DESIGN AND PLANNING DOCUMENT

Submitted By:

Aj Tarczy , BannerId: B00691379
Kshitij Paithankar ,BannerId: B00800573
Suman Singh BanerId: B00727915
Aadesh Shah, BannerId: B00802629
Awotunde Ayodeji, BannerId: B00766409

Table of Contents:

INTRODUCTION

The Proposed application brings TO-DO Lists, Calendars, and Group T-DO Lists into one app. On their personal TO-DO List, the user can create, add, edit, and delete tasks. If the user is a part of a group, they can also create a Group TO-DO List, and be able to have a shared TO-DO List for their group. The Calendar allows the user to schedule and manage events, so that they can better manage their non-immediate events. These parts of the application together serve the purpose of assisting in the organization of the users daily and longer-period tasks. This app would be especially useful to any individual person, business, or organization looking for a tool to help manage personal or group events.

This design document presents the designs used or intended to be used in implementing this project.

Purpose

The purpose of this document is to present a detailed description of the designs of the GetItDone application. This document is intended to be used as the design guidelines to implement the project.

SCOPE

This document gives a detailed description of the software architecture of the GetItDone Application. It specifies the structure and design of every part of the application. It discusses mockups, wireframes and displays some of the use cases that had transformed into sequential and activity diagrams. The class diagrams show how the programming team would implement the specific module.

OVERVIEW

This document contains discussions of the designs for the project with diagrams, samples of the UI in the application, sequence diagrams, what has been done so far, what we have left to do, and a list of all the tools, environments, and libraries used in the project.

2 .DESIGN ARCHITECTURE AND STORYBOARDING

Wireframes:

