SCRUM BOARD MOBILE APP

LOW ZI JIAN

UNIVERSITI TUNKU ABDUL RAHMAN

SCRUM BOARD MOBILE APP

LOW ZI JIAN

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Science (Hons.) Software Engineering

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

May 2019

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : 

Name : Low Zi Jian

ID No. : 1603295

Date : 20 April 2020

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"SCRUM BOARD MOBILE APP"** was prepared by **LOW ZI JIAN** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Honours) Software Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature	:	D.gunavathi
Supervisor	:	Miss Gunavathi a/p Duraisamy
Date	:	20 April 2020

Signature	:	
Co-Supervisor	:	
Date	:	

ACKNOWLEDGEMENTS

First of all, I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my heartiest gratitude to my supervisor, Miss Gunavathi for her invaluable advice, knowledge and her enormous support gave me strength to carry out the project until its completion.

In addition, I would also like to express my most sincere gratitude to my loving parents who have encourage not to give up. I am overwhelmed in humbleness and gratefulness to acknowledge my depth to all my friends who had helped and given me encouragement especially during the development of the applications. An addition credit to Lee Hoe Mun, Chin Kai Xiang and Tan Chee Kuan whose assisted me during hard times in fixing bugs of the applications.

Last but not least, I am eternally grateful to all frontlines and medical workers who are not bow to this unfortunate calamity and working hard to combat Covid-19. A special thanks for our government's effort in combating this global epidemic as well. May God protect us as a nation and spring us to rise back stronger from this adversity.

ABSTRACT

Despite the facts that physical Scrum Board promotes interaction between team members when team members are able to see board and discuss how different tasks are progressing during daily stand-up meeting. The common benefit of the Scrum board to visualise project progress through several columns of sticky notes and pretty easy to see what tasks are done and what still need to be finished. However, there are still drawbacks and limitation with the current circumstances when the project become more complex as days goes by. Hence, a mobile application for scrum member to keep track their sprint activity / tasks and view scrum board of projects remotely should be developed along with a web application that allows scrum master to manage the projects specify for Scrum Methodology.

With the implementation of the both web and mobile application, all information and scrum boards can be accessed remotely with only internet required. Besides, the proposed solution helps in increase efficiently of scrum master's in project control especially in managing backlog items. Furthermore, scrum master is able to export the contents from the web application in form of report either in Excel or Pdf format with just a single click of the mouse. Furthermore , there is a feature in the mobile application that allows scrum members to update the sprint tasks 'status which allow them to commit to the project and reduce time waster as it can avoid unproductive stand-up meetings or answering emails that are interrupting their daily work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

LIST OF SYMBOLS / ABBREVIATIONS

Baas	Backend-as-a-Service
CDE	Continuous Delivery
CEO	Chief Executive Officer
CI	Continuous Integration
DoD	Definition of Done
DSDM	Dynamic System Development Method
FDD	Feature Driven Development
FYP	Final Year Project
HCID	Human Computer Interaction and Design
IDE	Integrated Development Environment
IT	Information Technology
MCO	Movement Control Order
NPM	Node Package Manager
Paas	Platform-as-a-Service
PMI	Project Management Institute
PXP	Personal Extreme Programming
ROI	Return of Investment
SDLC	Software Development Life Cycle
UI	User Interface
UML	Unified Modelling Language
URL	Uniform Resource Locator
WBS	Work Breakdown Structure
XP	Extreme Programming

CHAPTER 1

INTRODUCTION

1.1 Introduction

This chapter displays a holistic view of the project charter. Section 1.1 explains the purpose of other sections in this chapter. Section 1.2 is about the background of the project. Section 1.3 describes the problem statement and limitations of current technologies and engineering exercises. Section 1.4 narrates the project objectives, the key performance indicators that will be used to assess the project's success. Section 1.5 is about the proposed solution to tackle the problem stated. Section 1.6 briefly describes the project approach such as software methodology that will be used. Section 1.7 will be the last session and it is about the project stakeholder.

1.2 Background

In 1986, Scrum was first introduced by two management experts, Hirotaka Takeuchi and Ikujiro Nonaka in "The New Product Development Game". In 1990, Schwaber and Sutherland implemented the Scrum approaches in their companies. After 5 years later, they presented an official introduction of the Scrum approach and its application to the whole world. In the 2000s onward, they started to promote and encourage other companies to practice Scrum approach by providing Scrum guidebook such as "The Scrum Guide" and professional Scrum accreditation series through Scrum Inc, Scrum Alliance, Scrum Org (Lynch, 2019).

Figure 1.1: Scrum inc Logo

Figure 1.2: Scrum Alliance Logo

Figure 1.3: Scrum org Logo

Scrum is one of famous agile methods practiced nowadays to replace the inadequacies of traditional development methodologies such as the Waterfall methodology. In 2017, there were 94 percent of respondents used Scrum in agile practice (ScrumAlliance, 2017) . Scrum is a better way to develop product by providing a clear and measurable structure that promotes iterative development, team collaboration and change recognition.

Scrum is the framework and Scrum tools are used facilitate this framework, emphasizing transparency and team collaboration in software development life cycle. Organizing workloads is also part of their nature. The most critical Scrum tool used is known as the Scrum Board. It provides a general overview of product backlog and allows development teams to plan their sprint activities with sticky notes on a white board. However, traditional Scrum Board has its limitations. It is undeniable more systematic if we could turn these sticking notes into task card in an online Scrum Board application. In addition, it is convenient as it synchronizes the activities discussed physically during daily stand-up meetings.

In conclusion, usage of online Scrum Board is the most cost- effective way to project control in Scrum development. It saves time for Scrum team to make corrections and adjust priorities of sprint activities remotely. In order to help green Scrum practitioners to enact effortlessly, the main goal of this project is to develop an online Scrum Board application designated for all general use in both mobile and web platform. The other goal of this project is to promote the idea of hybrid agile for either small or large, experienced or green developer teams to CDE especially using Scrum method.

1.3 Problem Statement

According to State of Scrum 2017 report by Scrum Alliance, it was 63 percent success rate of Scrum projects. Even though Scrum methodology has been introduced for quite a duration, Scrum agile method still isn't convincing for new users. There are not many organisations could fully utilise it due to less availability of Scrum software especially on mobile platform.

Since Scrum is a fast moving, rapid iterative development methodology, it is nearly impossible to plan the product development process in an early time without good experience and skills in agile methods or a remarkable insight onto the project's future. Besides, most of these softwares are monthly paid subscription and premium, which is unstainable for small company. Most importantly, it is difficult for the development team keep track of their progress without a good Scrum Board.

Traditional way of performing Scrum Board has disadvantages and brings great impact of team members' dedication to the project. It also dilutes team working spirit as Scrum Master having hard time to monitor and manage the progress especially team members who work in different countries (Gunga, Kishnah and Pudaruth, 2013).

Physical Scrum Board has limited access to monitor their user story development and testing according to their sprint cycle. It could be a nuisance and annoyance to development team to read the details of sticky notes or search their tasks within these notes. It is not suitable for larger and complex projects, especially for offices with limited space (Gunga, Kishnah and Pudaruth, 2013). It is very complicated to manage and maintain the project even it is separated in multiple tasks boards. In addition, the tasks allocated could no longer be distinguished by just using colours.

Furthermore, the Scrum master might perform updates onto these details anytime he needs to. A memorandum regarding the updates might be sent in the next working day or participating an extra stand up meeting which leads to an unnecessary waste of time. Scrum master usually has difficulties to manage sprint backlog and product backlog, completed or unfinished requirements with just paper documents. Some of these important logbooks and documents printed in ink has high chance to get missing or shredded accidentally, and even worse, without a backup. There is a high risk of duplication, distribution and alteration for these paper documents that contributes as a leaked patent of product to competitors.

In order to overcome the problems stated above, the aim of this project is to develop a mobile application for Scrum Board which synchronises with a web

application which serves as overall project management system to monitor the requirement management and project activities.

1.4 Project Objectives

The objectives of this project are:

1. To analyse project requirements and its specifications by reviewing existing similar applications during Project 1 period.
2. To develop Scrum board applications which are easily learned and adapted by inexperienced Scrum user.
3. To design and develop a prototype model of these Scrum Board applications during Project 1 period.
4. To develop and design web application to create and monitor the overall project activities and requirements in product backlog and sprint backlog within Project 2 period.
5. To develop and design a mobile application that serve as Scrum Board that has push notification function within Project 2 period.

1.5 Project Solution

The problem stated in this proposal is urged to be eradicated through the creation of Scrum Board mobile application along with its management system in web application. Potential users and stakeholders of the application such as our supervisor or software development team will provide requirements and expectations that drive this project to favourable result. Adoption of this Scrum Board application into their Scrum project will bring great impact and assistance in enhancing solidarity amongst development team members and seeking for better improvement of their final product.

This reliability and quality of Scrum Board application are decided by how responsive and the way of retaining a neat visualisation of icons and data. Most importantly, this management system should have a customizable Scrum Board with swim lanes that occupy with Sprint tasks for each Sprint Development that is uncomplicated to use and understand for green Scrum practitioner. Management of Sprint Backlog and Product Backlog of the management system is top priority as it the fundamental key of the mechanism of Scrum agile method. Moreover, Scrum Board mobile application should display the user tasks and its notifications at the mobile notification panel once there is immediate update from the Scrum Master and its

deadline. Security of this application should not be neglected since there will be multiple users using the application concurrently.

