantity="other">%1$d results for \"
    %2$s\": </item>
  </plurals>

  <!-- Search failure message. -->
  <string name="no_results">No results found
    for \"%s\"</string>
</resources>
```

styles.xml

```
<resources>

  <!-- Base application theme. -->
  <style name="AppTheme" parent="Theme.
    AppCompat.Light.DarkActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/
    colorPrimary</item>
    <item name="colorPrimaryDark">@color/
    colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent
    </item>
  </style>
  <style name="AppTheme.NoActionBar">
    <item name="windowActionBar">false</item>
    <item name="windowNoTitle">true</item>
  </style>
  <style name="AppTheme.AppBarOverlay" parent="
    ThemeOverlay.AppCompat.Dark.ActionBar" />
  <style name="AppTheme.PopupOverlay" parent="
    ThemeOverlay.AppCompat.Light" />

</resources>
```

styles.xml - v21

```
<resources>>
  <style name="AppTheme.NoActionBar">
    <item name="windowActionBar">false</item>
    <item name="windowNoTitle">true</item>
    <item name="
    android:windowDrawsSystemBarBackgrounds">
    true</item>
    <item name="android:statusBarColor">
    @android:color/transparent</item>
  </style>
</resources>
```

dimens.xml - w820dp

```
<resources>
  <!-- Example customization of dimensions
    originally defined in res/values/dimens.xml
     (such as screen margins) for screens with
    more than 820dp of available width. This
     would include 7" and 10" devices in
    landscape (~960dp and ~1280dp respectively)
    . -->
  <dimen name="activity_horizontal_margin">64dp
    </dimen>
</resources>
```

settings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen xmlns:android="http://schemas
     .android.com/apk/res/android" >
  <PreferenceCategory android:title="Display
    Settings">
    <ListPreference android:key="time_option"
      android:title="Display Time"
      android:summary="Actual"
      android:entries="@array/time_option_arr"
      android:entryValues="@array/val_arr"
      android:defaultValue="0" />
    <ListPreference android:key="date_range"
      android:title="Date Range"
      android:summary="Daily"
      android:entries="@array/date_range_arr"
      android:entryValues="@array/val_arr"
      android:defaultValue="0" />
    <ListPreference android:key="date_format"
      android:title="Date Format"
      android:summary="yyyy-M-d"
      android:entries="@array/date_format_arr"
      android:entryValues="@array/
    date_format_arr"
      android:defaultValue="yyyy-M-d" />
    <CheckBoxPreference android:key="
    time_format"
      android:title="Time Format"
      android:summary="Use 24-hour format"
      android:defaultValue="true" />
  </PreferenceCategory>
  <PreferenceCategory android:title="
    Notification Settings">
    <CheckBoxPreference android:key="
    vibrate_pref"
      android:title="Vibrate"
      android:summary="Vibrate on notification"
      android:defaultValue="true" />
    <RingtonePreference android:key="
    ringtone_pref"
      android:title="Set Ringtone"
      android:summary="Default"
      android:ringtoneType="all"
      android:showDefault="true"
      android:showSilent="false" />
    <!--Preference android:key="about_pref"
      android:title="About"
      android:summary="Credits" /-->
  </PreferenceCategory>
</PreferenceScreen>
```

activity_calendar_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.calendar.CalendarMain">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/
    content_calendar_main" />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_category_input.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.CategoryInput">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/
    content_category_input" />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_category_stat.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.CategoryStat">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/
    content_category_stat" />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_motivation_input.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.motivation.
    MotivationMain">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/
    content_motivation_input" />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_search_task.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".database_controller.
    SearchTask" >

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">
    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/content_search_task"
     />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_task_dashboard.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.TaskDashboard">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/
    content_task_dashboard" />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_task_input.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.TaskInput">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/content_task_input"
    />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_task_list_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.TaskListView">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/
    content_task_list_view" />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_task_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.TaskMain" >

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/content_task_main" /
    >

</android.support.design.widget.
    CoordinatorLayout>
```

activity_task_overview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
```

```
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.TaskOverview">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/
    content_task_overview" />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_task_search_choice.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.TaskSearchChoice"
    >

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/
    content_task_search_choice" />

</android.support.design.widget.
    CoordinatorLayout>
```

activity_task_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.
    CoordinatorLayout
  xmlns:android="http://schemas.android.com/apk
    /res/android"
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  xmlns:tools="http://schemas.android.com/tools
    " android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:fitsSystemWindows="true"
  tools:context=".view.tasks.TaskView">

  <android.support.design.widget.AppBarLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:theme="@style/AppTheme.
    AppBarOverlay">

