Software Requirements Specification

for

Task App

Version 6.0 approved

Prepared by <Daniel, Melisa & Ryan>

<team juiced>

<12/17/2020>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

Screen1	11/17	Added Image buttons	1
Screen1.1	11/19	implemented SQL	2
finalish	11/30	enabled SQL database to show information on Schedule page and Active Task page	3
stable2	11/30	Fixed a bug where entering the Active Task Screen crashed the app.	4
tap to dell	12/4	Added functionality to delete tasks when clicked on in the Active Task screen	5
tap to dell v1.1	12/13	Removed the Progress imageButton and fine tuned some Strings to be more presentable.	6

1. Introduction

1.1 Purpose

The purpose of this document is to explain the Task App and how it can improve lifes with its task management tools. It'll list all the tools the app holds and how it'll interact with external systems such as databases. The Task App version(3.0) requires an android device in order to use Task App.

1.2 Document Conventions

The SRS was written with an Arial font at size 11.

1.3 Intended Audience and Reading Suggestions

The intended audience of SRS is team members in order to guide them to the product's ultimate vision and goal. The document can also be read by users of the app to understand the app's features and intentions more. Marketing staff can use the document to lay out the features for advertisements to make the app look appealing to anyone with a phone. Readers of the document should start at the introduction

1.4 Product Scope

The purpose of the Task App is to provide users with a mobile task manager that is easy to use and well organized. We plan to use a cloud database to store our users login information to ease the access of their information across other android devices. We will provide a streamlined experience for the users to have their days fully planned at the touch of their fingerprints.
The Task App will be able to handle all the tasks the user would like to add. It will then organize the tasks with the priority feature and the deadline of the task. Users will be able to view tasks in a calendar view and push tasks into the out of sight page in order to free up space in the user's days.

1.5 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

Websites that helped in the creation of this SRS document include

- https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database#:~:text=A%20Software%20Requirements%20Specification%20

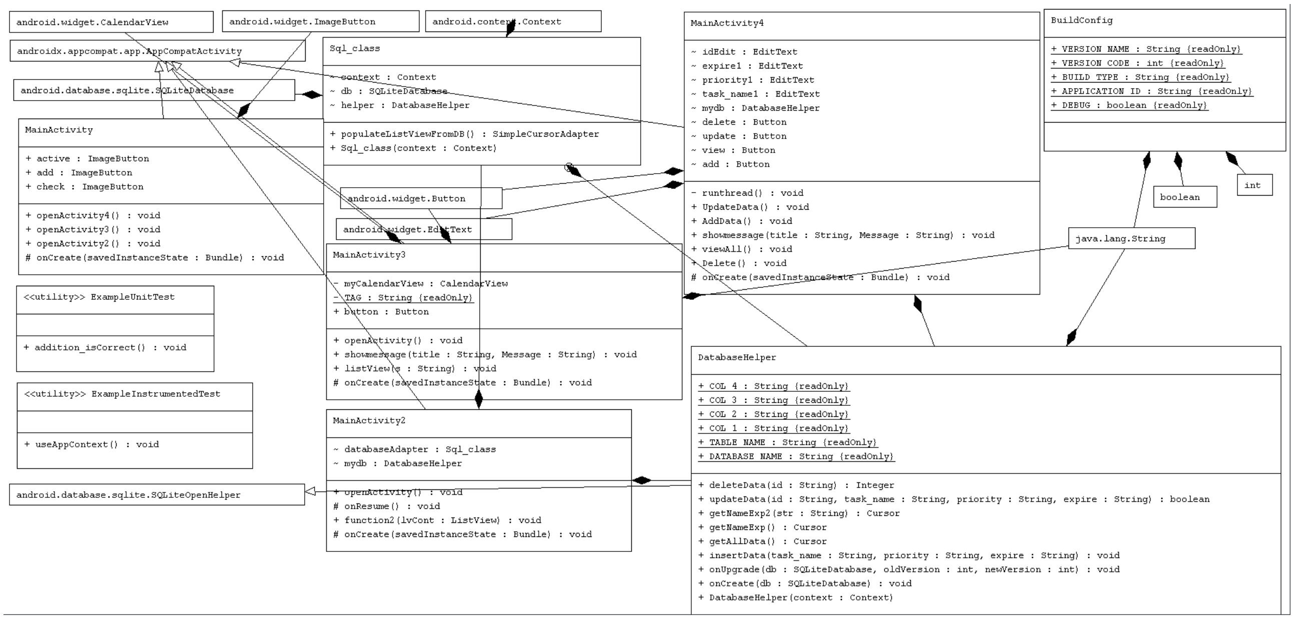
- http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs_example_2010_group2.pdf
- https://www.reqview.com/doc/iso-iec-ieee-29148-srs-example

2. Overall Description

2.1 Product Perspective

The Task App is a replacement for the physical planner and the clunky reminder apps that exist.