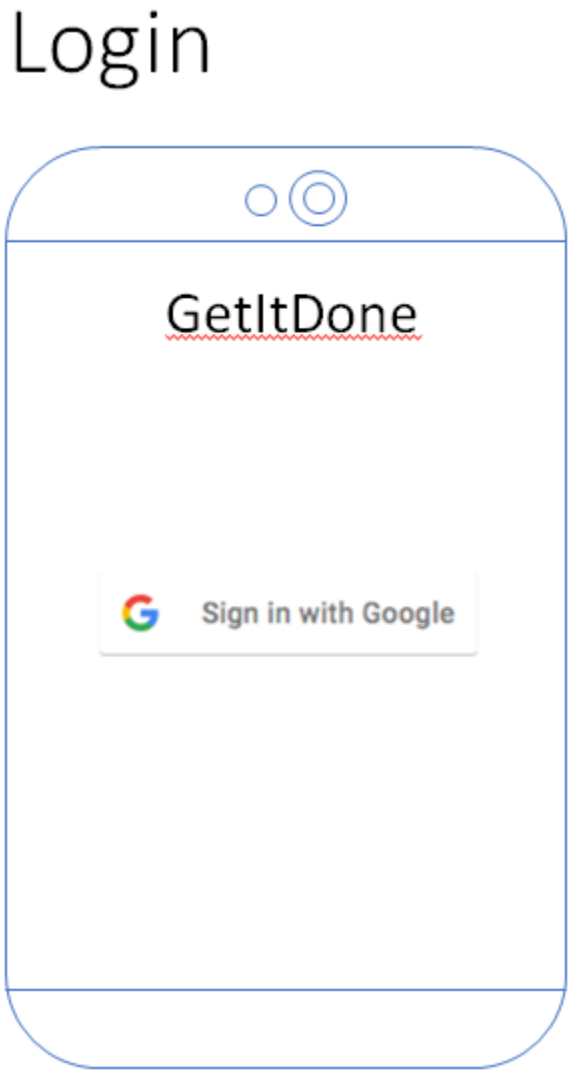


Figure 1: Login Page

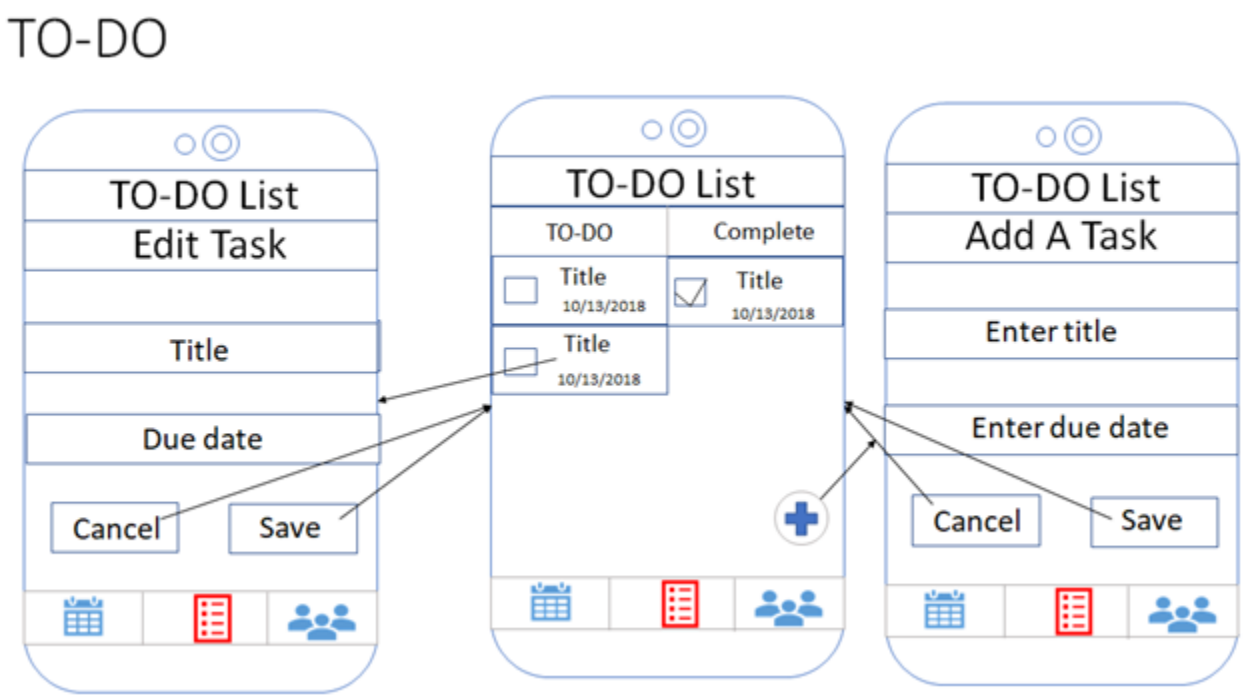


Figure 2: TO-DO List

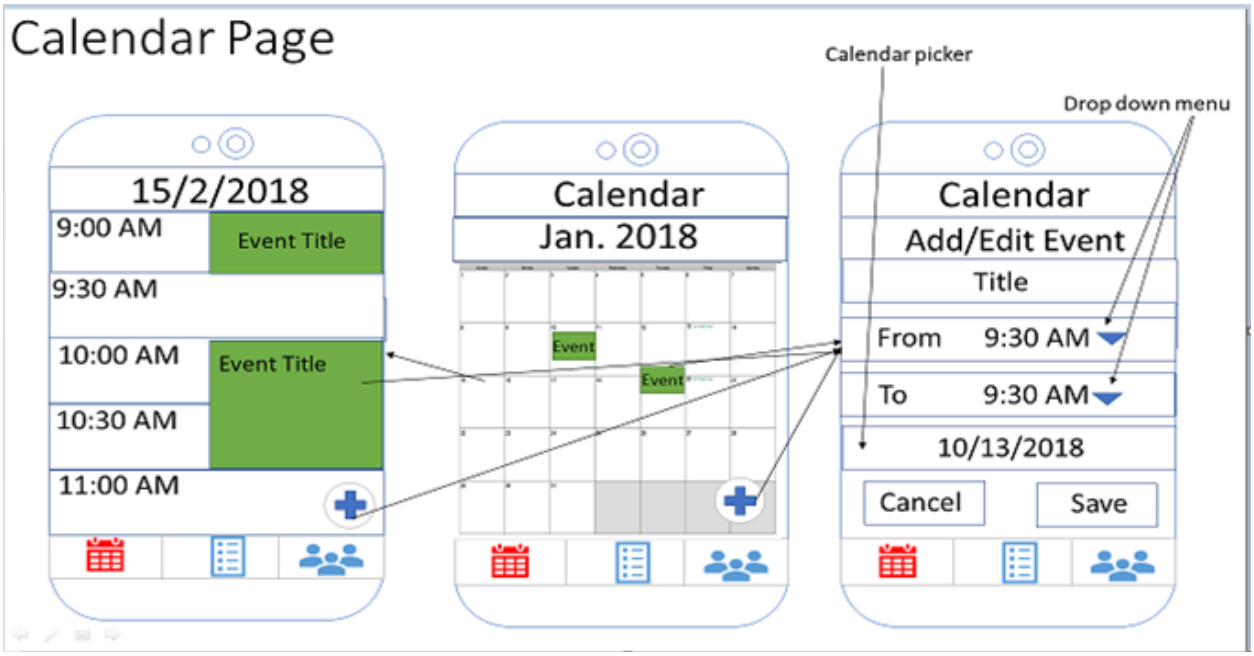


Figure 3: Calendar

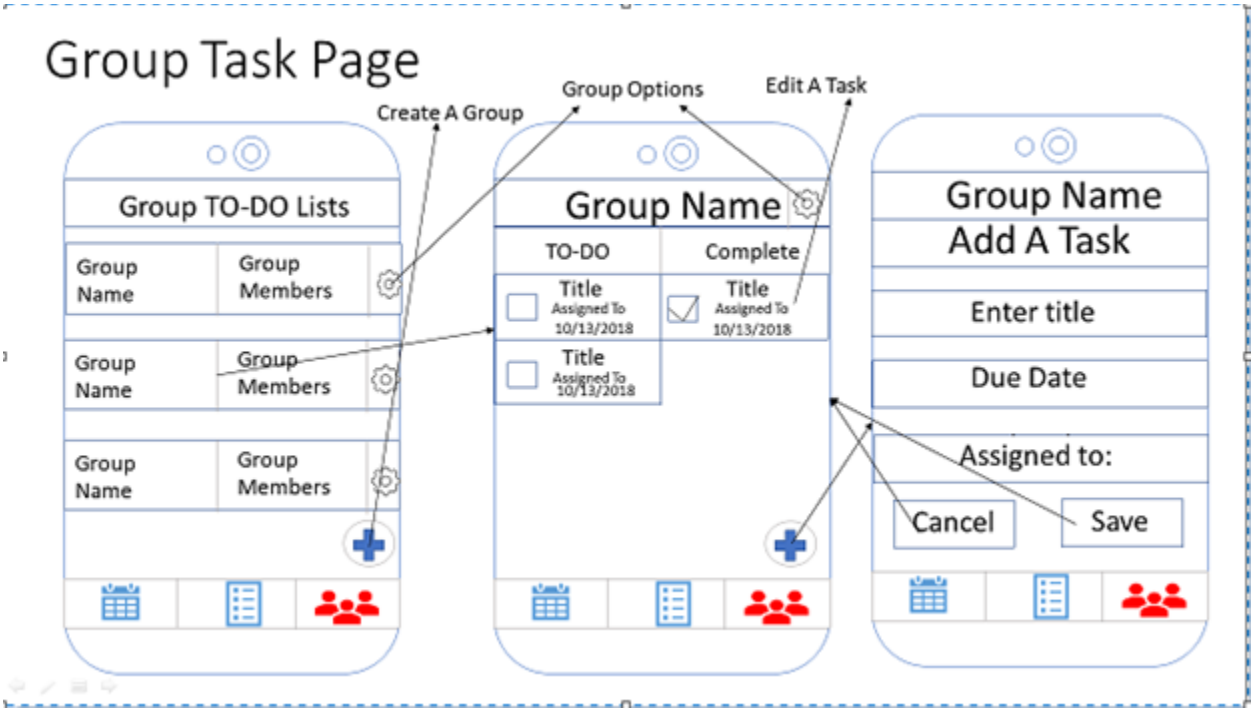


Figure 4: Group TO-DO List

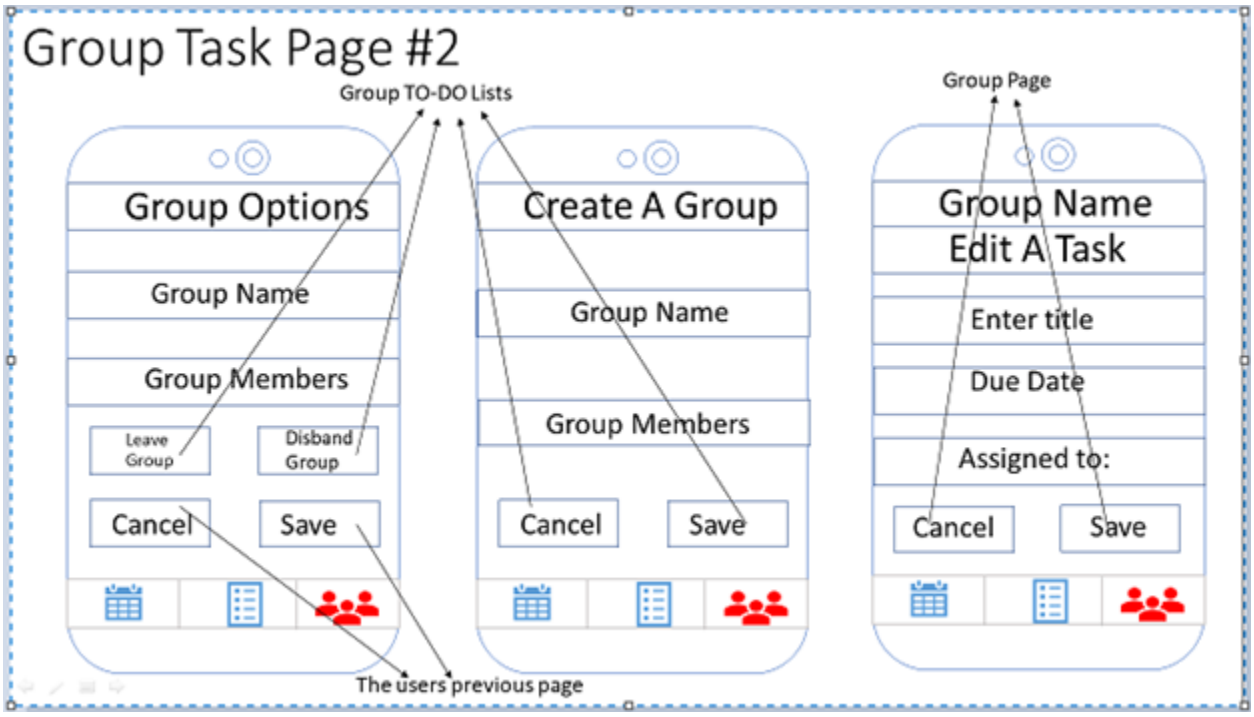


Figure 5: Group Task

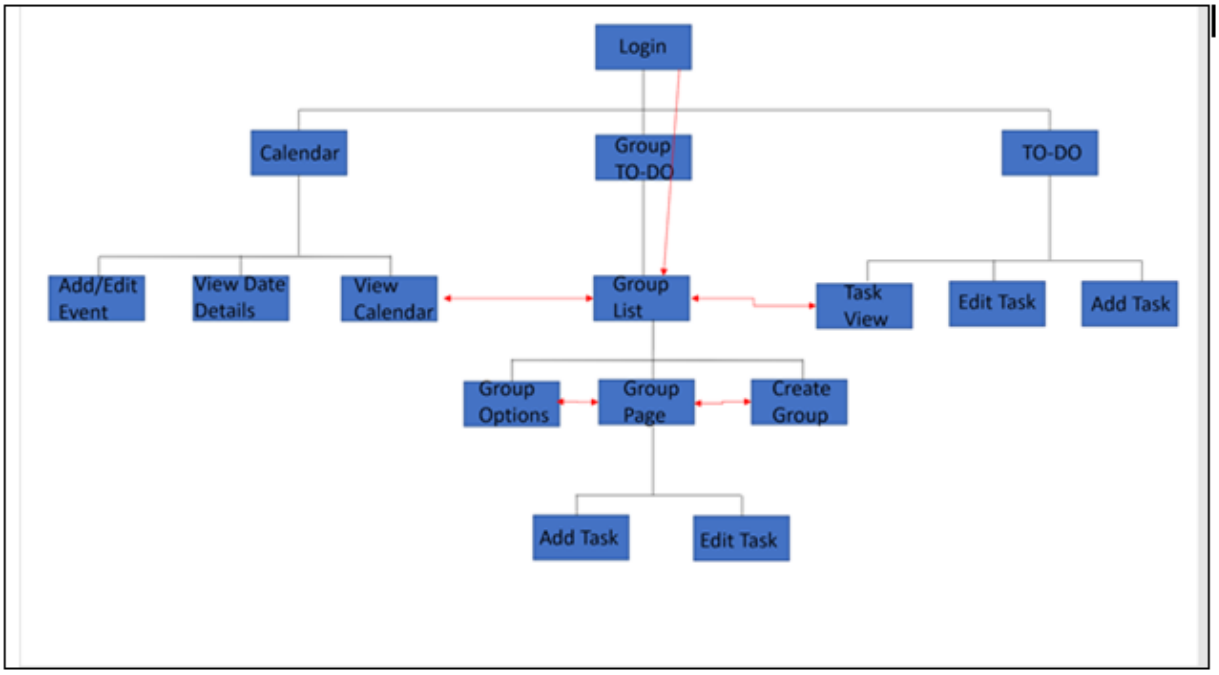


Figure 6: Clickstream

PROTOTYPE

A prototype is an early sample, model, or release of a product built to test a concept or process or to act as a thing to be replicated or learned from.

The prototypes for each of the 3 main sections of the application are shown below.

PERSONAL TO-DO

The Personal TO-DO Page contains a List View where each TO-DO item is appended using the Add activity opened by clicking on the Add button (Figure 7). It asks the user to enter the title and due date for the TO-DO. On saving, it appends the item onto the list (Figure 8).

The item in the TO-DO list can be deleted by swiping left on each item and then clicking on the delete button (Figure 9).