    <android.support.v7.widget.Toolbar
    android:id="@+id/toolbar"
      android:layout_width="match_parent"
    android:layout_height="?attr/actionBarSize"
      android:background="?attr/colorPrimary"
    app:popupTheme="@style/AppTheme.
    PopupOverlay" />

  </android.support.design.widget.AppBarLayout>

  <include layout="@layout/content_task_view" /
    >

</android.support.design.widget.
    CoordinatorLayout>
```

content_calendar_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto" android:layout_width="match_parent"
  android:layout_height="match_parent"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/activity_calendar_main"
  tools:context=".view.calendar.CalendarMain">

  <TableLayout
    android:id="@+id/calendar"
    android:layout_margin="10dp"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

</RelativeLayout>
```

content_category_input.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/activity_category_input
    "
  tools:context=".view.tasks.CategoryInput">

  <Button
    android:id="@+id/buttonCreateCategory"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:text="Create Category" />
  <TextView
    android:id="@+id/
    textViewRecordCountCategory"
    android:gravity="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text=""
    android:padding="1dp" />
  <ScrollView
    android:id="@+id/scrollViewRecordsCategory"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/
    textViewRecordCountCategory" >
    <LinearLayout
```

```
      android:id="@+id/
    linearLayoutRecordsCategory"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="vertical" >
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

content_category_stat.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  android:weightSum="10"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/activity_category_stat"
  tools:context=".view.tasks.CategoryStat">

  <RelativeLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#000" >
    <Spinner
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/categorySpinner"
      android:spinnerMode="dropdown" />
  </RelativeLayout>

  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="2"
    android:orientation="vertical">
    <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:textAlignment="center"
      android:text="Average Time Difference"
      android:id="@+id/lblTimeDiff" />
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:weightSum="6"
      android:padding="10dp">
      <LinearLayout
        android:orientation="vertical"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="3">
        <TextView
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr
    /textAppearanceSmall"
          android:text="Start time"
          android:id="@+id/lblStart" />
        <TextView
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr
    /textAppearanceMedium"
          android:text="Start time"
          android:id="@+id/txtStartTime" />
      </LinearLayout>
      <LinearLayout
        android:orientation="vertical"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="3">
        <TextView
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr
    /textAppearanceSmall"
          android:text="End time"
          android:id="@+id/lblEnd" />
        <TextView
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr
    /textAppearanceMedium"
          android:text="End time"
          android:id="@+id/txtEndTime" />
      </LinearLayout>
    </LinearLayout>
  </LinearLayout>

  <TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/
    textAppearanceMedium"
    android:textAlignment="center"
    android:text=""
    android:id="@+id/lblTasksUnder" />
  <ListView android:id="@+id/listView"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="7"
    android:listSelector="@android:color/
    transparent"
    android:cacheColorHint="#00000000" />
</LinearLayout>
```

content_monthview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:background="#293039"
  android:paddingLeft="25dp"
  android:paddingRight="25dp">
  <ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerVertical="true"
    android:id="@+id/imgLeft"
    android:src="@drawable/ic_prev_btn"/>
  <LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:orientation="horizontal">
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:id="@+id/txtDay"
      android:textStyle="bold"
      android:text="Week"
      android:textColor="#fff"
      android:textSize="20sp"/>
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:paddingLeft="4dp"
      android:id="@+id/txtYear"
      android:textColor="#fff"
      android:textStyle="bold"
      android:textSize="18sp"/>
  </LinearLayout>
  <ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerVertical="true"
    android:id="@+id/imgRight"
    android:layout_alignParentRight="true"
    android:src="@drawable/ic_next_btn"/>
</RelativeLayout>
```

content_motivation_input.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/
    activity_motivation_input"
  tools:context=".view.motivation.
    MotivationMain">

  <Button
    android:id="@+id/buttonCreateMotivation"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:text="Create Motivation" />

  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:weightSum="5">
    <TextView
      android:id="@+id/textViewTimeSet"
      android:gravity="center"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:text=""
      android:padding="1dp"
      android:layout_weight="2"/>
    <Switch
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:text="Motivation Notification"
      android:id="@+id/motivSwitch"
      android:layout_gravity="right"
      android:layout_weight="3" />
  </LinearLayout>

  <TextView
    android:id="@+id/
    textViewRecordCountMotivation"
    android:gravity="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text=""
    android:padding="5dp" />