These are some possible software development IDEs, that will be used in the project implementation and development phase.

Figure 1.4: Android Studio Logo

Figure 1.5: Visual Studio Code Logo

These are some possible prototype layout design tools or references that will be used in the project design phase.

Figure 1.6: Dribble Logo

Figure 1.7: Figma Logo

Figure 1.8: Enterprise Architect Logo

These are some possible software testing tools that will be used in the project testing phase.

Figure 1.9: Selenium Logo

Figure 1.10: Appium Logo

This is an overview of high-level architecture design of the final product. The native mobile application for Android will be developed using React native Firebase sync with web application which supported in Laravel Framework that uses bootstrap layout with support from firebase real time database and cloud storage. Firebase also provides user authentication to the login system.

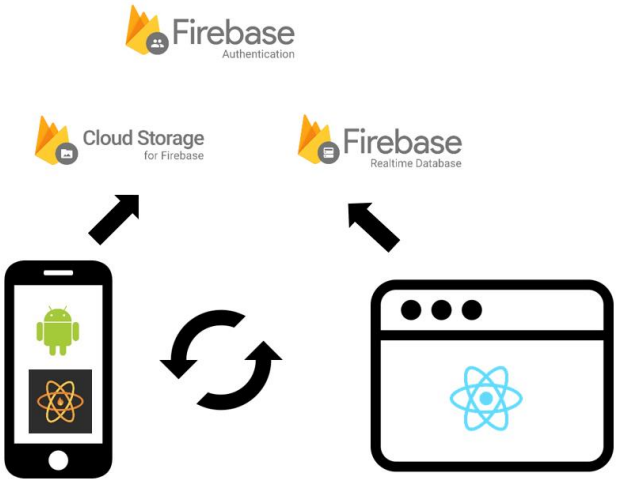


Figure 1. 11: High level architecture design of applications

1.6 Project Approach

The software development approach used in this project is Personal Extreme Programming (PXP) since the project has tight deadlines and inadequate working resources. PXP is one of the agile development methodologies which promotes adaptive planning and continuous improvements. In addition, this project requires a

lot of engagement with stakeholders. Every feedback made by the stakeholder will bring an impact to the requirements of the project. This iterative and flexible approach will fully eradicate problem of frequent changes in the project. PXP lifecycle is designed based on incremental build model Although the Scrum Board application is developed with a simple design at the beginning, but I believe that the stakeholder ‘s satisfaction will increase as more app modules are delivered continuously and rapidly in each small release. Testing on the Scrum Board application is performed concomitantly in each cycle will accumulate its quality as well as strengthen trust of the stakeholders toward it.

One of best PXP practices that could be implemented in this project is “40 - Hour Week”. It allows autonomous developer to stop working when he is getting stressed or tired and no longer production. There is a need to develop a prototype in the project 1 session and the duration for it is very limited. ‘Simple Design’ of the PXP practices could help to smooth progression of the development of prototype without further worry about adding new features.

1.7 Scope of the Project

1.7.1 Target Users and Stakeholders

The main deliverance of this project is proposed specifically toward new and inexperienced Scrum user. They are able to learn the basics of agile project management using Scrum by using the Scrum Board applications. Hence, the target users of this Scrum Board applications are inexperienced Scrum practitioners, including the roles of Scrum Master, Scrum Team and Product Owner.

The stakeholders of this project are my Final Year Project (FYP) supervisor, Miss Gunavathi and UTAR as it has ownership of the Scrum Board mobile application and its web management system.

1.7.2 Modules Covered

The modules stated below shall be provided by Scrum Board application in order to achieve the project objectives:

The Scrum Board application shall be run on the mobile platform (Android) and web platform.

Scrum Board mobile application:

Main user: Scrum Team

(i) Update the status of sprint task card

The Scrum Board mobile application shall allow user to update their progress of their own task cards. There shall be 3 types of status of task card which are to-do, ongoing, completed given by default. User shall be able to remark inside the allocated task card. Any changes made will be saved and reflected in the web application.

(ii) Display an overview of Scrum Board of ongoing sprint

The Scrum Board mobile application will display a general overview of current sprint progress in Scrum Board format. Detailed Information of task card in swim lanes of the Scrum Board will be displayed when the user selects it.

(iii) Display notifications of updates or changes

The Scrum Board mobile application will notify any recent updates made by Scrum Master through web application or to remind the user's tasks 'deadlines. For example, the application will notify the user a week before the task meets its deadline by default.

(iv) Display a simple list of user's sprint tasks

The Scrum Board mobile application will be filtered automatically and display only the user's sprint tasks.

(v) Display available and unassigned sprint tasks

The Scrum Board mobile application will display a list of unassigned sprint tasks. User shall be able to select these tasks and get approval from the Scrum Master. Once permission is granted, the sprint task will be added in the user's sprint tasks list.

(vi) Display a list of highlights

The Scrum Board mobile application will display a list of recent updates made by other users in the Scrum Team or Scrum Master. Detailed Information of highlight will be displayed when the user selects it.

Scrum Board web application:

Main user: Scrum Master

(i) Display an overview of customizable Scrum Board of ongoing sprint and previous sprint

The Scrum Board web application shall serve as the overall project management system for Scrum master to monitor the requirement management and project activities. Scrum Board web application shall display a detailed overview of Scrum Board of ongoing sprint with swim lanes. Detailed Information of task card in swim lanes of the Scrum Board will be displayed and configurable when the user selects it. User could also drag and drop the task card in between swim lanes. Moreover, user can create or delete sprint task card and assign to invited Scrum team member. Any updates/changes made from mobile application shall be reflected in the mobile application.

(ii) Maintain multiple projects concurrently

The Scrum Board web application shall allow user to create Scrum Board for multiple projects. User could only add maximum 9 team members in a single project.

(iii) Maintain product backlog and sprint backlog

User can create and maintain the product backlog and sprint backlog with the Scrum Board web application. They could export these backlogs items in excel extension from the Scrum Board web application.

(iv) Provides a history log of updates and changes

The Scrum Board web application shall display a history log of all updates and changes made in a project. Besides, it also shows a list of completed sprints.

(v) Display notifications of updates or changes

Scrum Board web application will show notification to user of new updates of Scrum team member’s activities from the mobile application.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter describes the researches made for the project's literature review. Section 2.1 explains the purpose of other sections in this chapter. Section 2.2 is about the purpose and how the literature review is done in this project. Section 2.3 describes the concepts of Scrum Methodology. Section 2.4 describes the concepts of Scrum Board. Section 2.5 lists out the reviews on existing Scrum Board applications. Section 2.6 briefly describes fact-findings on the adopted project methodology. Section 2.7 will the last session and it is about findings on project development tools.

2.2 Literature Review

The main objective of Literature Review is to perform research and evaluation of available literatures which related to the project's topic area. During the research, critical analysis is accomplished in order to synthesise the right amount of information which helps those who might be reading this proposal to gain better understanding. In this project, the literature review is organised into main 3 domains which are concepts of Scrum Methodology and Scrum Board, research on project methodology and development tool, reviews and evaluation on existing Scrum Board applications. A literature review matrix is generated afterward.

To present the literature in an organised way, integration and summarisation of knowledge related to the project has been done. There are several sources of Information that inspire us of the thought of developing an application which will overcome the drawbacks of the existing applications.

Primary Resources:

- Web Sites
- Online Journals
- Online Articles

Secondary Resources:

- Online Public Reviews and Discussion

2.3 Scrum Methodology

This section explains about the basics understanding of Scrum methodology which includes topics such as:

- What is Scrum and Sprint?
- How Scrum Works?
- Scrum Development: who is involved?
- The three most important Scrum Artifacts
- What is the Definition of Done (DoD)?
- Scrum Metrics

2.3.1 What is Scrum and Sprint

Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value. Scrum is used to develop all kinds of technology products such as software, hardware and non-technology products such as autonomous vehicles, schools, government or even managing business and marketing operations. It is widely used for individuals and societies (Schwaber and Sutherland, 2017) .As stated in State of Scrum 2017-2018, Scrum agile method mainly used in IT , followed by Software Development and Product Development. Recommended Scrum team size is 3 to 9. Having more than nine members requires too much coordination and fewer than three Development Team members decrease interaction and results in smaller productivity gains (Schwaber and Sutherland, 2017) .

Sprint is the core of Scrum. It is a container of activities that leads to the production of profitable product. Sprints duration should be consistent. Each sprint is strategized and required to be completed with a consistent duration of one month or less. Short sprint takes about 1 to 2 weeks and long sprint takes about 3 to 4 weeks (Levison, 2013) .There will be an increment of the product's features and reduction of its defects in every new sprint.

In conclusion, a successful product was built on sufficient number of sprints in Scrum.