2.2 Product Functions


android.widget.CalendarView
android.widget.ImageButton
android.content.Context
androidx.appcompat.app.AppCompatActivity
android.database.sqlite.SQLiteDatabase
Sql_class
~ context : Context
~ db : SQLiteDatabase
~ helper : DatabaseHelper
+ populateListViewFromDB() : SimpleCursorAdapter
+ Sql_class(context : Context)
MainActivity
+ active : ImageButton
+ add : ImageButton
+ check : ImageButton
+ openActivity4() : void
+ openActivity3() : void
+ openActivity2() : void
# onCreate(savedInstanceState : Bundle) : void
android.widget.Button
android.widget.EditText
MainActivity4
~ idEdit : EditText
~ expire1 : EditText
~ priority1 : EditText
~ task_name1 : EditText
~ mydb : DatabaseHelper
~ delete : Button
~ update : Button
~ view : Button
~ add : Button
- runthread() : void
+ UpdateData() : void
+ AddData() : void
+ showmessage(title : String, Message : String) : void
+ viewAll() : void
+ Delete() : void
# onCreate(savedInstanceState : Bundle) : void
BuildConfig
+ VERSION NAME : String {readOnly}
+ VERSION CODE : int {readOnly}
+ BUILD TYPE : String {readOnly}
+ APPLICATION ID : String {readOnly}
+ DEBUG : boolean {readOnly}
boolean
int
java.lang.String
<<utility>> ExampleUnitTest
+ addition_isCorrect() : void
<<utility>> ExampleInstrumentedTest
+ useAppContext() : void
MainActivity3
- myCalendarView : CalendarView
- TAG : String {readOnly}
+ button : Button
+ openActivity() : void
+ showmessage(title : String, Message : String) : void
+ listView(s : String) : void
# onCreate(savedInstanceState : Bundle) : void
MainActivity2
~ databaseAdapter : Sql_class
~ mydb : DatabaseHelper
+ openActivity() : void
# onResume() : void
+ function2(lvCont : ListView) : void
# onCreate(savedInstanceState : Bundle) : void
android.database.sqlite.SQLiteOpenHelper
DatabaseHelper
+ COL 4 : String {readOnly}
+ COL 3 : String {readOnly}
+ COL 2 : String {readOnly}
+ COL 1 : String {readOnly}
+ TABLE NAME : String {readOnly}
+ DATABASE NAME : String {readOnly}
+ deleteData(id : String) : Integer
+ updateData(id : String, task_name : String, priority : String, expire : String) : boolean
+ getNameExp2(str : String) : Cursor
+ getNameExp() : Cursor
+ getAllData() : Cursor
+ insertData(task_name : String, priority : String, expire : String) : void
+ onUpgrade(db : SQLiteDatabase, oldVersion : int, newVersion : int) : void
+ onCreate(db : SQLiteDatabase) : void
+ DatabaseHelper(context : Context)

The product must perform well as a task organizer for a person's day to day through its different pages the user is allowed to access such as:

- Add new task
- Enter task details
- View Daily Tasks
- Edit previously made tasks
- View a calendar view of the tasks

2.3 User Classes and Characteristics

Users of the Task App should be able to edit and view their tasks throughout the day. The app will take input from the user involving the name of the task, date, and priority. Once completed the task will join the others in the task list. (Explain the types of classes how what access what)

The user will be able to use the following functions:

- Add new task
    - Name
    - Date
    - Priority
- Edit existing tasks
- View tasks in calendar view
- View tasks in list view

The number of tasks that can be added is unlimited, but the history will disappear over time.

2.4 Operating Environment

The hardware platform required for Task App is an android device running on Android Studio 4.1.1. The Task App must be connected to the internet in order to access the user’s account as well as back up their information to a database. The Task App will work with an online database to store and back up users information.

2.5 Design and Implementation Constraints

The amount of time allotted to the Task App team will impede the team as they have to work within the constraints of about half a semester as well as working while working on other class’s work. The use of a database will raise security concerns as well as concerns of how reliable the database chosen is.