  <ScrollView
    android:id="@+id/
    scrollViewRecordsMotivation"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/
    textViewRecordCountMotivation" >
    <LinearLayout
      android:id="@+id/
    linearLayoutRecordsMotivation"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="vertical" >
    </LinearLayout>
  </ScrollView>
</LinearLayout>
```

content_search_task.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
/*
** Copyright 2010, The Android Open Source
   Project
**
** Licensed under the Apache License, Version
   2.0 (the "License");
** you may not use this file except in
   compliance with the License.
** You may obtain a copy of the License at
**
**    http://www.apache.org/licenses/LICENSE-2.0
**
** Unless required by applicable law or agreed
   to in writing, software
** distributed under the License is distributed
    on an "AS IS" BASIS,
** WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND
   , either express or implied.
** See the License for the specific language
   governing permissions and
** limitations under the License.
*/
-->
<!-- Layout for SearchableActivity.
 -->
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent">
  <TextView
    android:id="@+id/text"
    android:textColor="?
    android:textColorPrimary"
    android:textSize="17dp"
    android:text="@string/search_hint"
    android:background="@android:drawable/
    title_bar"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" />
  <ListView
    android:id="@+id/list"
    android:layout_width="fill_parent"
    android:layout_height="10dp"
    android:layout_weight="1" />
</LinearLayout>
```

content_task_dashboard.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  android:weightSum="10"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/activity_task_dashboard
    "
  tools:context=".view.tasks.TaskDashboard" >