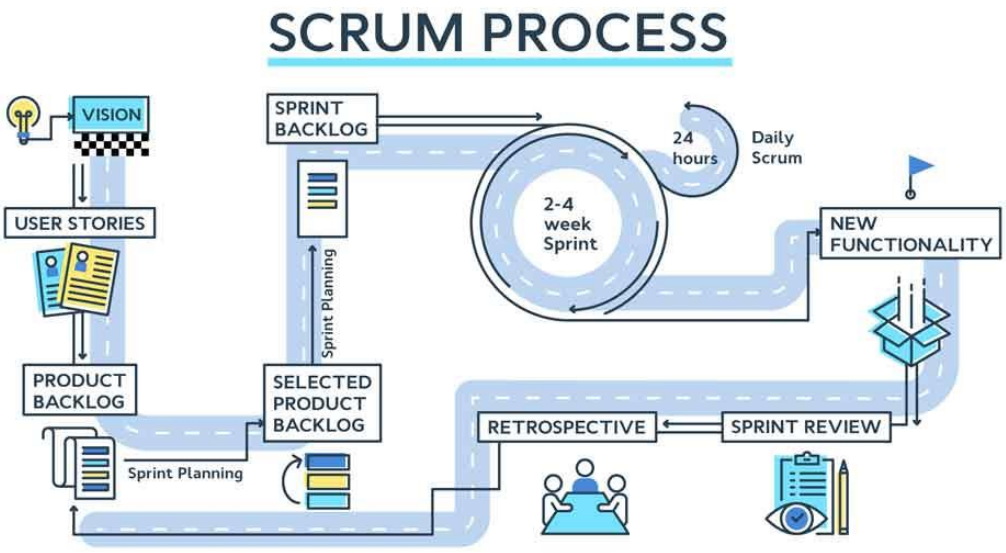


Figure 2.1: An overview of Scrum process

2.3.2 How Scrum Works?

At the stage of project initiation, the product owner gathers customers / client's business needs and transform them into user stories, generally known as user requirements. The Project Vision also must be taken importantly and should be included with collected user stories in the project charter.

When the user stories have been confirmed and also signed off, the project team, product owners and the stakeholders work on breaking down the user stories into a more detailed list of product backlog item. The list of Product Backlog is further sorted based on their priority and story points.

During the sprint planning, the Scrum team discuss about sprint backlog items (each unique individual tasks that break down from the product backlog item) , sprint goal of a new sprint. Proper time and effort estimation were made before the scrum master clarify discussed details with the product owner.

A Sprint is defined to produce a new functionality or a fully deployable, commercial product. During a sprint, Daily Scrum, a 15-minute time box meeting is held every day of the sprint to keep track of progress and guarantee completed activities are meeting the scope (Schwaber and Sutherland, 2017). At the end of the sprint, a Sprint Review meeting is held to present the product and achievements of the recent sprint to the stakeholder.

Finally, a Sprint Retrospective occurs simultaneously after the Sprint Review. It is an inspection of previous Sprint and provide insights of improvement to be implemented in following Sprint. The development team seizes the opportunity to seek for potential risks and major items that can be consolidated to save time and effort for the next sprint.

These sprint iterations are executed repetitively until all product backlog items have been completed. Figure 2.1 shows a complete overview of scrum process.

2.3.3 Scrum development: who is involved?

Figure 2.2 shows an overview of Scrum roles and they are:

2.3.3.1 Scrum Team

A Scrum team accomplish the tasks in each sprint (Team Player) , such as business analysts, system analysts, developers (Permana, 2015). They deliver products iteratively and incrementally , maximising opportunities for feedback (Schwaber and Sutherland, 2017).

2.3.3.2 Scrum Master

A Scrum Master monitors progress of sprint, tasks allocation, delivers the idea of Product Owner correctly to the development team. Moreover, a scrum master is responsible to help the organization to understand and enact Scrum and empirical product development (Schwaber and Sutherland, 2017).

2.3.3.3 Product Owner

Product owner plays a vital role in managing product backlogs, optimising making product backlog items transparent , reaching an understandable level for the scrum team (Schwaber and Sutherland, 2017).

Figure 2.2: An overview of Scrum Roles (ScrumAlliance, n.d.)

2.3.4 The three most important Scrum Artifacts

2.3.4.1 Product Backlog

Figure 2.3: Sample of product backlogs

Product backlog is the list of everything that is known to be needed in the product e.g.: enhancements, requirements, features and many more. Product backlogs are dynamic and will never be completed because requirements will never stop changing due to market conditions or

advancement of technology. This explains product backlog refinement is frequently implemented during the project life cycle. This attribute of a transparent and clear product backlog item:

- User Story / Requirement:
  Define a distinct piece functionality that has yet to be delivered for a product.
- Priority:
  Uses MoSCow Prioritization method, separate to 4 stages of prioritization categories which are must have, should have, could have and will not have.
- Estimates / Story Points:
  Offer relative values for the level of effort required to complete a product backlog item, taking into account:
  - The amount of work that needs to be completed
  - The risk and uncertainty brought by the item
  - The complexity of the task

  The scale for the point system is Fibonacci sequence, best to use range is 1 to 21. The larger the story point / estimate is, the more the backlog item should be prioritised. The most popular technique for estimating story points is Planning Poker.
- Type
  These are the 6 types of product backlog item:
  - Feature (or User Story): can be big which known as “epics” or small known as “quickies”
  - Change
  - Defect
  - Technical Debt
  - Business case
- Status
  The current completion status of the product backlog item.

2.3.4.2 Sprint Backlog

Sprint Backlog is the set of product Backlog items that are selected for a sprint. The Sprint Backlog is a forecast by the development team about what functionality will be in the next increment and the work needed to deliver that functionality into a “Done” increment.

2.3.4.3 Increment

Increment is the sum of all the product backlog items completed during a sprint and the value of the increments of all previous Sprints (Schwaber and Sutherland, 2017). Each increment should meet the “definition of done” , it is inspected and acknowledged by the product owner at the end of Sprint.

Other than these, there is a crucial Scrum artifact known as User Story which defines the general product features from an end-user / stakeholder perspective that is presented as a simplified description of the product's requirements. These user stories are further broken into detailed Product Backlog items by Product Owner.

2.3.5 What is the Definition of Done (DoD)?

Definition of Done (DoD) is a collection of valuable deliverables required to produce software (Panchal, 2011). Every Scrum Team has its own DoD and consistent acceptance criteria across various levels including Definition of Done for a Feature (Story or Product Backlog Item), Definition of Done for a Sprint (Collection of features developed within a sprint), Definition of Done for a Release (Potentially shippable state).

The product backlog items must meet acceptance criteria (Mitchell, 2017) follows by:

- Reviewed by stakeholders
- Pass and complete unit acceptance testing of User Story
- Pass and complete quality assurance test
- Documentation related to User Story is completed
- All issues are fixed
- Successful demonstration to stakeholders

2.3.6 Scrum Metrics

According to Sealights (n.d.), these are some of the essential sprint metrics which are easy to learn through visualisation of data:

- **Burndown chart:** an overview of progress over time while completing a project. As product backlog items are completed over time, the graph "burned down" to zero.' It could be also used as a sprint burndown chart, provide quick visualisation of sprint progress and remaining tasks in sprint backlog
- **Team velocity:** measures the velocity of the development by displaying an overview of sum of completed user stories for each sprint, provides insights to the team for better estimation onto time and workloads they could complete in the future
- **ROI (Return of Investment):** measures the total revenue generated from a product and the cost of sprint used to develop it. If economic value of the project is higher than the cost it used, it means the project is in "healthy" state
- **Capital Redeployment:** an alternative option of ROI to determine the condition of project by calculating actual cost which is the cost of sprints to develop the required backlog item and opportunity cost which is an additional cost that to improve the current product out / in of the product backlog scope.
- **Customer satisfaction:** ensures that the product is developed at optimal level of acceptance and satisfaction, metrics used: NPS (Net Promoter Score)

2.3.7 Summary

Scrum is an agile way to manage project usually software development. A Scrum project's artifacts are Product Backlog, Sprint Backlog, Product Increment and User Stories. It is composed of multiple Sprints with consistent duration. A Scrum project usually takes up 3 to 9 team members. A Sprint usually carried out in 2 weeks. A Sprint's success depends on the leverage in achieving "the definition of done" and contributes heavily to a project's success. Moreover, Scrum metrics also determines the success of the project.

2.4 Scrum Board

Exploring existing Scrum board applications is the best way to elicit functional and non-functional requirements. Comparing their drawbacks and unique features helps to

create new development solutions to increase the reliability and convenience of the final product. Moreover, visual layouts of these applications provide great inspiration and guidance to the project's user interface design. These are a few topics that illustrates the discussed idea:

- What is Scrum Board?
- The basic functionalities of Scrum Board
- Format of Scrum Board
- Physical Scrum Board vs Online Scrum Board
- What are the existing Scrum Board Applications?

2.4.1 What is Scrum Board?

Scrum Board is used to help teams to manage and perform the projects using the Scrum project management framework rapidly, seamlessly and flawlessly. It is a visual representation of sprint backlog items that needed to be done. Scrum Board can be used in both physical form (whiteboard and sticky notes) and virtual form (software and tools). Scrum Board helps Scrum Master and his teams to keep track of the progress of the tasks and perform regular updates before meeting the deadline.

The importance of the usage Scrum Board is to encourage team interaction and discussion through emphasizing on the visualisation of the sprint cycle. Another key to promote team collaboration in Scrum Board is its functionality to support full team commitment by displaying an isolation view of tasks for each user and tune out the rest.