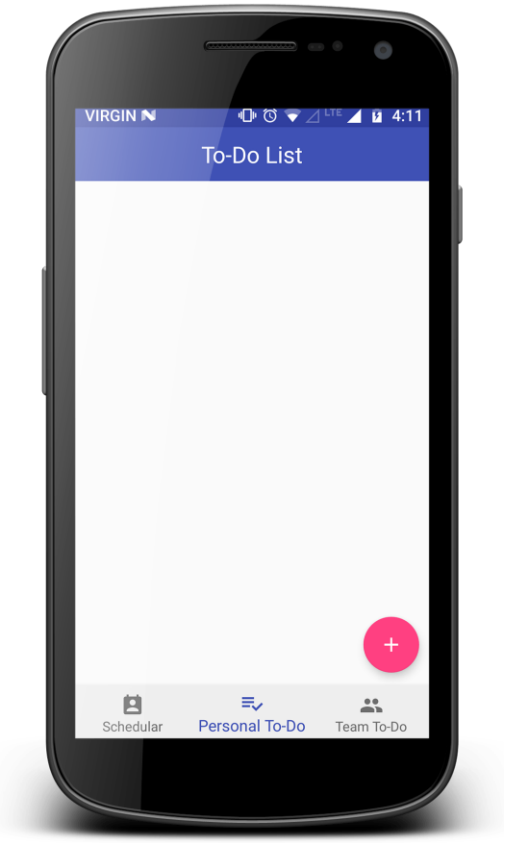


Figure 7: TO-DO List

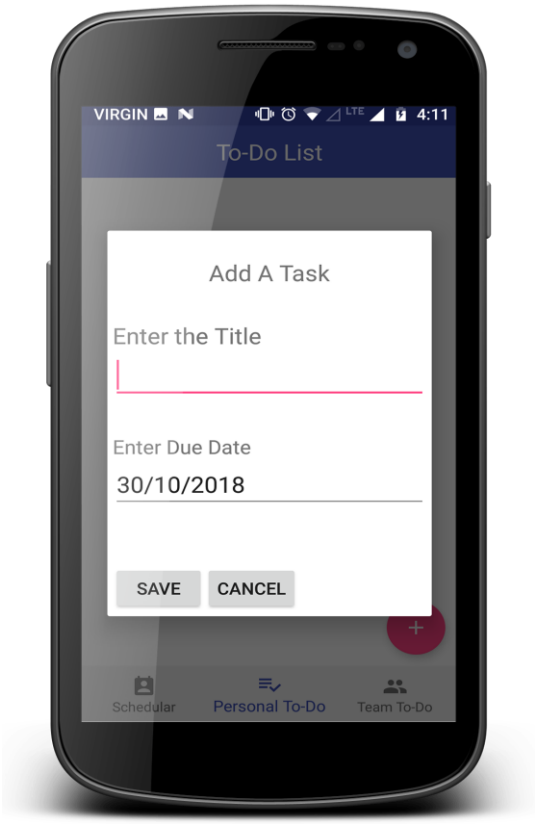


Figure 8: Create task

Figure 9: Date Picker

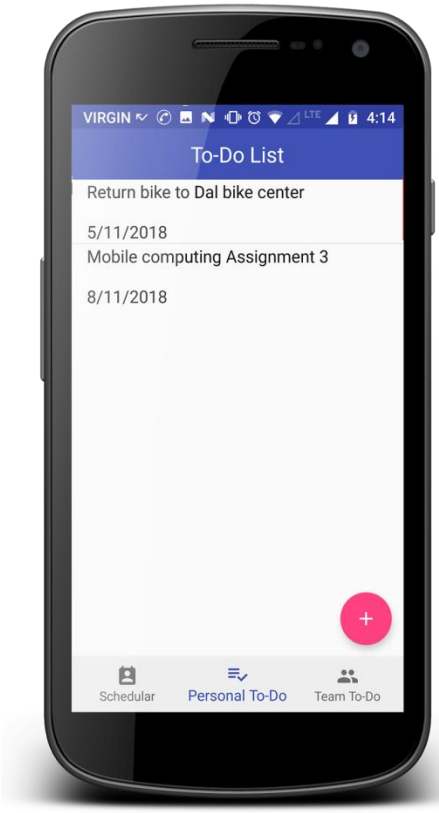


Figure 10: To-do List created

Calendar

The calendar is based on an event scheduler, which allows you to create, save, modify and delete events. The prototype shows the current UI of the application.

User can pick the date, and time from the calendar, as well as being able to create, save, modify and delete event. After creation, the events are saved for future reference.