  <RelativeLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#000" >
    <Spinner
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/dashboardSpinner"
      android:spinnerMode="dropdown" />
  </RelativeLayout>
  <ListView android:id="@+id/listView"
      android:layout_width="match_parent"
      android:layout_height="0dp"
      android:layout_weight="7"
      android:listSelector="@android:color/
    transparent"
      android:cacheColorHint="#00000000" />
  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="3"
    android:id="@+id/buttonView"
    android:orientation="vertical" >
    <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:weightSum="2">
      <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="Calendar"
        android:id="@+id/calendarShow"
        android:layout_gravity="
    center_horizontal"
        android:layout_weight="1" />
      <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="Today's Tasks"
        android:id="@+id/currentDayShow"
        android:layout_gravity="
    center_horizontal"
        android:layout_weight="1"
        />
    </LinearLayout>
    <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:weightSum="2">
      <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="Category Statistics"
        android:id="@+id/categoryStatShow"
        android:layout_gravity="
    center_horizontal"
        android:layout_weight="1" />
      <Button
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:text="Motivations"
        android:id="@+id/motivationShow"
        android:layout_gravity="
    center_horizontal"
        android:layout_weight="1"
        />
    </LinearLayout>
  </LinearLayout>
</LinearLayout>
```

content_task_input.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto" android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:paddingLeft="@dimen/
    activity_horizontal_margin"
  android:paddingRight="@dimen/
    activity_horizontal_margin"
  android:paddingTop="@dimen/
    activity_vertical_margin"
  android:paddingBottom="@dimen/
    activity_vertical_margin"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/activity_task_input"
  tools:context=".view.tasks.TaskInput">

  <LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignParentTop="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true">

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:text="Task Name"
      android:id="@+id/txtTaskName" />

    <EditText
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/editTaskName"
      android:layout_gravity="center_horizontal
    " />

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:text="Task Description"
      android:id="@+id/txtTaskDesc" />

    <EditText
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:id="@+id/editTaskDesc"
      android:layout_gravity="center_horizontal
    " />

    <TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textAppearance="?android:attr/
textAppearanceMedium"
  android:text="Category"
  android:id="@+id/txtCategory" />

<Spinner
  android:id="@+id/spinnerCategory"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:background="@android:drawable/
btn_dropdown"
  android:spinnerMode="dropdown" />

<TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textAppearance="?android:attr/
textAppearanceMedium"
  android:text="Priority Level (1 - highest
, 5 - lowest)"
  android:id="@+id/txtPriority" />

<Spinner
  android:id="@+id/spinnerPriority"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:background="@android:drawable/
btn_dropdown"
  android:spinnerMode="dropdown" />

<LinearLayout
  android:layout_height="wrap_content"
  android:layout_width="match_parent"
  android:layout_weight="1"
  android:orientation="horizontal" >

  <LinearLayout
     android:layout_width="0dp"
     android:layout_height="wrap_content"
     android:orientation="vertical"
     android:layout_weight="0.5" >

     <TextView
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:textAppearance="?
android:attr/textAppearanceMedium"
       android:text="Time From"
       android:id="@+id/txtTimeStart" />

     <EditText
       android:id="@+id/timePickerStart"
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:layout_gravity="
center_horizontal"
       android:focusableInTouchMode="false"
 />
   </LinearLayout>

   <LinearLayout
     android:layout_width="0dp"
     android:layout_height="wrap_content"
     android:orientation="vertical"
     android:layout_weight="0.5" >

     <TextView
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:textAppearance="?
android:attr/textAppearanceMedium"
       android:text="Time End"
       android:id="@+id/txtTimeEnd" />

     <EditText
       android:id="@+id/timePickerEnd"
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:layout_gravity="
center_horizontal"
       android:focusableInTouchMode="false"
 />
    </LinearLayout>

</LinearLayout>

<LinearLayout
android:layout_height="wrap_content"
android:layout_width="match_parent"
```

```
android:weightSum="5"
android:orientation="horizontal" >
  <LinearLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_weight="3" >
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr
/textAppearanceMedium"
      android:text="Date"
      android:id="@+id/txtFromDate" />

    <EditText
      android:id="@+id/datePickerStart"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="
center_horizontal"
      android:focusableInTouchMode="false"
/>
  </LinearLayout>

  <RelativeLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_weight="2" >
    <CheckBox
      android:layout_width="wrap_content"
      android:layout_height="match_parent"
      android:text=""
      android:id="@+id/toCalendarSwitch"
      android:layout_gravity="right"
      android:showText="false"
      android:scaleX="1.25"
      android:scaleY="1.25"/>
    <TextView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:layout_toRightOf="@id/
toCalendarSwitch"
      android:textAppearance="?android:attr
/textAppearanceSmall"
      android:textAlignment="center"
      android:text="Add to Google Calendar"
      android:id="@+id/txtGoogleCalendar"
      android:padding="5dp"
      />
  </RelativeLayout>
</LinearLayout>

<LinearLayout
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:weightSum="2"
  android:orientation="horizontal">
  <LinearLayout
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:orientation="horizontal">
    <RelativeLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
      <Switch
        android:layout_width="wrap_content"
        android:layout_height="match_parent
"
        android:text=""
        android:id="@+id/alarmSwitch"
        android:layout_gravity="left"
        android:layout_weight="1"
        android:showText="false" />
      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent
"
        android:layout_toRightOf="@id/
alarmSwitch"
        android:textAppearance="?
android:attr/textAppearanceSmall"
        android:textAlignment="center"
        android:text="Alarm"
        android:id="@+id/txtAlarm"
        android:padding="5dp"
        />
    </RelativeLayout>
  </LinearLayout>
  <LinearLayout
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:orientation="horizontal">
    <RelativeLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content">
    <Switch
      android:layout_width="wrap_content"
      android:layout_height="match_parent"
      android:text=""
      android:id="@+id/motivSwitch"
      android:layout_gravity="left"
      android:layout_weight="1"
      android:showText="false" />
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="match_parent"
      android:layout_toRightOf="@id/
motivSwitch"
      android:textAppearance="?android:attr
/textAppearanceSmall"
      android:textAlignment="center"
      android:text="Turn on motivations"
      android:id="@+id/txtMotiv"
      android:padding="5dp"
      />
    </RelativeLayout>
  </LinearLayout>
</LinearLayout>

<TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textAppearance="?android:attr/
textAppearanceMedium"
  android:text="Repeat"
  android:id="@+id/txtRepeat" />

<Spinner
  android:id="@+id/spinnerRepeat"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:background="@android:drawable/
btn_dropdown"
  android:spinnerMode="dropdown" />

<TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textAppearance="?android:attr/
textAppearanceMedium"
  android:text="End Date"
  android:id="@+id/txtToDate"
  android:visibility="gone" />

<EditText
  android:id="@+id/datePickerEnd"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:layout_gravity="center_horizontal
"
  android:focusableInTouchMode="false"
  android:visibility="gone" />

<Button
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="Add"
  android:id="@+id/inputButton"
  android:layout_gravity="center_horizontal
" />

  </LinearLayout>
</ScrollView>
```

content_task_list_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  android:orientation="horizontal"
    android:layout_width="match_parent"
  android:layout_height="match_parent">

  <LinearLayout
    android:orientation="vertical"
    android:layout_width="wrap_content"
    android:layout_height="match_parent">
  </LinearLayout>

</LinearLayout>
```

content_task_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/activity_task_main"
  tools:context=".view.tasks.TaskMain">

  <Button
    android:id="@+id/buttonCreateTask"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:text="Create Task" />
  <TextView
    android:id="@+id/textViewCurrDate"
    android:gravity="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text=""
    android:padding="1dp" />
   <ListView android:id="@+id/listView"
    android:layout_width="fill_parent"
     android:layout_height="fill_parent"
     android:listSelector="@android:color/
    transparent"
     android:cacheColorHint="#00000000" />
</LinearLayout>
```

content_task_overview.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  android:weightSum="10"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/activity_task_overview"
  tools:context=".view.tasks.TaskOverview">
  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/
    textAppearanceLarge"
    android:textAlignment="center"
    android:text=""
    android:id="@+id/lblTaskname" />