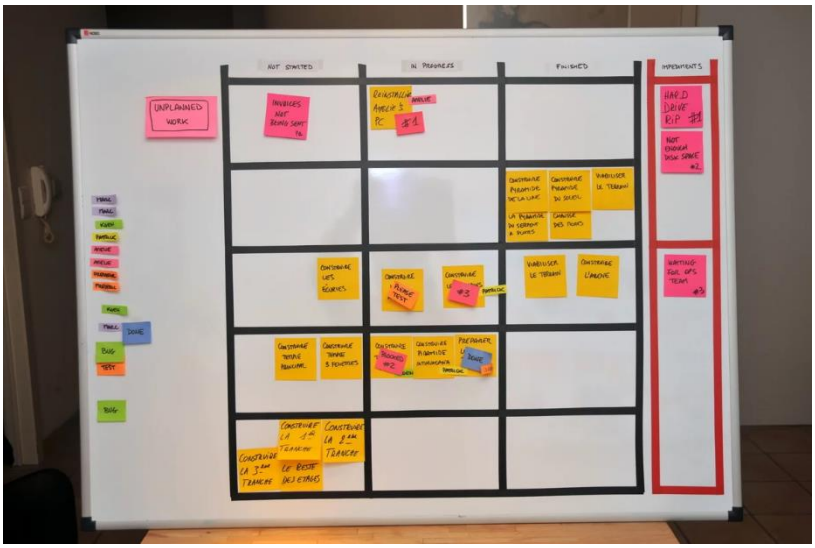


Figure 2.4: Scrum Board (Smartsheet, 2019)

2.4.2 Format of Scrum Board

In its simplest form, a Scrum Board displays a list of works/tasks (sprint backlog items) of user story card (Product Backlog item) that needs to be accomplished in the current sprint. A Scrum Board has three pillars that indicate the progress of these tasks. These are the column typically included on a Scrum Board

- Stories / Product Backlog: The product backlog item that are included in the current sprint
- To-Do: The tasks for a story that haven't started yet
- In-Progress: All tasks for work that started
- Done: All completed tasks.

Besides, the title of Scrum Board usually named after the Sprint Goal or the number of ongoing Sprint.

How does the Scrum team avail the Scrum Board? Sprint backlog items in To-Do column are allocated to the team members. The team members put in effort and show their capability in completing the allocated task by moving these backlog items from To-Do column to In-Progress column, finally end in Done column with the acknowledgement of the Scrum Master.

2.4.3 Physical Scrum Board vs Online Scrum Board

This is features comparison of Physical Scrum Board and Online Scrum Board. The most effective Scrum Board depends on the size of the teams. An Online Scrum Board is more suitable for large, remote team. On the other hand, Physical Scrum Board that focuses on physical interactions is more satisfactory for small, local team. Consequently, the initial phase in actualizing Scrum is separating the exceptional needs, qualities, and shortcomings of your group before choosing how you will sort out and track the work being finished (Smartsheet, n.d.).

Table 2.1: Physical vs Online Scrum Board (Smartsheet, 2019)

Feature	Physical Scrum Board	Online Scrum Board
Works for local teams	✓	✓
Works for remote teams		✓
Customization options (color, column names, task information)	✓	✓
Intuitive for new Scrum teams	✓	
Automatic updates/alerts		✓
Track changes/history		✓
Easy to visualize	✓	
One team member can make adjustments at a time	✓	
Promotes face-to-face interaction	✓	
Attach documents		✓
Control sharing/editing permissions		✓

Table 2.1 shows that physical Scrum Board indeed has higher visibility and transparency of current status of the Sprint progress to not only to the important roles of the Scrum process, also can be viewed by stakeholders, business team, CEO and everyone (Fadillah, 2018) . However, Online Scrum Board could compromise large space for Sprint Backlogs items with sub tasks and go paperless. Historical movement and important event can be tracked using Online Scrum Board. Graphical output such as burndown chart, Team Velocity is easier to be calculated and delivered with lesser effort with the help of computational system in Online Scrum Board application.

2.4.4 Summary

Scrum Board is the tool that highlights Scrum framework ‘s principles which are inspection, adaption and transparency. It is suggested to combines both the advantages of both. physical Scrum Board improves team collaboration, communication and interaction where the online Scrum Board allows remote and asynchronous collaboration. Leverage both for what they do the best.

2.5 Existing Scrum Board Applications

This chapter discusses about the functionality of each existing Scrum board management tool and explains why some of these tools aren't suitable for new Scrum Users.

2.5.1 Jira

Figure 2.5: Jira Logo

Jira is one of the best software development tools used by agile teams which is highly recommended. It has the best configuration setting to adapt to the needs of different niches and industries as shown from Figure 2.6 to 2.8. It provides project template compatible with hybrid Scrumban approach, combines Scrum and Kanban to adapt to every user scenario as shown in Figure 2.9. Moreover, there a lot of scrum metrics graph and chart in Jira which can be used to monitor the project's health and risks as shown in Figure 2.12. It is genuinely flexible project management which actively promote team collaboration as it provides beneficial hint to team members in improving the Scrum progress. To illustrate this point, Jira pushes notifications to team members and recommends things they might be interested in, promote communication, sharing ideas amongst team members. Jira is a rich toolkit for developers. Figure 2.14 displays the Jira Query Language (JQL), which is used to create all filters and manipulations efficiently. Besides, it can be connected to Bitbucket and GitHub to provide backlog-to - deployment traceability, and synchronization with build, test and release activities on the application as shown in Figure 2.13. In addition, its usage can be further extended with flexible add-ons, plugins, REST and JAVA Apis. Figure 2.15 shows the Jira's mobile application in Android and IOS platform that allows remote teams to work on different location.