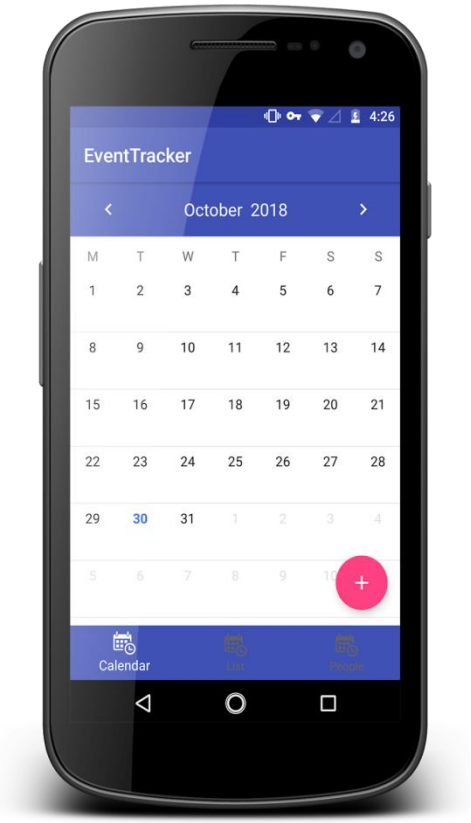


Figure 11: Calendar View

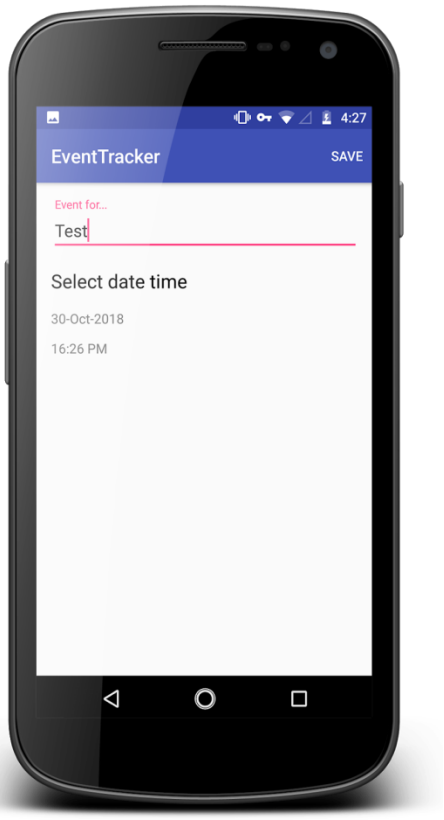


Figure 12: Event creation

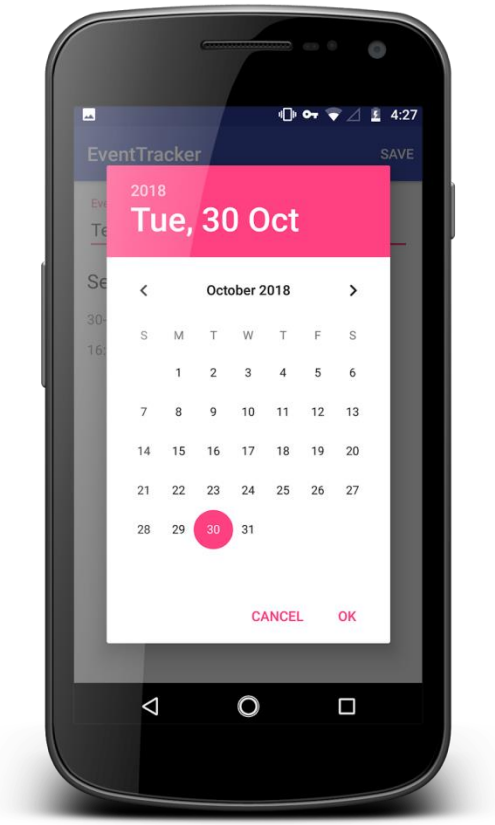


Figure 13: Calendar View

Figure 14: Calendar View

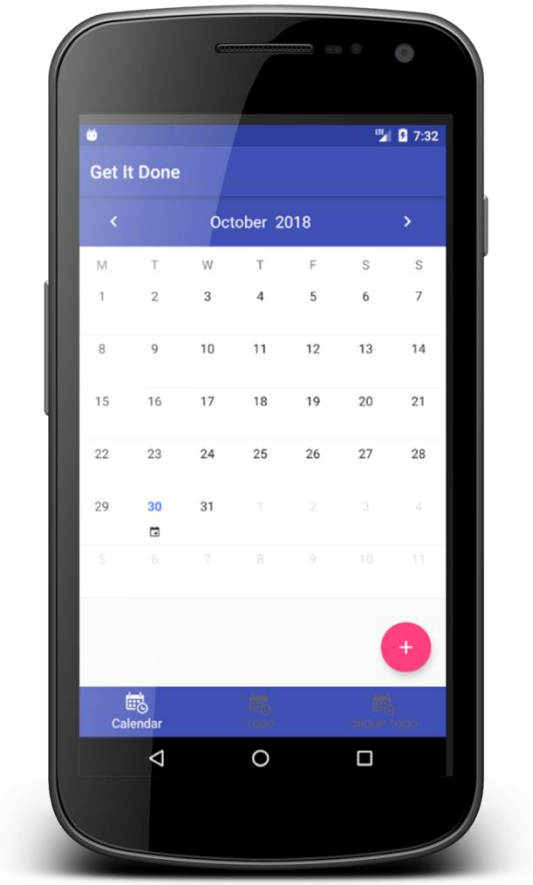


Figure 15: Saved Event

3. DESIGN CONSIDERATIONS

System Environment:
The GetItDone Application is designed to work on all Android Operating systems running on Marshmallow OS or earlier. The Application works off-line and may require GPS, if some of the bonus functionality is implemented.

Constraints:
The application is designed using Android Studio, and will only run on Android devices.

Assumptions:
The user of application is aware of basic operations of a mobile phone. The user also understands the standard terms used in each part of the application.

Design Methodology:

The system is designed with flexibility for further development, and modification. The system is divided into manageable processes that are grouped to sub-modules, and modules that are built with abstraction.

4. DETAILED DESIGN

Component design And Use Case

1. **Personal TO-DO List**

This is the fragment of the project whereby users can easily create a list of TO-DO activities, so that they have a tool which assists in managing short term tasks.

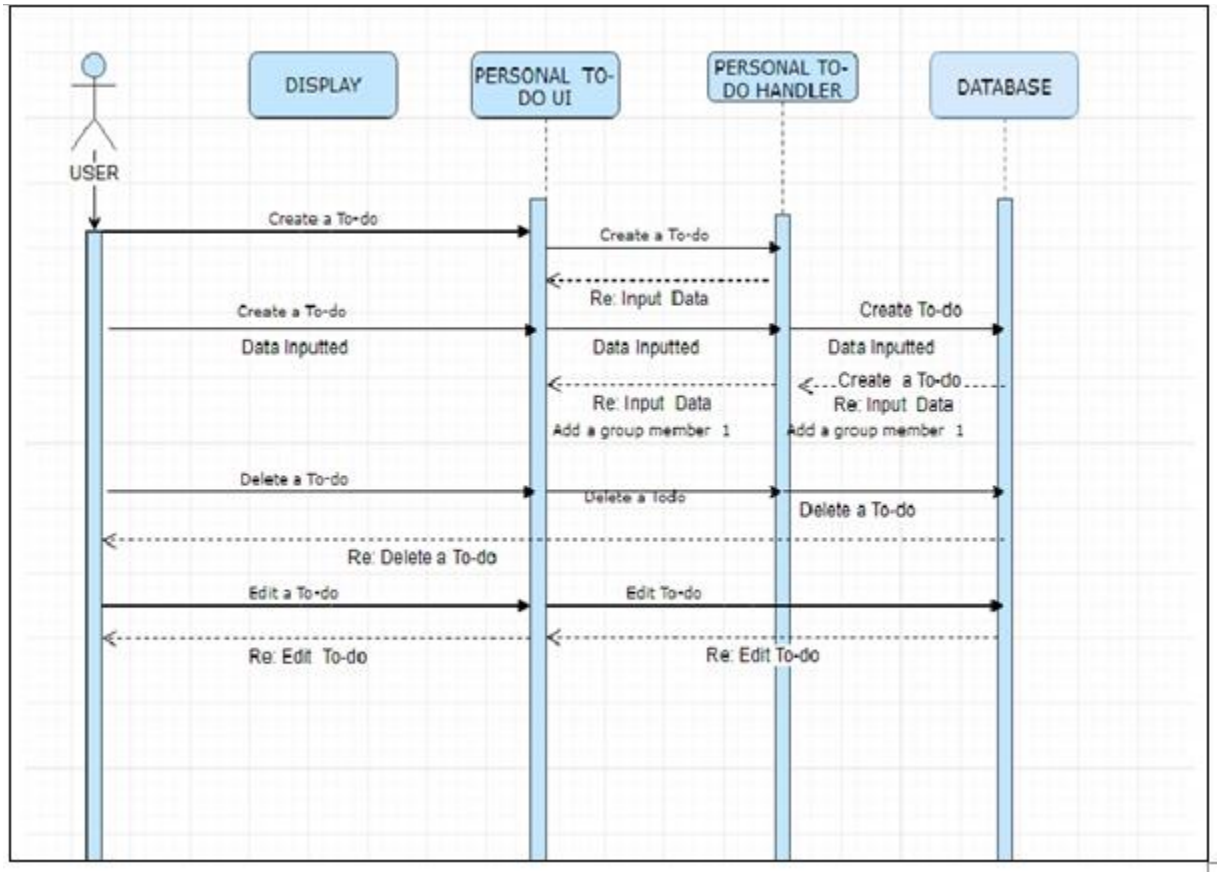
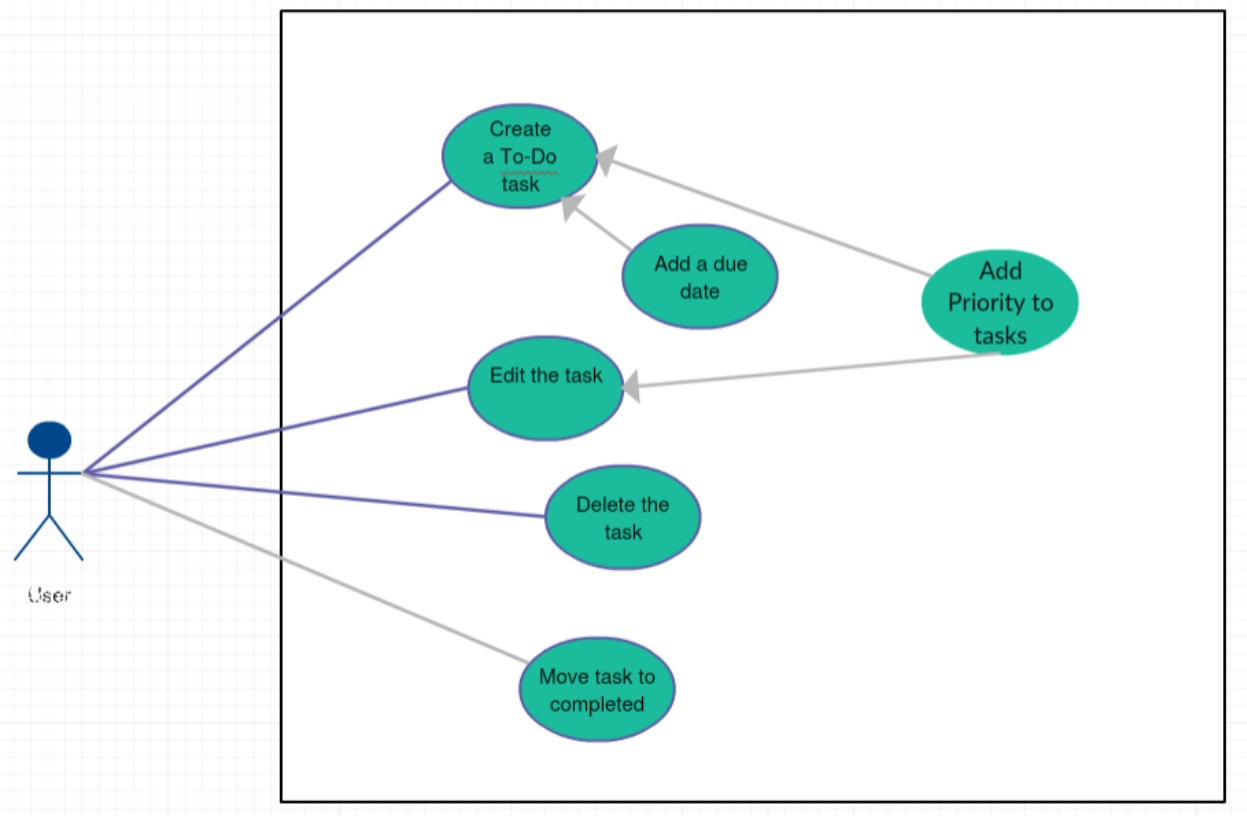