  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="2"
    android:orientation="vertical">
    <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:textAlignment="center"
      android:text="Average Time Difference"
      android:id="@+id/lblTimeDiff" />
    <LinearLayout
      android:orientation="horizontal"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:weightSum="6"
      android:padding="10dp">
      <LinearLayout
        android:orientation="vertical"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="3">
        <TextView
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr
    /textAppearanceSmall"
          android:text="Start time"
          android:id="@+id/lblStart" />
        <TextView
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr
    /textAppearanceMedium"
          android:text="Start time"
          android:id="@+id/txtStartTime" />
      </LinearLayout>
      <LinearLayout
        android:orientation="vertical"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_weight="3">
        <TextView
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr
    /textAppearanceSmall"
          android:text="End time"
          android:id="@+id/lblEnd" />
        <TextView
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr
    /textAppearanceMedium"
          android:text="End time"
          android:id="@+id/txtEndTime" />
      </LinearLayout>
    </LinearLayout>
  </LinearLayout>

  <TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/
    textAppearanceMedium"
    android:textAlignment="center"
    android:text=""
    android:id="@+id/lblTasksUnder" />
  <ListView android:id="@+id/listView"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="7"
    android:listSelector="@android:color/
    transparent"
    android:cacheColorHint="#00000000" />
</LinearLayout>
```

content_task_search_choice.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  xmlns:app="http://schemas.android.com/apk/res
    -auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/
    activity_task_search_choice"
  tools:context=".view.tasks.TaskSearchChoice">

  <TextView
    android:id="@+id/textViewRecordCount"
    android:gravity="center"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Choose what to search:"
    android:padding="1dp" />
```

```
<RadioGroup xmlns:android="http://schemas.
  android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:orientation="vertical">
  <RadioButton android:id="@+id/nameTask"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Name of Task"
    android:onClick="onRadioButtonClicked"/>
  <RadioButton android:id="@+id/categoryTask"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Category"
    android:onClick="onRadioButtonClicked"/>
  <RadioButton android:id="@+id/priorityLevel
  "
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Priority Level"
    android:onClick="onRadioButtonClicked"/>
  <RadioButton android:id="@+id/dateRange"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Date Range"
    android:onClick="onRadioButtonClicked"/>
</RadioGroup>

<EditText
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:id="@+id/taskName"
  android:hint="Task Name"
  android:visibility="gone" />

<Spinner
  android:id="@+id/spinnerCategory"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:background="@android:drawable/
  btn_dropdown"
  android:spinnerMode="dropdown"
  android:visibility="gone" />

<Spinner
  android:id="@+id/spinnerPriority"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:background="@android:drawable/
  btn_dropdown"
  android:spinnerMode="dropdown"
  android:visibility="gone" />

<LinearLayout
  android:layout_height="wrap_content"
  android:layout_width="match_parent"
  android:orientation="horizontal"
  android:id="@+id/linearLayoutDateRange"
  android:visibility="gone" >

  <LinearLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_weight="0.5" >
    <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
  textAppearanceMedium"
      android:text="Time From"
      android:id="@+id/txtDateStart" />

    <EditText
      android:id="@+id/datePickerStart"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="
  center_horizontal"
      android:focusableInTouchMode="false" />
  </LinearLayout>

  <LinearLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_weight="0.5" >
    <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
  textAppearanceMedium"
      android:text="Time End"
      android:id="@+id/txtDateEnd" />

    <EditText
      android:id="@+id/datePickerEnd"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="
  center_horizontal"
      android:focusableInTouchMode="false" />
  </LinearLayout>
</LinearLayout>

<Button
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:text="Next"
  android:id="@+id/nextSearchButton"
  android:layout_gravity="center_horizontal"
  />
</LinearLayout>
```

content_task_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
  "
  xmlns:app="http://schemas.android.com/apk/res
    -auto" android:layout_width="match_parent"
  android:layout_height="match_parent"
    android:paddingLeft="@dimen/
    activity_horizontal_margin"
  android:paddingRight="@dimen/
    activity_horizontal_margin"
  android:paddingTop="@dimen/
    activity_vertical_margin"
  android:paddingBottom="@dimen/
    activity_vertical_margin"
  app:layout_behavior="@string/
    appbar_scrolling_view_behavior"
  tools:showIn="@layout/activity_task_view"
  tools:context=".view.tasks.TaskView">