2.6 User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

The Task App comes with no help manual, but instead will flow similar to other apps that many are familiar with. The Task App has labeled buttons that will make the app easy to navigate creating and viewing tasks. The buttons follow a standard app layout that will make it easy for users' fingers to reach and use the standard keyboard to enter information that users are familiar with.

2.7 Assumptions and Dependencies

We’re assuming that a 3rd party plugin within Android Studio would be necessary to save app data locally, such as SQLite. We also assumed that online cloud options would work as well such as API services, setting up an SQL server that runs online, or an online instance service database such as Google Firebase or Microsoft Azure or MongoDB. If any of the latter options were to be chosen, then our project would have external dependencies.

3. External Interface Requirements

3.1 User Interfaces

When first opening the app users will be met with the home page, see Figure 1. The user will have three options on this screen which are add task, active task view, and check schedule. The progress option on the prototype not being available at this time due to budget constraints.

The add task button should be the first option first time users choose in order to start their task lists. In Figure 2, the add task screen can be seen. Here the user can add details about their tasks including name, priority, and a date for it to expire. Lastly, they will add the task to their task list.

After adding tasks the user will have the two different views of their tasks. First is the task list view which is a traditional view of the tasks in order of expiration date, see Figure 3. From here the tasks can be edited or deleted to the users preferences. Their other option is to view their tasks in calendar view, see Figure 4. Calendar view will show an entire month and show the tasks on the days they expire.

Sample Screens (Prototype)