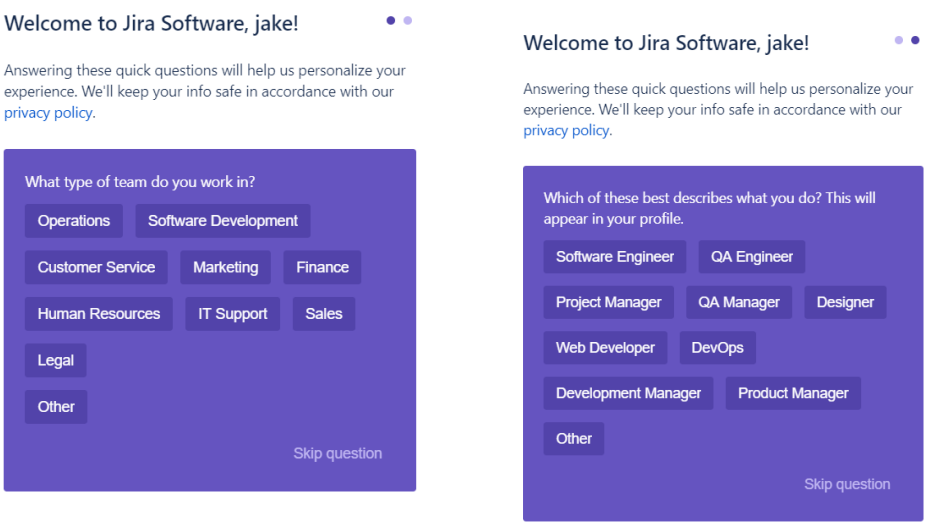


Figure 2.6: Select various Project type and Project role in a new Jira Project

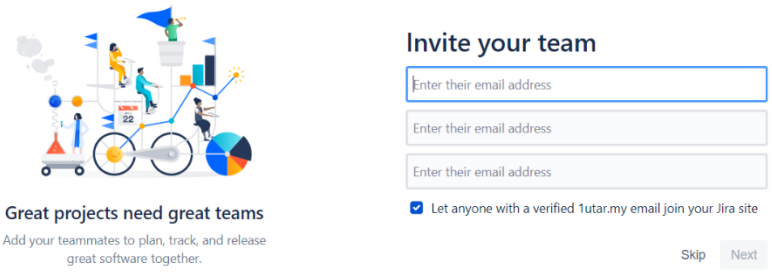


Figure 2.7: Jira project invitation function

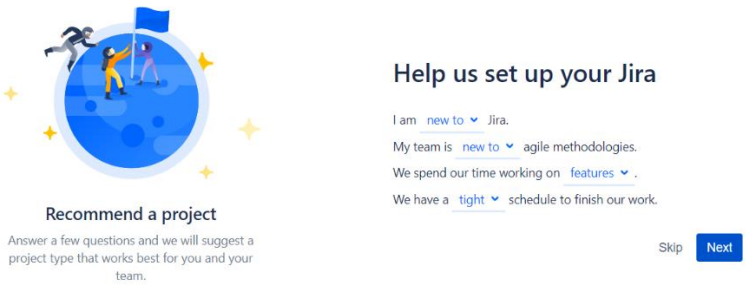


Figure 2.8: Jira smart project configuration

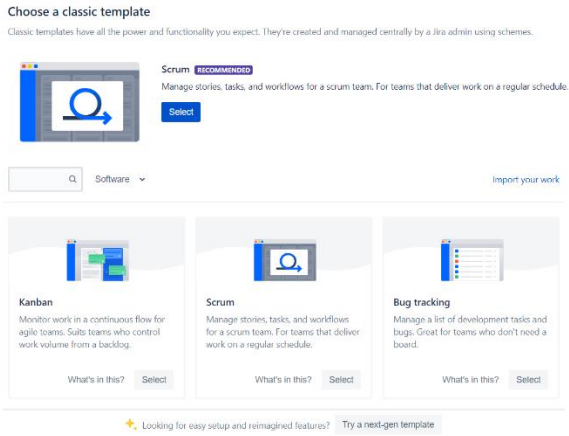


Figure 2.9: Choosing a project template

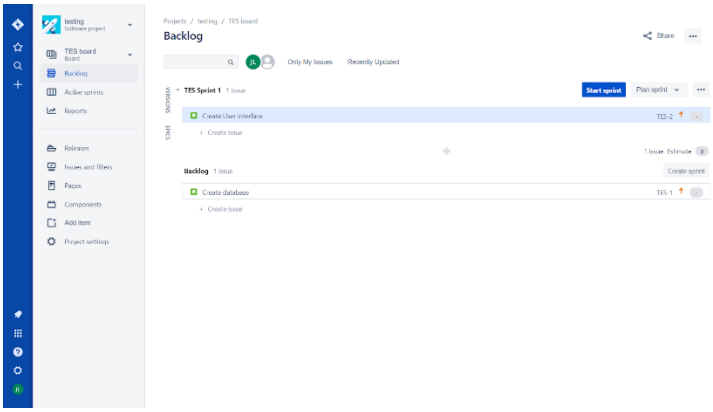


Figure 2.10: Product and sprint backlog management system of Jira

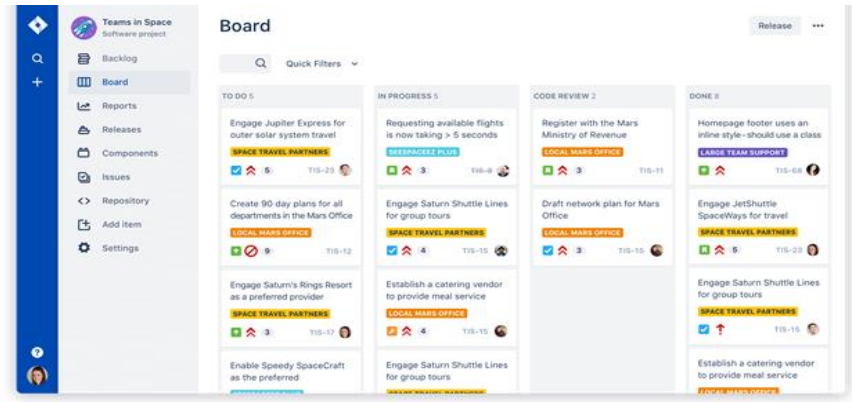


Figure 2.11: Jira Scrum Board

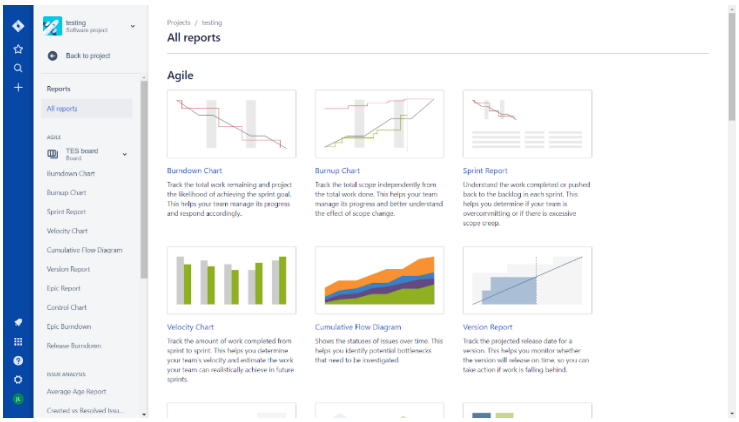


Figure 2.12: Scrum Metrics in Jira

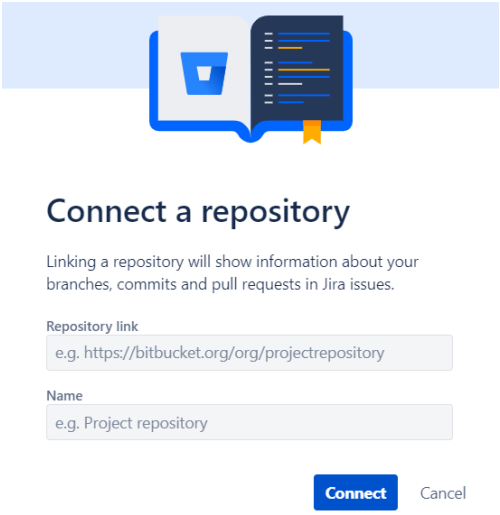


Figure 2.13: Connecting project to third party repository in Jira

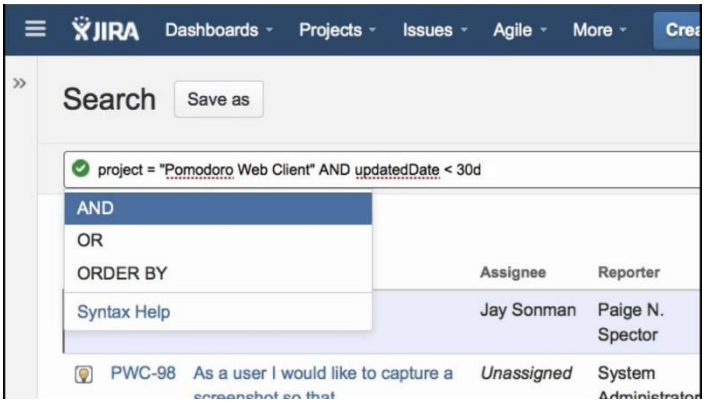


Figure 2.14: Jira Query Language (JQL)

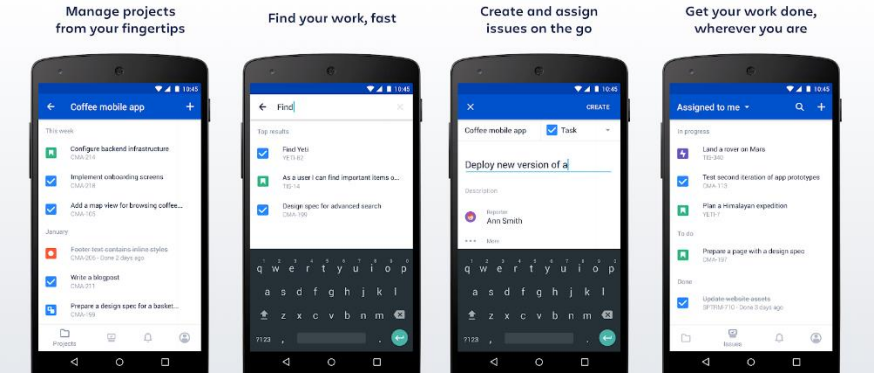


Figure 2.15: Jira Mobile Application

2.5.2 Yodiz

Figure 2.16: Yodiz Logo

Yodiz is a simple but comprehensive agile tool to manage Agile projects of all levels of complexity. Yodiz has similar functions with Jira. It supports product backlog management, Kanban / ScrumBan / Scrum management. It also offers extensive reporting capabilities like Scrum metrics such as Team Velocity graph. Moreover, it also helps in maximize efficiency, increase visibility and ultimately deliver faster by integrating with third-party applications Continuous Integration Tools (CI) like Jenkins, GitHub, etc. Most importantly, it also has its own associated mobile application. The only difference is it is much more affordable for small project teams as it has lesser functionalities compared to Jira. Yodiz has better Scrum board tasks filtration compared to Jira. Figure 2.17 shows the Scrum Board layout and navigation bar of Yodiz software.

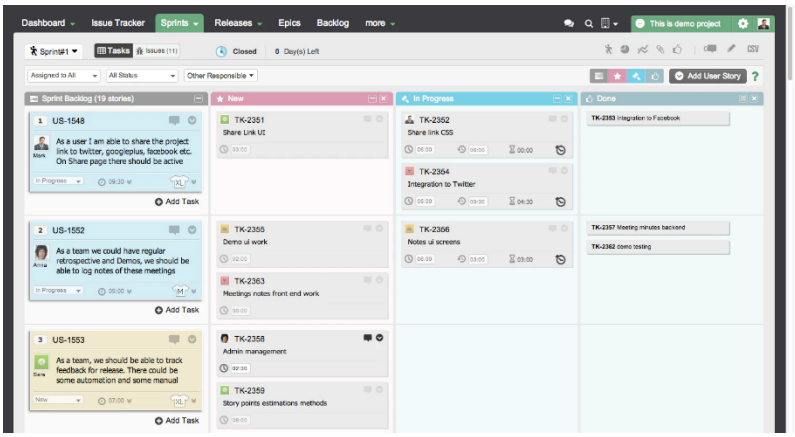


Figure 2.17: Yodiz Scrum Board

2.5.3 Scrumwise

Figure 2.18: Scrumwise Logo

Scrumwise is the most basic Scrum Board application tool for Scrum beginners without any need for training. It has the similar extensible function similar to Jira and Yodiz. The main difference is its straightforward as a scrum tool for backlog management, iteration management, progress tracking and task management. It has the simplest visual display which contributes a lot in its ease of use. Thus, it is highly recommended to new Scrum practitioners. Moreover, user can export all data for easy ad-hoc analysis and reporting (export to CSV File or XML file) in Scrumwise. Figure 2.19 shows the Scrum Board layout and navigation bar of Yodiz software.