  <LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignParentTop="true"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    >
    <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
    textAppearanceLarge"
      android:text=""
      android:textAlignment="center"
      android:id="@+id/textViewCurrDate"
      />
  <LinearLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:layout_weight="1"
    android:orientation="horizontal"
    android:weightSum="10" >
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:text="Task Name"
      android:id="@+id/lblTaskName"
      android:layout_weight="9"/>
    <Switch
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Alarm"
      android:id="@+id/alarmSwitch"
      android:layout_gravity="right"
      android:layout_weight="1" />
  </LinearLayout>


    <TextView
      android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/
  textAppearanceMedium"
    android:text=""
    android:id="@+id/txtTaskName"
    android:layout_gravity="center_horizontal
  "
    android:textColor="@color/colorView" />

<LinearLayout
  android:layout_height="wrap_content"
  android:layout_width="match_parent"
  android:layout_weight="1"
  android:orientation="horizontal"
  android:weightSum="10" >
  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/
  textAppearanceMedium"
    android:text="Task Description"
    android:id="@+id/lblTaskDesc"
    android:layout_weight="9"/>
  <Switch
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Motivations"
    android:id="@+id/motivSwitch"
    android:gravity="right"
    android:layout_weight="1" />
</LinearLayout>

  <TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/txtTaskDesc"
    android:textAppearance="?android:attr/
  textAppearanceMedium"
    android:text=""
    android:layout_gravity="center_horizontal
  "
    android:textColor="@color/colorView" />

<LinearLayout
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:layout_weight="1"
    android:orientation="horizontal"
    android:weightSum="10" >
    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
  textAppearanceMedium"
      android:text="Category"
      android:id="@+id/lblCategory"
      android:layout_weight="9"/>
    <Switch
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Add to Calendar"
      android:id="@+id/toCalendarSwitch"
      android:gravity="right"
      android:layout_weight="1" />
  </LinearLayout>

  <TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/txtCategory"
    android:textAppearance="?android:attr/
  textAppearanceMedium"
    android:text=""
    android:layout_gravity="center_horizontal
  "
    android:textColor="@color/colorView" />

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/
  textAppearanceMedium"
    android:text="Priority Level"
    android:id="@+id/lblPriority" />

  <TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/txtPriority"
    android:textAppearance="?android:attr/
  textAppearanceMedium"
  android:text=""
  android:layout_gravity="center_horizontal
"
  android:textColor="@color/colorView" />

<LinearLayout
  android:layout_height="wrap_content"
  android:layout_width="match_parent"
  android:layout_weight="1"
  android:orientation="horizontal" >

  <LinearLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_weight="0.5" >

    <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr
/textAppearanceMedium"
      android:text="Time From"
      android:id="@+id/lblTimeStart" />

    <TextView
      android:id="@+id/txtTimeStart"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="
center_horizontal"
      android:textAppearance="?android:attr
/textAppearanceMedium"
      android:text=""
      android:textColor="@color/colorView"
/>

  </LinearLayout>

  <LinearLayout
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:orientation="vertical"
    android:layout_weight="0.5" >

    <TextView
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr
/textAppearanceMedium"
      android:text="Time End"
      android:id="@+id/lblTimeEnd" />

    <TextView
      android:id="@+id/txtTimeEnd"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="
center_horizontal"
      android:textAppearance="?android:attr
/textAppearanceMedium"
      android:text=""
      android:textColor="@color/colorView"
/>

  </LinearLayout>

</LinearLayout>

<TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textAppearance="?android:attr/
textAppearanceMedium"
  android:text="Repeat"
  android:id="@+id/lblRepeat" />

<TextView
  android:id="@+id/txtRepeat"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:textAppearance="?android:attr/
textAppearanceMedium"
  android:text=""
  android:layout_gravity="center_horizontal
"
  android:textColor="@color/colorView" />

<TextView
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
```

```
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:text="Start Date"
      android:id="@+id/lblFromDate" />

    <TextView
      android:id="@+id/txtDateStart"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="center_horizontal
    "
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:text=""
      android:textColor="@color/colorView" />

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:text="End Date"
      android:id="@+id/lblToDate" />

    <TextView
      android:id="@+id/txtDateEnd"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="center_horizontal
    "
      android:textAppearance="?android:attr/
    textAppearanceMedium"
      android:text=""
      android:textColor="@color/colorView" />

  <LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:weightSum="2">

    <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Start Task"
      android:id="@+id/startActual"
      android:layout_gravity="center_horizontal
    "
      android:layout_weight="1"/>

    <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="End Task"
      android:id="@+id/endActual"
      android:layout_gravity="center_horizontal
    "
      android:layout_weight="1" />