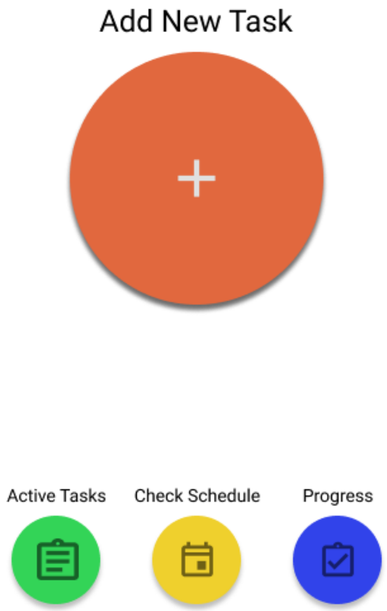


Figure 1 - Home/add screen

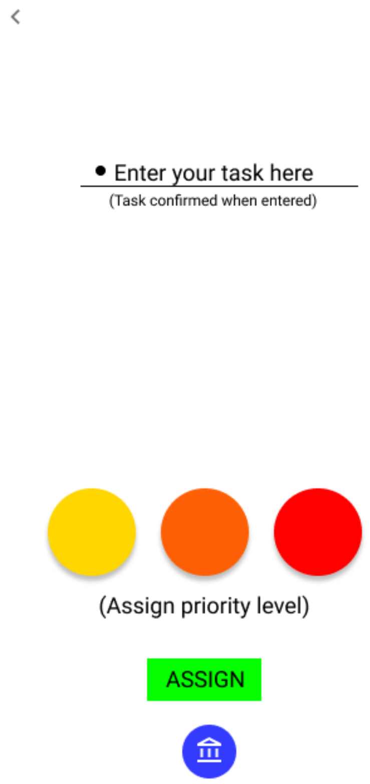


Figure 2 - Enter task information

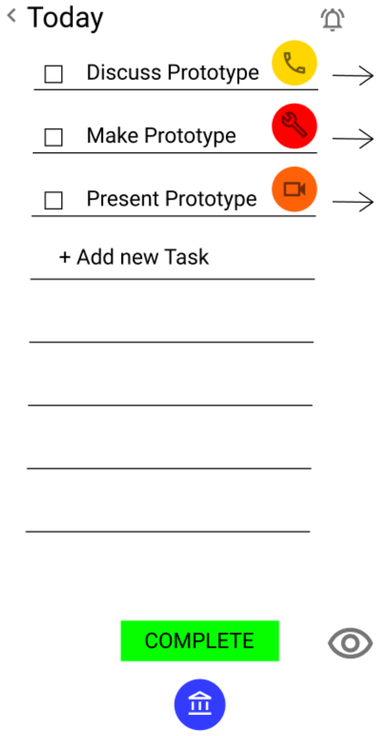


Figure 3 - Task list

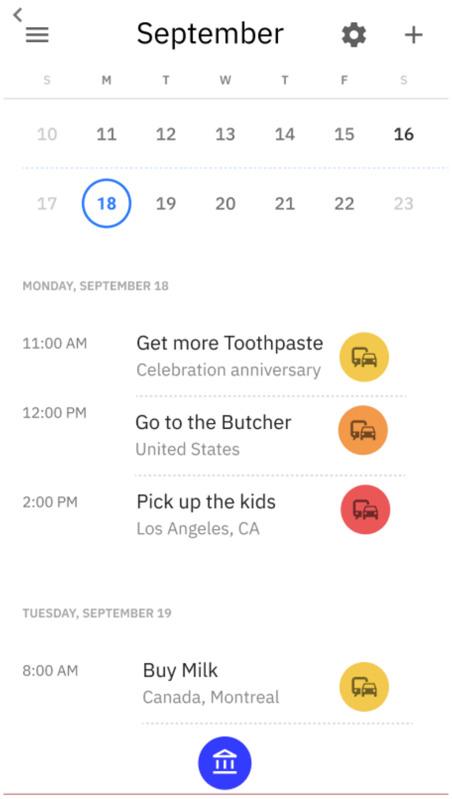


Figure 4 - Calendar view

The Task App will follow a circular theme throughout its pages to look sleek and modern. It will take advantage of the back button included on most androids that's located near the bottom left, but will also support its own dedicated buttons to go back pages.

3.2 Hardware Interfaces

- The Task App is a mobile application that has few hardware interfaces as it is mostly self contained
  - Windows if the app is emulated
  - Android phone OS when downloaded onto an android device

3.3 Software Interfaces

Software used	Description
Operating system	We chose to use the Android OS since its best for our mobile audience
Database	To save the tasks the user creates we chose to use SQL

Android Studio	Used to code the app

3.4 Communications Interfaces

The Task App is mostly self contained since it does not communicate with outside systems. The communication is done within the app itself and the phone it is on.

4. System Features

4.1 Dashboard

4.1.1 Add Task Icon

This icon redirects the user to the Add Task page. Here the user can create new tasks.

4.1.2 Active Tasks Icon

This icon redirects the user to the Active Tasks page where the user can see the tasks that need to be completed that day.

4.1.3 Check Schedule Icon

This icon will allow the user to see their upcoming tasks in a calendar view.

4.2 Add Task Feature

4.2.1 Description of the Task

At the time of the creation of any new task the user is required to add the name of the task they are creating. There will be an infinite amount of characters available, so they have the freedom to enter as much or as little detail as they choose.

4.2.2 Assigning Priority

When adding a new task, the user has the ability to assign a priority level to the task. The priority levels can be selected by choosing one of the three icons available at the bottom of the page. These icons include yellow (low importance), orange (medium importance), and red (high importance).

4.2.3 Date and Location

Besides adding the description and priority level of the task, the user will also be required to add a date and location for the completion of the task. The date will be used to determine when the task needs to be completed, therefore reminding the user the task needs to be completed the day it was assigned to. The location will be used to *remind the user of where the task needs to be completed, this feature will be used as a reference to make the user's experience with this app a satisfactory experience.*

○ 4.3 Active Task Feature

4.3.1 Calendar View

The user can view a calendar, in which they can view what tasks are due on specific dates. This feature allows the user to see upcoming tasks in a calendar or a weekly view.

4.3.2 Add New Task

The Add New Task feature in the part of the app, when selected will cause the user to be redirected to the page where new tasks are created.

4.3.3 Mark Task as Complete

When a task is complete, the user can select the box next to the task and it will be marked as complete once the "Complete" button is selected. Once checked off as complete the task will be removed and will be able to be seen in the Progress tab.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- Functional requirements
  - System must be able to accept input from the user
  - System must be able to enter input from the user into the database
  - System must be able to retrieve and display items within the database
  - System must be able to delete items from the database and update accordingly
  - System must be able to cycle through a calendar system and display items from the database
- Nonfunctional requirements.
  - System shall be using internal memory, rather than obtaining data from the internet.
  - System shall be usable to the extent that the app is opened and shall not close unless it crashes or too much memory is being used for the phone and will close automatically.
  - System shall be able to input as much information as the user wants until it exceeds the capacity of the device's memory.

5.2 Safety Requirements

If the database were to experience damage, such as a data breach, then there would be a possibility of the tasks of the users being released to the public or untrusted partners.

5.3 Security Requirements

The data the app uses must be held in a database in order to function. However, holding data in a database will pose security risks since the database has the possibility of being breached.