Figure 2.19: Scrumwise Scrum Board

2.5.4 Why some existing Scrum Board applications are unsuitable for Scrum beginners

- **Heavy weight and complex layout**

  Heavyweight Scrum Board applications such as Jira are suitable for professional Scrum users whose have experience in either small or big Scrum project and undergone proper training. Usability of these applications could be very complex and sophisticated for new Scrum users. Due to variation of functions, the user interface and the workflow for each task are confusing for first users. As consequences, they will have higher failure rate in carrying out even the most fundamental function of Scrum tool. Furthermore, they cannot fully utilise the extensible functionalities of these applications which might lead to unnecessary project costs. To illustrate this point, creating a backlog item in Jira is difficult for new Scrum users without proper guidance as shown in Figure 2.19. Since both collection of sprint backlog items and backlog items are integrated together, the overall layout is very complex. The workflow of creating a new backlog item is convoluted likewise.

Figure 2.20: Creating a backlog item without the create icon

- **Subscription fee**

  Most of the existing Scrum Board applications are freemium. Scrum beginner can only access to these applications during a short trial length and sometimes come with other constraints such as limited numbers of projects, or the need to register with personal billing information. When the trial length has ended, the Scrum beginner has to pay subscription fee monthly in order to keep his project alive.

2.5.5 Summary

Table 2.1 concludes the comparison between the addressed existing Scrum Board applications. Scrumwise is none the than the most suitable Scrum tools which can be

easily used by Scrum beginners who are also new in project management. To overcome Scrumwise's shortcomings, professional scrum tools such as Jira, Yodiz or others existing applications could be used as a reference or a guide in developing current project's Scrum applications.

Table 2.2: Comparison between Jira, Yodiz, Scrumwise

	Jira	**Yodiz**	**Scrumwise**
Product Backlog, Sprint and Burndown charts	Yes	Yes	Yes
Visualisation	Yes	Yes	Yes
Reports	Yes	Yes	Yes
Notifications	Yes	Yes	Yes
Integration (mobile and other PM tools)	Yes	Yes	No
Board configuration and filters usage complexity	High	High	Intermediate
Performance in handling complex nested tasks	Intermediate	Intermediate	Low
Project planning properties	High	Intermediate	Low
Pricing	$10 /user/month	$3/user/month	$7.50/user/month
Overall complexity	High	Intermediate	Low

The basic functions of a Scrum tool are shown as below:

- Product Backlog, Sprint Backlog, User Stories Management
- Creation of Sprint
- Customizable Scrum Board with swim lanes separate Sprint activities into three phases which are “To-Do”, “In Progress” and “Done” by default
- Creation of Sprint activity card with configurable settings such as:
    - Developer who responsible for the activity
    - Due date of the activity
    - Description for the activity
    - Remark for the activity
    - Attachment and documents
    - Priority
    - Colours to determine its category

    And other addition functions such as:
    - Git Integration, CI integration and others third parties’ applications
    - Dashboard with open issues
    - Visualisation of burndown chart, team velocity and other Scrum Metrics

2.6 Project Methodology

This section delivers the research made in determining the most suitable software methodology for this project.

2.6.1 Traditional methodology vs Agile methodology and Intro to Hybrid

The PMI’s broad definition of project management methodology is helpful - ‘A methodology is a system of practices, techniques, procedures and rules used by those who work in a discipline’ (Muslihat, 2018a) . Traditional methodology uses a linear approach where stages of software development process must be completed in sequential order. It is heavyweight and exhausting. Developers have to anticipate all of the system requirements to be done before planning of how the system will work. This conscientious way of planning documents includes effort in obtaining project artifacts through complete and accurate reports, documents and UML diagrams often overtire the development team (Nishijima and Dos Santos, 2013). This is the reason that agile development methodologies are introduced. Agile methodologies are lightweight and best suited for iterative and incremental project. It can meet the

demands of the highly competitive and constant movement of the software industry (Muslihat, 2018b). It is also a useful tool to tackle complex and uncertain project situations.

Table 2.3: A Comparison between Traditional Framework and Agile Framework (John, 2013)

Traditional Framework	Agile Framework
Customer get involved early in the processes but tended to keep at arm's length once the project begins	Customer involvement is the key here as more closely involved at the time of work being performed
Escalation to managers when problem arise	When problems occur, team works together resolve them internally
Heavy upfront analysis and design	Daily stand-up meetings are held to discuss work done yesterday, plan of today and impediments if any
More serious about processes than the product	Agile methods less focus on formal and directive processes
Product is planned extensively and then executed and tested	Work is delivered to client in small and frequent releases to get rapid feedback loops
Traditional models favors "Anticipation"	Agile model favors "Adaption"

Table 2.2 provides an overview of comparison between Traditional and agile methodology framework. Traditional methodologies such as waterfall model forces structure organization and suited for milestone - focused development. It might not be flexible as adaptive and agile methodologies such as Extreme Programming, Scrum, Kanban method which provides a clear and measurable structure that fosters iterative development, team collaboration and change recognition. Each of these methodologies approaches has its own pros and cons, developers usually having tough time to choose from. This is where hybrid methodology is introduced. According to conference report of Chilean Computer Science Society in 2011 (Carvalho et al., 2012), there is a 16 percent increase in productivity in projects developed using the hybrid process. Although most of the studies are conducted with small size, medium size project, it still proved that productivity is increased and requirements specifications, architecture

and test cases are partially preserved with more stringent project scope. In conclusion, Hybrid process is potential to get real development productivity gain and it is now and the future.

2.6.2 Can a lone developer use Agile?

Due to many drawbacks of traditional method, agile is encouraged as it has less bureaucracy and strike a balance between having no defined process (for software development) and having a too prescriptively defined process (Akpata and Riha, 2004). importantly, there is less documentation and more adaptable.

The truth of the works of lone developer involve a regular number of meetings with the client and focus a lot in developing the product that meet all requirements as the client proposed. Documents and forms are usually replaced by these real- time communication between client and developer to minimize the risk of project overhead. There isn't an exact project management methodology for One- Person projects or speaking of impossibility to lone developers who use agile in their project.

All projects have different scope, deliverables, objectives and resources. Neagu (2013) said that any methodologies can be applied as long as the project requirements can be achieved. For instance, a Scrum method is theoretically unusable for lone developer when there is no Scrum Master or Product Owner available. However, developer can volatile the Scrum methodology original requirements and tailor it for the specific project. Utilise the methodology along with work plan such as Gantt chart with milestone, a work breakdown structure is essential for each project’s success. Once the planning stage is completed, stick to work plan and provide timely feedback to stakeholders.

2.6.3 Agile methodologies comparison

The phases of agile development cycle are flexible and always evolving at project advance. It is comprised of planning, requirement analysis, Design, Implementation & coding or development, testing and deployment. Once the project scope and objectives are deemed viable and feasible, developers can start to identify features and prioritize them before assigning it into an iteration. An overview of agile development cycle is shown in Figure 2.20.

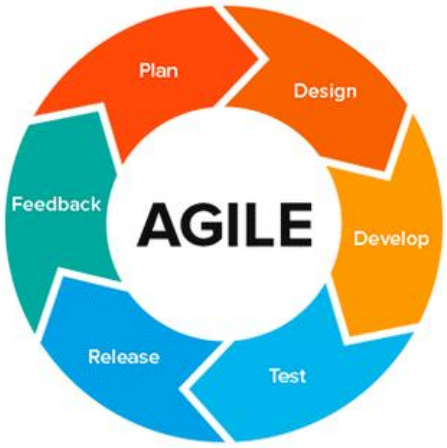


Figure 2.21: The agile development life cycle (Pino, 2018)

Agile is a framework and there are a number of methods within the Agile movement. There are:

- **Extreme Programming (XP):**

  Doing development robustly to ensure quality

  Pros:

- Improve quality and ensure responsiveness to evolving customer requirements
- Principles very similar to scrum
- Focus on technical practices around designing code and testing
- Apply Test - driven development (TDD), pair programming and continuous integration
- Documentation is kept to the barest minimum
- Facilitate daily stand up meetings just like Scrum
- Experiment technologies such as development software and tools

  Cons:

- A lot of short releases / version of product
- Simple layout design and solution, might miss out the details of features
- Inadequate documentations
- Effectiveness depends on the people involved
- Exact possibilities and future outcomes are really unknown

- **Scrum:**

Available to small, cross functional, self-managing team

Pros:

- Emphasize collaboration between scrum team roles
- Maintain Scrum Artifacts to keep track project's transparency and progress
- Developing incrementally by using time box constraints in each development cycle as known as sprint
- Review meeting is held to discover methods to improve the current product in next sprint

Cons:

- Requires highly expert Scrum team
- Not suitable for big-sized projects
- Heavy budget and time in Scrum process

- **Lean:**

Streamlining and eliminating waste to deliver more with less

Pros:

- Lower budget & time requirements
- Allows for delivery of product early
- Maximise continuous improvement and eliminating wastage with three principles: Muda, Mura and Muri

Cons:

- Workability of the team decides success of software development process
- Excessive flexibility leads developer to lose focus

And other methods such as Feature Driven Development (FDD), Dynamic System Development Method (DSDM), Kanban and etc.