  </LinearLayout>

    <LinearLayout
      android:orientation="vertical"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_gravity="center_horizontal
    ">

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text="Statistic for this day"
        android:id="@+id/header"/>

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text="Estimated Start"
        android:id="@+id/lblEstStart"/>

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text=" "
        android:id="@+id/estStart"
        android:textColor="@color/colorView" />

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text="Actual Start"
        android:id="@+id/lblActStart"/>

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text=" "
        android:id="@+id/actStart"
        android:textColor="@color/colorView" />

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text="Estimated End"
        android:id="@+id/lblEstEnd"/>

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text=" "
        android:id="@+id/estEnd"
        android:textColor="@color/colorView" />

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text="Actual End "
        android:id="@+id/lblActEnd"/>

      <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:textAppearance="?android:attr/
    textAppearanceMedium"
        android:text=" "
        android:id="@+id/actEnd"
        android:textColor="@color/colorView" />

    </LinearLayout>
  </LinearLayout>

</ScrollView>
```

display_category_input_form.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools
    "
  android:layout_width="match_parent"
  android:layout_height="match_parent" >

  <EditText
    android:id="@+id/editTextCategoryName"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:hint="Category Name"
    android:singleLine="true" >
    <requestFocus />
  </EditText>
</RelativeLayout>
```

display_motivation_input_form.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
```

```
  xmlns:tools="http://schemas.android.com/tools
    "
  android:layout_width="match_parent"
  android:layout_height="match_parent" >

  <EditText
    android:id="@+id/editTextMotivation"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:hint="Motivation Name"
    android:singleLine="true" >
    <requestFocus />
  </EditText>
  <Spinner
    android:layout_below="@id/
    editTextMotivation"
    android:id="@+id/spinnerCategory"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@android:drawable/
    btn_dropdown"
    android:spinnerMode="dropdown" />
</RelativeLayout>
```

display_search.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!--
/*
** Copyright 2010, The Android Open Source
    Project
**
** Licensed under the Apache License, Version
    2.0 (the "License");
** you may not use this file except in
    compliance with the License.
** You may obtain a copy of the License at
**
**    http://www.apache.org/licenses/LICENSE-2.0
**
** Unless required by applicable law or agreed
    to in writing, software
** distributed under the License is distributed
    on an "AS IS" BASIS,
** WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND
    , either express or implied.
** See the License for the specific language
    governing permissions and
** limitations under the License.
*/
-->
<!-- Layout for list items in the search
    results.
 -->
<LinearLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="5dp">
  <TextView
    android:id="@+id/task_searched"
    style="@android:style/TextAppearance.Large"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    />
  <TextView
    android:id="@+id/task_definition"
    style="@android:style/TextAppearance.Small"
    android:singleLine="true"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" />
</LinearLayout>
```

display_stat.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="10dip" >

  <TextView
    android:id="@+id/month_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentRight="true"
    android:layout_alignParentTop="true"
    android:paddingTop="6sp"
    android:textColor="#777777"
    android:textSize="9sp" />

  <TextView
    android:id="@+id/year_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentRight="true"
    android:layout_below="@id/month_tv"
    android:textColor="#777777"
    android:textSize="9sp" />

  <TextView
    android:id="@+id/date_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_toLeftOf="@id/year_tv"
    android:paddingRight="1sp"
    android:textColor="#777777"
    android:textSize="27sp" />

  <TextView
    android:id="@+id/msg_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_toLeftOf="@id/date_tv"
    android:textColor="#587498"
    android:textSize="18sp" />

  <TextView
    android:id="@+id/time_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_below="@id/msg_tv"
    android:paddingTop="5sp"
    android:textColor="#777777"
    android:textSize="12sp" />

  <TextView
    android:id="@+id/priority_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toRightOf="@id/time_tv"
    android:layout_below="@id/msg_tv"
    android:paddingTop="5sp"
    android:paddingLeft="10sp"
    android:textColor="#777777"
    android:textSize="12sp" />

  <TextView
    android:id="@+id/est_start_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_below="@id/time_tv"
    android:paddingTop="5sp"
    android:paddingLeft="10sp"
    android:textColor="#777777"
    android:textSize="12sp" />

  <TextView
    android:id="@+id/act_start_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toRightOf="@id/est_start_tv"
    android:layout_below="@id/priority_tv"
    android:paddingTop="5sp"
    android:paddingLeft="10sp"
    android:textColor="#777777"
    android:textSize="12sp" />