5.4 Software Quality Attributes

ADAPTABILITY: The app should be able to work on a number of different Android devices with the same usability of each of them
AVAILABILITY: The tasks should be available to view and create any time of day
CORRECTNESS: The task should correctly display the name and date in order for the task to be possible to complete
FLEXIBILITY: The app should be able to handle a varied array of tasks as well as a many tasks as the user requires
INTEROPERABILITY: The task should take the information about the tasks and display them correctly to the list as well as to the calendar section
MAINTAINABILITY: The app should be able conutnine to function without much need for updates
PORTABILITY: The app should feel more portable than a traditional planner and be easy to check throughout the day
RELIABILITY: The app should be able to be accessed at any point of the day and should always display the correct information
ROBUSTNESS: The app should be able to handle as many tasks as the user creates without errors
USABILITY: The app should feel easy to use and be able to handle any kind of task the user desires to enter

5.5 Business Rules

The task apps team relied on everyone to do the job they were assigned while still helping others on the team when they needed it.

6. Other Requirements

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Term	**Definition**
OS	Operating system
SQL	Used to communicate with database

Database	An organized collection of logically collected data

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

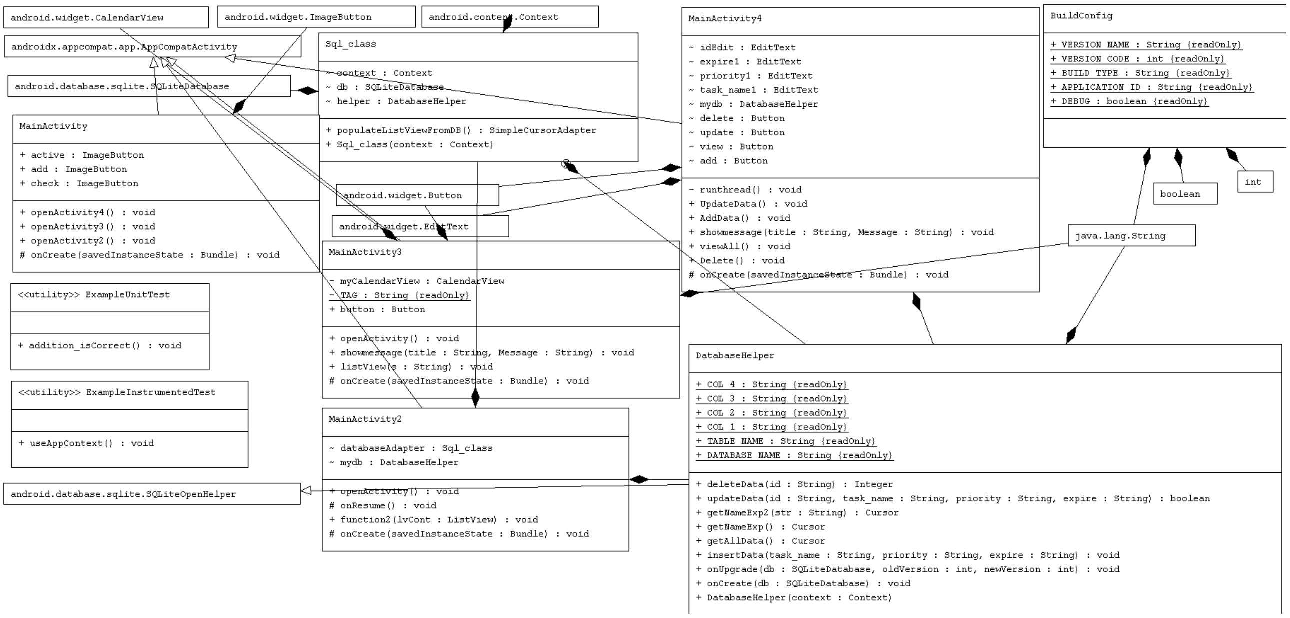

android.widget.CalendarView
android.widget.ImageButton
android.content.Context
androidx.appcompat.app.AppCompatActivity
android.database.sqlite.SQLiteDatabase
Sql_class
~ context : Context
~ db : SQLiteDatabase
~ helper : DatabaseHelper
+ populateListViewFromDB() : SimpleCursorAdapter
+ Sql_class(context : Context)
MainActivity
+ active : ImageButton
+ add : ImageButton
+ check : ImageButton
+ openActivity4() : void
+ openActivity3() : void
+ openActivity2() : void