2.6.4 Challenges of Agile methodologies

Adoption of agile methodologies will likely pose several challenges. It is really difficult to make the project evolve to the best balance of agile and plan-driven methods that fits its current situation. Delcheva (2017) clearly shows the productivity of the project is affected by:

(i) Communication challenges

Collaboration between team members through communication is critically important. The lack of communication between the project actors leads to project failure (Delcheva, 2017). Communication between the management and development team is relied on their proficiency in social skills (Delcheva, 2017). Messages and ideas are not clearly conveyed through project presentation and might cause huge misunderstanding and lead to conflicts. Language barrier and informal speaking also bring a huge impact to the productivity of the team.

(ii) Leadership challenges

In agile development, the management focuses on leadership and collaboration within projects (Gunga, Kishnah and Pudaruth, 2013). A good manager work towards empowering the team, consolidate the team working spirit. A leader with a bad mindset will lead to disorganisation of team structure and cause unnecessary problems that delay the delivery of the product.

There are also other significant challenges in agile method such as the workability of the development team. Engagement with stakeholders and users or managing project resources like product backlog depends a lot on skill set possessed by the developer.

2.6.5 Personal Extreme Programming – An agile process for autonomous developers

Personal Extreme Programming (PXP) is a modification through combination of Extreme Programming (XP) and Personal Software Process (PSP) which is explicitly specified to be used by individual engineers (Dzhurov, Krasteva and Ilieva, 2009). It is having similar core principles of XP which preaches the values of community, simplicity, feedback and courage for rapidly developing high quality in the fastest way possible. It explains the reason that It is the most suitable method for start-ups and to deal with certain circumstances where a system has to be developed in a time constraint. It will be more effective to deal with frequent changes of requirements.

The main difference between PXP and XP is the number of developers is involved in the project. XP allow a pair of programmers to work collaboratively where

one person produces the code and the other constantly reviews his work. PXP is the modified version of XP so that it can fit in a lone programmer situation.

PXP is an approach tries to balance between “too light” and “too heavy” methodologies (Umphress and Agarwal, 2008). PXP process is iterative and includes some inline iterations and cycles. Requirements and tasks planning are revised simultaneously in iteration. Documentation regarding the requirements analysis, improvement suggestions and defects details are maintained and presented at the end of iteration.

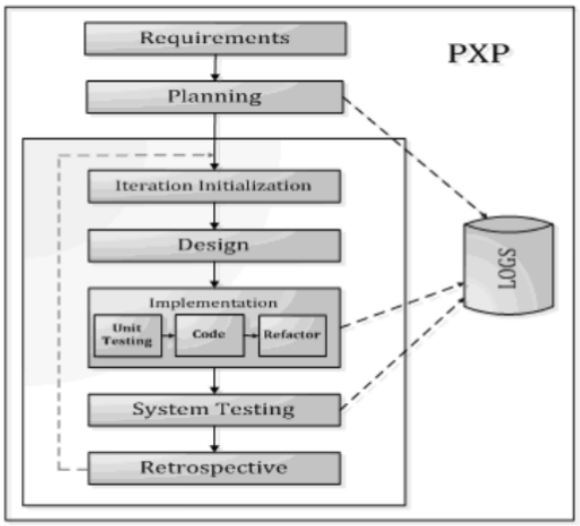


Figure 2.22: PXP process phases (Dzhurov, Krasteva and Ilieva, 2009)

The best part of process phases of PXP is developer keeps everything simple and aim to design the system to meet only the current client requirement without trying to make guesses what would be required in the future. PXP has slightly better planning effectiveness due to the greater number of tasks being defined – it is easier to better estimate a set of smaller tasks compared to a single big task (Dzhurov, Krasteva and Ilieva, 2009). This really saves a lot of time for developer to carry out sufficient testing to increase the reliability of the products. Besides, all code and design are developed by the same person, with lesser interruption or conflicts as no pair programming is implemented. PXP adhere sustainable pace for the project, developer tends to pursuit for excellence in great condition without getting overworked due to work culture (Akpata and Riha, 2004). PXP optimizes the development process by focusing on test driven development which allows defects to be prevented during implementation. As Dzhurov, Krasteva and Ilieva (2009 , p.258) point out that the number of defects found

in the PXP phase is significantly lower compared to ad-hoc development as illustrated in figure 2.22.

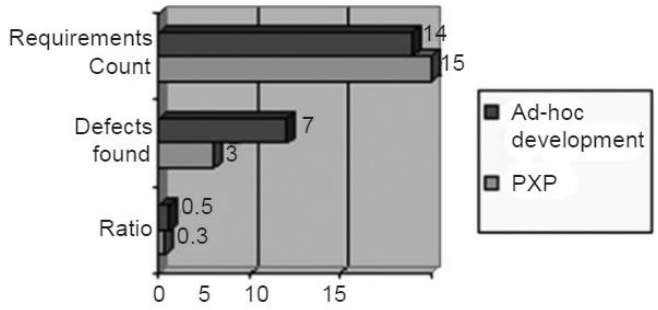


Figure 2.23: Ration between found defects and requirements in both project phases (Dzhurov, Krasteva and Ilieva, 2009)

2.6.6 Summary

Table 2.4 shows an overview of the comparison between some of most common -used Agile methodologies.

Table 2.4: A Comparison between Agile methodologies (Madadipouy, 2015)

Parameters	Scrum	DSDM	Crystal	ASD	XP
Process type	Several releases	Not specified	Incremental paces/iteration	Incremental paces/iteration	Several releases/iterative
Process period	30 days term/release	Not specified	Not specified	Not specified	Two weeks term/iterative
Focus	Self organization of team members and prioritize the requirement	Higher acceptance probability of changes Use 80:20 roles	People has the most influence on software quality It looks to adjust for every project separately	Address issues such as social, cultural and team skills	Communication Simplicity Courage Feedback
Development speed	Fast	Very fast	Fast	Fast	Very fast
Type of projects	Short-terms	Small/medium	Not specified	Large/complex	Small/large/complex Short/long term
Special feature	15 min daily meeting	Dynamic development Involve prototyping method	It allows agile team to select the most suitable method	Human collaboration Team self organization	It is suitable for a small team's workforce (5 to 15) team members) Maximizing communication which leads to enhancing team-work Pair programming
Additional features	Flexible, Adaptable, Empirical	Flexible, fast, collaborative	Fast, collaborative	Rapid, iterative	Flexible, using coding standard, simple design, planning game

Personal Extreme Programming (PXP) is a software development process for a single person team. It is based on the values of Extreme Programming (XP) i.e. simplicity, communication, feedback, and courage (Umphress and Agarwal, 2008). It works by keeping the important aspects of XP and refining the values so that they can fit in a lone programmer situation (Umphress and Agarwal, 2008). PXP is adopted because it is tailored and designed for lone developer usage than other available agile

method (Dzhurov, Krasteva and Ilieva, 2009) . Moreover, the current project has limited resources and time which cause difficulty in wrapping other agile methods together with PXP. Based on one of PXP methodology , developers should measure, track and analyse their daily work (Umphress and Agarwal, 2008) . This is one of the subsidiary reasons of adopting PXP in this project when it is compulsory to update the logbook twice a week for supervisor's monitoring job. PXP focuses on test driven development which saves a lot of time during testing. There are approximately 85 percent business logic code coverage by unit tests. There are other ongoing projects are happening concurrently with the current project might exhaust the developer easily. PXP is encouraged as it utilizes "40 hours a week" principle to ensure that developer is not overtired for the next iteration (Umphress and Agarwal, 2008).

2.7 Project Development Tools

This section delivers an introduction and explanation about project development tools for web application development and mobile application development.

2.7.1 Mobile Application Development

2.7.1.1 React Native Firebase

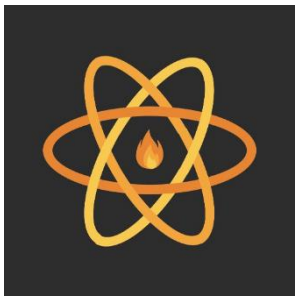


Figure 2.24: React Native Firebase Logo

React Native Firebase is an integration of React Native and Firebase. React native is a framework created by Facebook for building native mobile application just using JavaScript and React. It uses native component to create UI, along with JSX. Its coding style is quite similar to web development, which makes it simple to understand and be learnt. The React Native allows its codebase to be run on both IOS and Android platform, which requires less resources when there is no need to develop both separate

platforms individually (Trnka, 2018). Using React Native also greatly speed up the development process and maintenance process especially toward a project with limited time and resources like this. There are a lot of React Natives modules that can be installed directly on the application to perform specific feature Moreover, React Native has fast build time thanks to Hot Reloading and Live Reloading functions compared to the tedious and long build time of native application development (Trnka, 2018).

Figure 2.25: React native paper Logo

Figure 2.26: React native vector icons Logo

The front-end development of the mobile application would be completed using React native paper (shown in Figure 2.25), a high quality, standard compliant Material Design library which makes the react native application look and feel native. Moreover, React native Vector icons (shown in Figure 2.26) package is also used in this project.

2.7.1.2 Firebase

Figure 2.27: Firebase Logo

Firebase is a freemium Backend -as -a Service (BaaS) platform, created by Google. It is a user-friendly platform which allows to build a web / mobile application without server-side programming language such as PHP (Patel, 2016). Firebase provides real time database which sync data in user devices rapidly and accurately. It also provides cloud storage and user authentication. It also provides hosting service for web application. Figure 2.25 displays an overview of Firebase's functions.