  <TextView
    android:id="@+id/est_end_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_below="@id/est_start_tv"
    android:paddingTop="5sp"
    android:paddingLeft="10sp"
    android:textColor="#777777"
    android:textSize="12sp" />
  <TextView
    android:id="@+id/act_end_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toRightOf="@id/est_end_tv"
    android:layout_below="@id/act_start_tv"
    android:paddingTop="5sp"
    android:paddingLeft="10sp"
    android:textColor="#777777"
    android:textSize="12sp" />

  <TextView
    android:id="@+id/time_diff_start_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toRightOf="@id/act_start_tv"
    android:layout_below="@id/priority_tv"
    android:paddingTop="5sp"
    android:paddingLeft="10sp"
    android:textColor="#777777"
    android:textSize="12sp" />

  <TextView
    android:id="@+id/time_diff_end_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toRightOf="@id/act_end_tv"
    android:layout_below="@id/
    time_diff_start_tv"
    android:paddingTop="5sp"
    android:paddingLeft="10sp"
    android:textColor="#777777"
    android:textSize="12sp" />

</RelativeLayout>
```

display_taskmain.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.
    android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:padding="10dip" >

  <TextView
    android:id="@+id/month_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentRight="true"
    android:layout_alignParentTop="true"
    android:paddingTop="6sp"
    android:textColor="#777777"
    android:textSize="9sp" />

  <TextView
    android:id="@+id/year_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentRight="true"
    android:layout_below="@id/month_tv"
    android:textColor="#777777"
    android:textSize="9sp" />

  <TextView
    android:id="@+id/date_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_toLeftOf="@id/year_tv"
    android:paddingRight="1sp"
    android:textColor="#777777"
    android:textSize="27sp" />

  <TextView
    android:id="@+id/msg_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_toLeftOf="@id/date_tv"
    android:textColor="#587498"
    android:textSize="18sp" />

  <TextView
    android:id="@+id/time_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_below="@id/msg_tv"
    android:paddingTop="5sp"
    android:textColor="#777777"
    android:textSize="12sp" />

  <TextView
    android:id="@+id/priority_tv"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toRightOf="@id/time_tv"
    android:layout_below="@id/msg_tv"
    android:paddingTop="5sp"
    android:paddingLeft="10sp"
    android:textColor="#777777"
    android:textSize="12sp" />

</RelativeLayout>
```

XI. Acknowledgement

At last. Acknowledgements. Woooo! As I am typing this, it is hard for me to think of the words to put in here. There are too many things that I want to say but just cannot find the words for it. And you guys know that I'm not really good with words. But anyway, here it goes:

Thank you. Thank you. Thank you.

First of all, thank you God, for your guidance and wisdom that You have provided. Especially during the times where I almost gave up, You gave me hope.

To my family, friends, mentors, and others who believed in me. To those who kept reminding and pushing me to strive and not give up.

I would like to thank my parents, Ruben and Ren, and siblings, Ken, Zeek, and Shae, for their support and understanding, especially during the dark times when there are too many academic-related things that I need to do.

To my adviser, Ma'am Sheila Magboo, thank you for your support and advice. Sorry ma'am for the late consultations and other troubles that I may have caused you. Thank you ma'am!

To all my professors from first year to fourth year, thank you for the knowledge that you provided to us. May we able to use this knowledge to our advantage in our lives.

To Ate Eden, thank you for always entertaining us when we are always asking questions, even if we are starting to be annoying.

To my good friends in ComSci Batch 2012 who I shared my struggles with during the challenging times of our college years, thank you for being there.

To Hotties, thank you very much guys for every moment that we spent with each other. Thank you for being the "support" group through the hell weeks and life problems that we all faced. Thank you also for the times and memories that we shared. Please know that I will treasure those things. Shoutout to the Hotties peeps: Alee, Eva, Krizia, Bella, Christine, Cristine, Rachelle, Rain, Earl, Marvin, Joseph, Ruah, and Mark. Hi guys!!! Again, thank you for everything. I love you guys! :)

To my bestfriend, my ”uwian” buddy, my partner on most projects ever since CS11, I would like to say hi. Haha. Hi Reena. Thank you for being my bestfriend that I can rely to. Thank you for being there to listen to my rants and my stories, even though I think I'm starting to be annoying. Thank you for sticking with me through thick and thin. Again thank you for everything. I love you ;)

I would like to give thanks to my friends Rae, for the ”motherly” advice and sermons, and Angel, for the corny jokes and ”hugot” stories. The things that you both shared really helped during the times that I'm doing my SP. Haha thank you! :)

To my laptop, thank you for not giving up on me even if you are running 24/7 already.

Thank you also for those brands of coffees, that I will not advertise since I'm not paid to do so, for keeping me awake during those needed times.

Again, thank you to all of you. I could not have done everything without all of you.

That is all. #laban #puso