Figure 16: Sequence diagram for TO-DO List



Personal TO-DO List Use Case:

Create a To-Do task

STEP1:The user can create a new task by clicking on the add button.
STEP2:The user has to specify the title, due date and priority for the tasks.
STEP3:The user can then save the task.

2. Edit the To-Do task

STEP 1:The user can edit a created task by tapping on it and then editing the title, due date or priority.

3. Delete the task

STEP1:The user can delete the task by swiping left on each task item and then clicking on the delete button for respective list items.

4. Move the task to completed

STEP1:Once the task is completed, the user can move the task in a new list - completed task list. The user can at any time move back the item from the completed list to the task list.

2. Event Calendar –

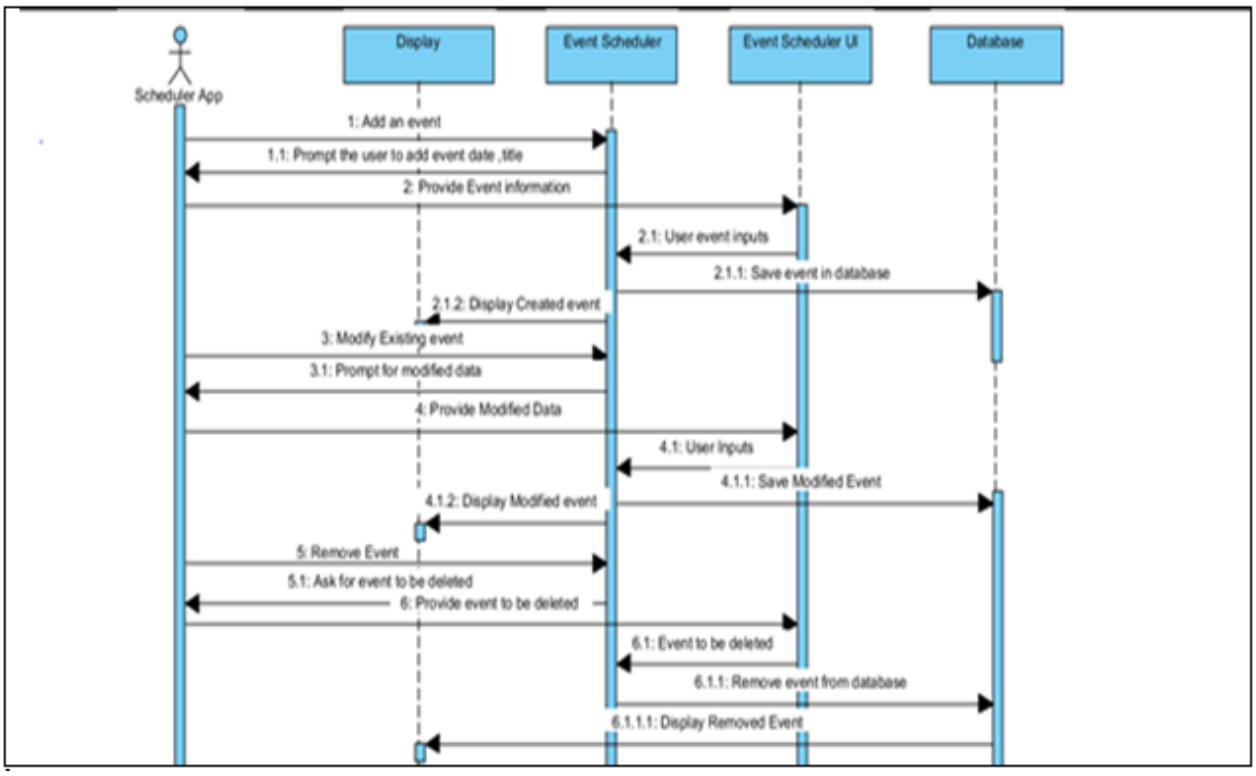


Figure 17: Sequence diagram for Event Calendar

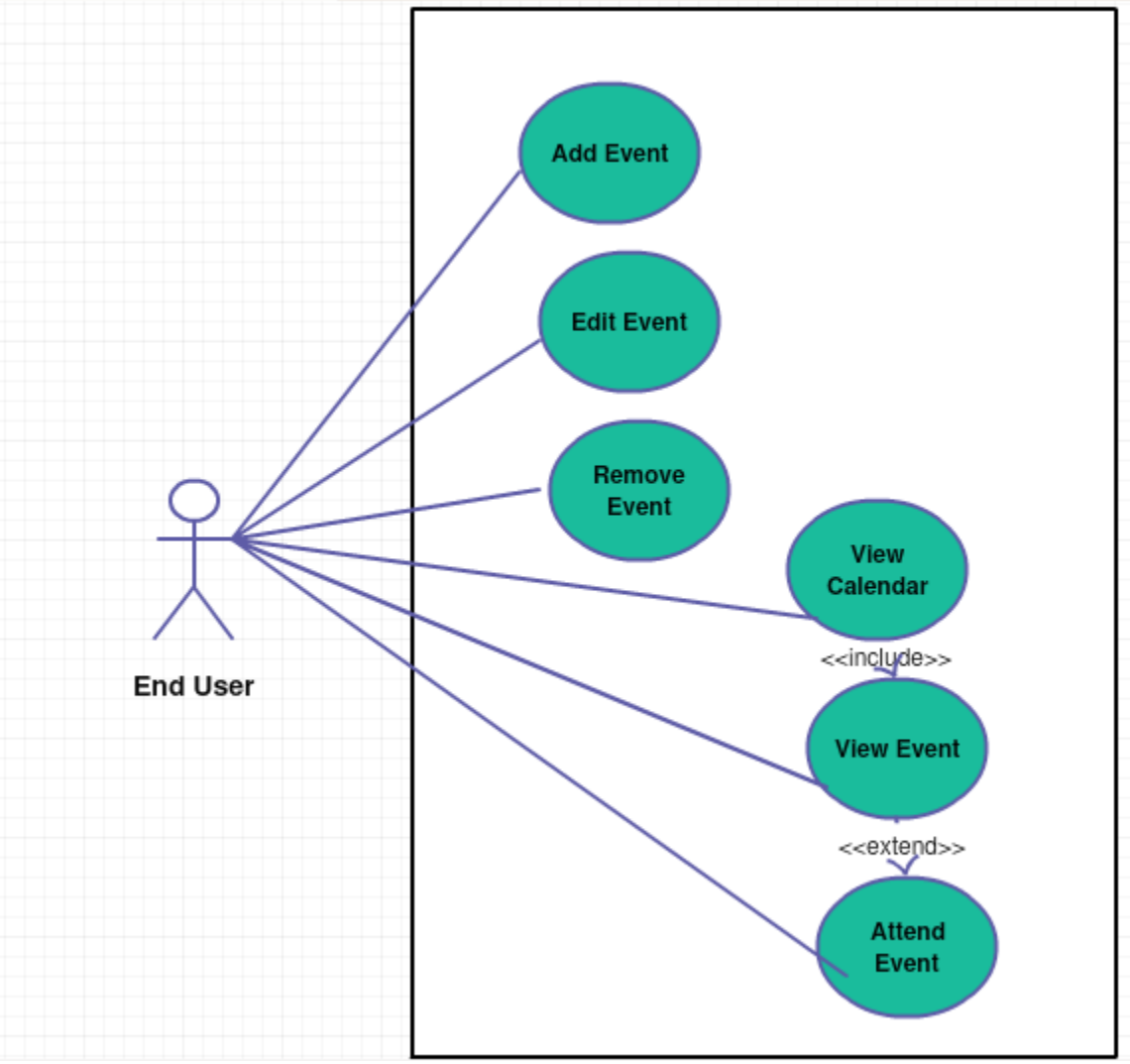


Figure 18: Use case diagram for Calendar Event

Calendar Event Use Cases

1. Create Event

STEP 1: The user selects date from the software display.
STEP 2: The user makes a change for date and Time.
STEP 3: The user confirms that the change is wanted.
STEP 4: The software saves the change.

2. Edit One Event in Calendar

STEP 1: The user selects an event from the software display and indicates that they would like to change it.
STEP 2: The user makes a change.
STEP 3: The user confirms that the change is wanted.
STEP 4: The software saves the change.

3. Delete One Event in Calendar

STEP1: The user selects an event from the software display (Calendar).
STEP2: User deletes the event.
STEP3: The user confirms that the change is wanted.
STEP4: The software saves the change.

4. View Calendar
The user launches and sees the Calendar Software.

3. Group TO-DO

This is a fragment of the app that helps group members collaborate, and share tasks.

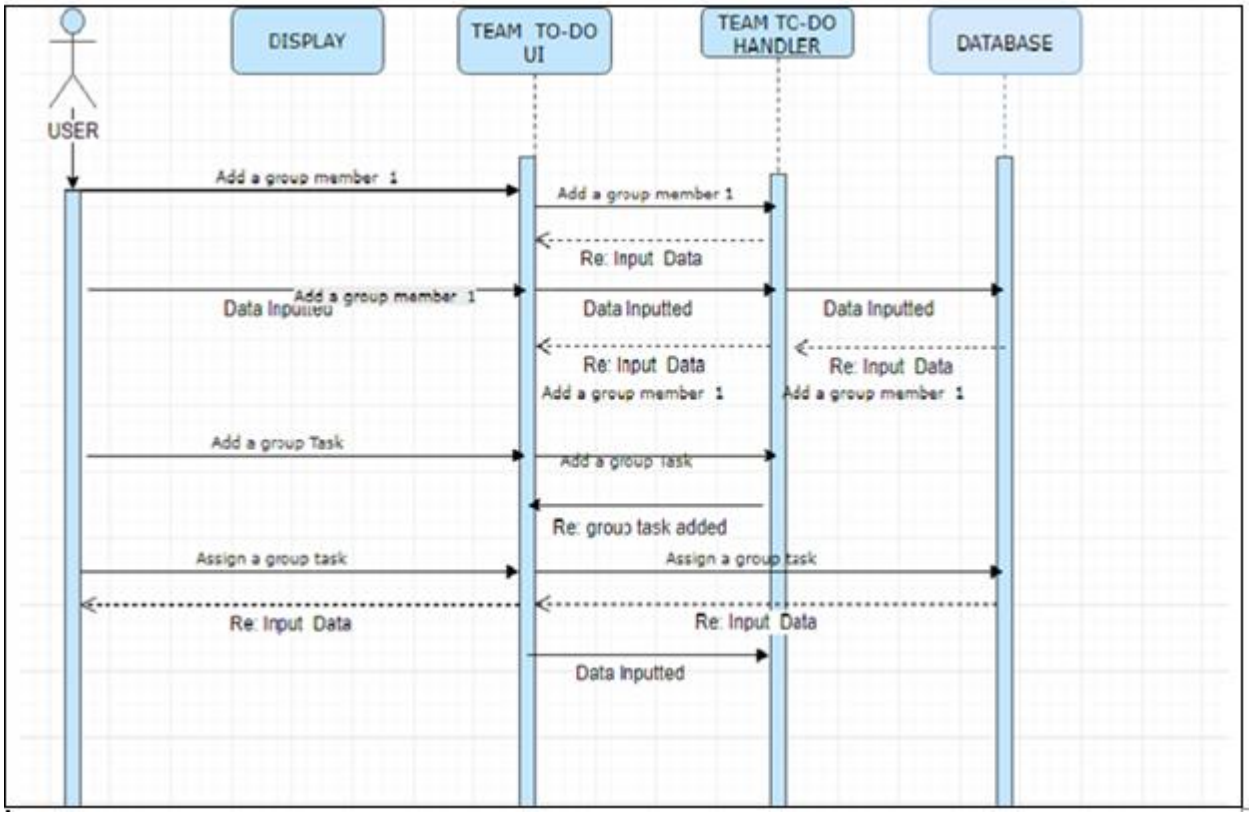


Figure 19: Sequence diagram for Group Task

1. Create Group Task

STEP 1: launch the Application.
STEP 2: Click on the Group icon Button.
STEP 3: Click on Add Task Button.
STEP 4: Input the Task to be done in the pane
STEP 5: Click on SAVE button

2. Create Group Member

STEP 1: launch the Application.
STEP 2: Click on the Group icon Button.
STEP 3: Click button Create Member.
STEP 4: Input detail of the Member.
STEP 5: Click on SAVE button.