# onCreate(savedInstanceState : Bundle) : void
android.widget.Button
android.widget.EditText
MainActivity3
- myCalendarView : CalendarView
- TAG : String {readOnly}
+ button : Button
+ openActivity() : void
+ showmessage(title : String, Message : String) : void
+ listView(s : String) : void
# onCreate(savedInstanceState : Bundle) : void
<<utility>> ExampleUnitTest
+ addition_isCorrect() : void
<<utility>> ExampleInstrumentedTest
+ useAppContext() : void
MainActivity2
~ databaseAdapter : Sql_class
~ mydb : DatabaseHelper
+ openActivity() : void
# onResume() : void
+ function2(lvCont : ListView) : void
# onCreate(savedInstanceState : Bundle) : void
android.database.sqlite.SQLiteOpenHelper
MainActivity4
~ idEdit : EditText
~ expire1 : EditText
~ priority1 : EditText
~ task_name1 : EditText
~ mydb : DatabaseHelper
~ delete : Button
~ update : Button
~ view : Button
~ add : Button
- runthread() : void
+ UpdateData() : void
+ AddData() : void
+ showmessage(title : String, Message : String) : void
+ viewAll() : void
+ Delete() : void
# onCreate(savedInstanceState : Bundle) : void
BuildConfig
+ VERSION NAME : String {readOnly}
+ VERSION CODE : int {readOnly}
+ BUILD TYPE : String {readOnly}
+ APPLICATION ID : String {readOnly}
+ DEBUG : boolean {readOnly}
boolean
int
java.lang.String
DatabaseHelper
+ COL 4 : String {readOnly}
+ COL 3 : String {readOnly}
+ COL 2 : String {readOnly}
+ COL 1 : String {readOnly}
+ TABLE NAME : String {readOnly}
+ DATABASE NAME : String {readOnly}
+ deleteData(id : String) : Integer
+ updateData(id : String, task_name : String, priority : String, expire : String) : boolean
+ getNameExp2(str : String) : Cursor
+ getNameExp() : Cursor
+ getAllData() : Cursor
+ insertData(task_name : String, priority : String, expire : String) : void
+ onUpgrade(db : SQLiteDatabase, oldVersion : int, newVersion : int) : void
+ onCreate(db : SQLiteDatabase) : void
+ DatabaseHelper(context : Context)

Figure 5: Project UML Diagram

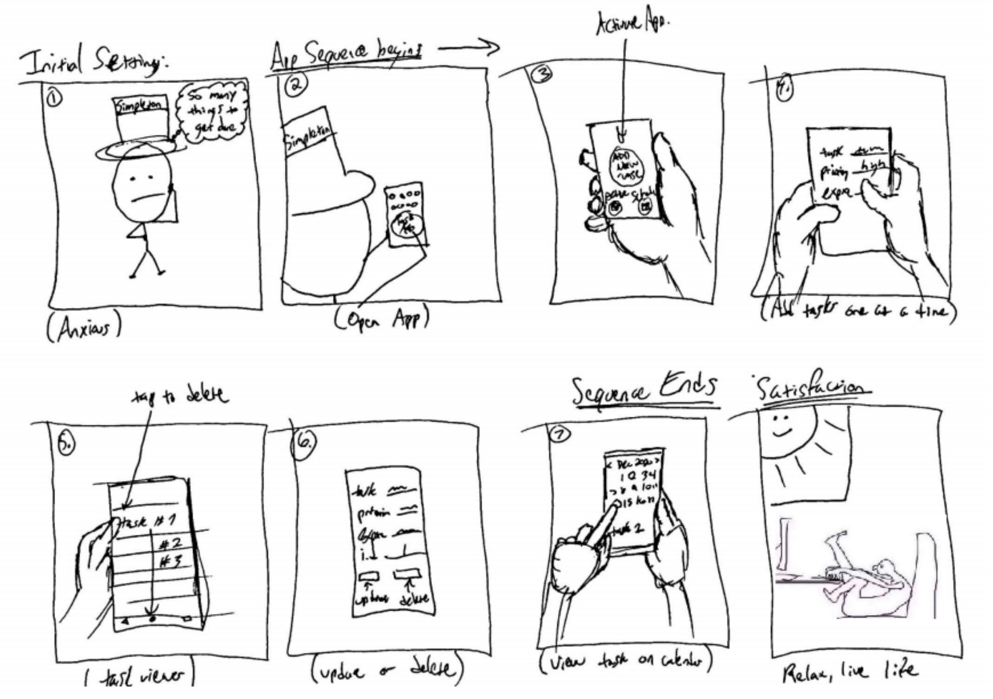


Figure 6: User story containing user role, goal and acceptance criteria

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>