3. Assign a task to group member.

STEP 1: launch the Application
STEP 2: Click on the Group icon Button
STEP 3: Click on the task display
STEP 4: click on the button "ADD MEMBER"
STEP 5: Select the member from the list of Members displayed
STEP 6: Click on the button "SAVE"

7. DATABASE DESIGN

We will use a Firebase database with the following design:
GetItDone

- Users
    - User ID
        - User ID
        - Name
        - Google login key
        - TO-DO
            - Tasks
                - Task ID
                - Task title
                - Task due date
        - Calendar
            - Events
                - Event ID
                - Event Title
                - Event start time
                - Event end time

- Group TO-DO
    - Group ID
        - Group ID
        - Group Name
        - Group Creator
        - Group Members
        - Tasks
            - Task ID
            - Task Title
            - Task due date
            - Task Assignee

8. TEST CASES

Junit Test cases

JUnit is a unit testing framework for the Java programming language. JUnit has been important in the development of test-driven development.

Espresso

Espresso is a testing framework for Android, and it makes it easy to write reliable UI test cases. Espresso automatically synchronizes your test actions with the user interface of your application. The framework also ensures that your activity is started before the tests run. It also lets the test wait until all observed background activities have finished.

We will write Espresso Test cases for UI testing.

9. SPRINT/ PRODUCT BACKLOGS

Sprint is a repeatable fixed time-box during which a "Done" product of the highest possible value is created. Sprint lies at the core of the Sprint agile methodology and can be thought of as an event which wraps all other Scrum events like Daily Scrums, Scrum Review and Sprint Retrospective. Like all scrum events, Sprints also have a duration. Usually, a Sprint lasts for one month or less, the outcome of the sprint being a deliverable.

In this project, we have divided tasks among ourselves as features for three independent components: TO-DO List, Calendar, and Group TO-DO Lists. For each component, features (Minimum, Expected, and Bonus) are divided among developers and we are planning to achieve minimum functionality by Project Update 1, Expected Functionality by Update 2 and Bonus with final submission. We meet every week to discuss the challenges and for integrating the work we have done till the date.

We are using the Issues feature on GitLab to assist with managing our tasks. Labels have been added to the issues with various useful purposes such as how many points a task is worth(1 point if a task should take 90 minutes or less, 3 points if it should take between 2 and 5 hours, or 5 if it is estimated to take more then 5 hours). Other labels include labels for every screen in the application, to help identify which issue corresponds to which section of the application, as well as labels for if the task has to do with the Google Authentication or Firebase integration, that we plan on incorporating into our app.

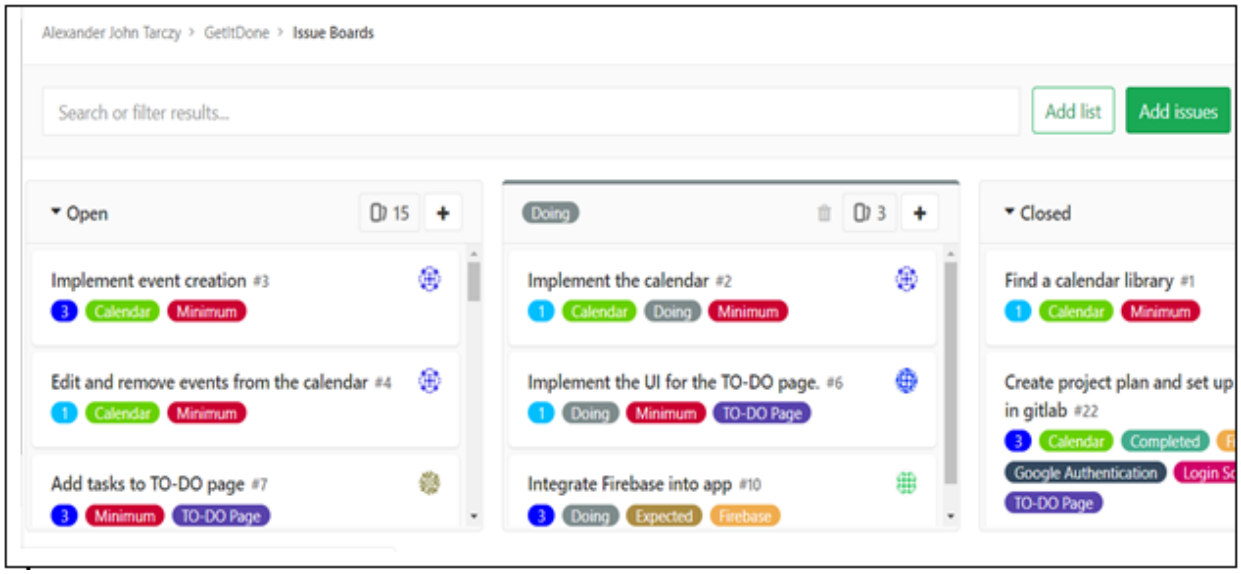


Figure 20 : GitLab Issues Screenshot

10. GITLAB

Find the source code for project.

https://git.cs.dal.ca/tarczy/GetItDone

REFERENCES

1. IEEE Standard 1016-1998, IEEE Recommended Practice for Software Requirements Specifications, IEEE Computer Society, 1998.
2. Mokhov, S. (2010). Selected Project Requirements. In Concordia. Retrieved from http://users.encs.concordia.ca/~c55414/selected-project-requirements.txt
3. En.wikipedia.org. (2018). Website wireframe. [online] Available at: https://en.wikipedia.org/wiki/Website_wireframe [Accessed 31 Oct. 2018].
4. En.wikipedia.org. (2018). Scrum (software development). [online] Available at: https://en.wikipedia.org/wiki/Scrum_%28software_development%29 [Accessed 31 Oct. 2018].
5. En.wikipedia.org. (2018). JUnit. [online] Available at: https://en.wikipedia.org/wiki/JUnit [Accessed 31 Oct. 2018].
6. En.wikipedia.org. (2018). Espresso. [online] Available at: https://en.wikipedia.org/wiki/Espresso [Accessed 31 Oct. 2018].