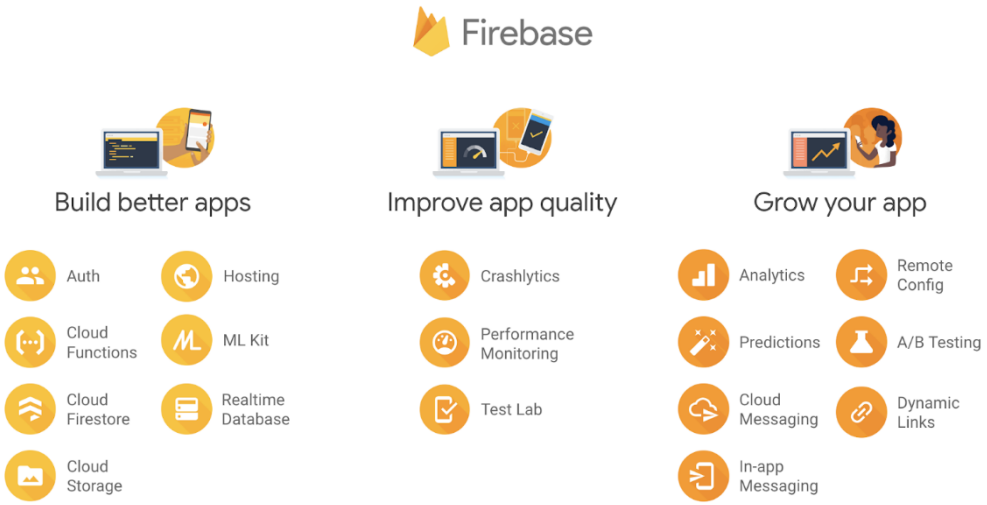


Figure 2.28: Complete functions of Firebase (Singh, 2018)

The main reason of implementing Firebase in this project is to save time and reduce the risk of having errors the back end of both applications. Besides, there are very few easy to configure and free hosting service on the internet. There is a significant difference in the pricing if compare Firebase with 000WebHost or Surge. Firebase is most suitable and easier to be managed Platform as a service (PAAS) for freelancer projects.

		SPARK Free Generous limits for hobbyists	FLAME $25 per month Predictable pricing for growing apps	BLAZE Pay as you go Commodity pricing for apps at scale
Included Free Analytics, App Indexing, Authentication, Cloud Messaging, Crash Reporting, Dynamic Links, Invites, Notifications & Remote Config		✓	✓	✓
Realtime Database	Simultaneous connections	100	Unlimited[1]	Unlimited[1]
	GB stored	1 GB	2.5 GB	$5/GB
	GB downloaded	10 GB	20 GB	$1/GB
	Automated backups	✗	✗	✓
Storage	GB stored	5 GB	50 GB	$0.026/GB[2]
	GB downloaded	30 GB	50 GB	$0.12/GB[2]
	Uploads & downloads	50,000 of each	100,000 of each	$0.01/K[2]
Hosting	GB stored	1 GB	10 GB	$0.026/GB
	GB downloaded	10 GB	50 GB	$0.15/GB
	Custom domain hosting & SSL	✓	✓	✓
Test Lab	Daily quota or hourly rate	15 tests per day[3]	15 tests per day[3]	hourly, per device[4]
Google Cloud Platform	Use BigQuery & other IaaS[5]	✗	✗	✓

Figure 2.29: Firebase pricing (ATIMI, n.d.)

2.7.2 Web Application Development

2.7.2.1 React

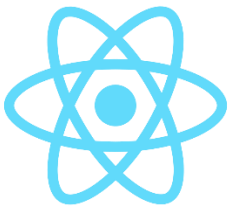


Figure 2.30: React.js Logo

Front- end web development using React.js is easy to learn when you have basic knowledge in JavaScript or about React Native. This project uses React.js because it boosts productivity and facilitates further maintenance by reusing components. Moreover, React.js ensures faster rendering with Virtual Dom which used to support heavy loaded and dynamic software solutions. While the performance is guaranteed, the code is stabilized. There are other React libraries that can be used to sooth the development process such as:

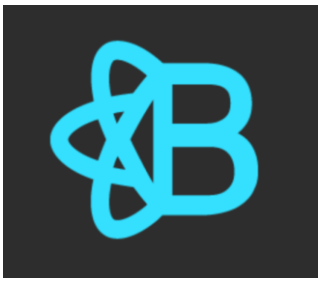


Figure 2.31: React Bootstrap

React bootstrap (shown in Figure 2.31) library is used for front end development. It provides basic UI components and its grid system allows developers to produce a responsive website. Hence, it is simple, eye catching, light and user friendly. Why not use Laravel? Laravel focuses more on back end scripting which is not really being utilised in this project as Firebase has been used.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

Why is a software development methodology so important? It is a framework that is used to structure, plan and control the process of developing a good quality maintainable software within a reasonable time and at an affordable cost. Every software methodology has its own software development life cycle (SDLC) to create a high-quality software. Figure 3.1 shows all stages involved in a fundamental SDLC.

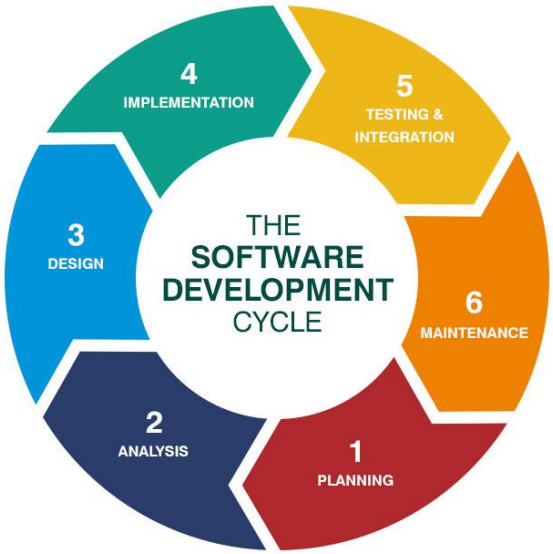


Figure 3.1: Software Development Life Cycle (SDLC) (Singh, 2016).

This chapter explains the methodology and work plan for this project. The methodology of this project. Section 3.1 explains the purpose of other sections in this chapter. Section 3.2 is about an overview of the Gantt Chart and its milestones of the project. Section 3.3 describes why the project methodology is adopted. Section 3.4 narrates the project's work breakdown structure. Section 3.5 briefly describes the application of Trello Board in this project.

3.2 Gantt Chart

Weeks	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13
Tasks	27/5 - 2/6	3/6 - 9/6	10/6 - 16/6	17/6 - 23/6	24/6 - 30/6	1/7 - 7/7	8/7 - 14/7	15/7 - 21/7	22/7 - 28/7	29/7 - 4/8	5/8 - 11/8	12/8 - 18/8	19/8 - 25/8
1.Project Preparatory													
1.1 Simple Research on interested proposal title													
1.2 Approach designated Supervisor													
1.3 Registration of FYP title													
2. Project Planning and Defining													
2.1 Project initiation documentation													
2.1.1 Define Project Goal													
2.1.2 Define Project Background													
2.1.3 Define Project Problem Statement													
2.1.4 Define Project Objectives													
2.1.5 Define Project Solution													
2.1.6 Define Project Approach													
2.1.7 Define Project Scope													
2.2 Project Scheduling													
2.2.1 Schedule tasks and workflow using Trello													
2.2.2 Create Gantt Chart													
2.2.3 Create Work Breakdown Structure													
3. Literature Review and Analysis													
3.1 Research on concept of Scrum Board													
3.2 Research on Scrum Methodology													
3.3 Review on existing applications													

3.4 Making conclusion for all research													
4. Requirement Analysis and Gathering													
4.1 Gather Requirement from stakeholders													
4.2 Analyse requirements													
4.3 Clarify requirements with stakeholders													
5. Software Design													
5.1 UI layout design for applications													
5.1.1 Web application UI design													
5.1.2 Mobile application UI design													
5.2 System Design													
5.2.1 Create use case diagram													
5.2.2 Create data flow diagram													
6. Research on Software Methodology and Development tools													
6.1 Research on Project's adopted software methodology													
6.2 Research on project development tools													
7. Prototype Development													
7.1 Revaluate requirements													
7.2 Build the prototype with prototyping tools / high level language													
7.3 Evaluate the prototype													
8. Preparation for Presentation													
8.1 Prepare presentation slides and script													
8.2 Rehearsal for presentation													
8.3 Revaluate the prototype													

Figure 3.2: Project 1's Gantt Chart

Weeks	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13
Tasks	14/1 - 19/1	20/1 - 26/1	27/1 - 2/2	3/2 - 9/2	10/2 - 16/2	17/2 - 23/2	24/2 - 1/3	2/3 - 8/3	9/3 - 16/3	23/3 - 29/3	30/3 - 5/4	6/4 - 12/4	13/4 - 19/4
1. Development of Scrum Board Web application													
Iteration 1: Login, Register and Forgot Password													
1.1 Construct Login Screen UI & develop its feature													
1.2 Construct Register Screen UI & develop its feature													
1.3 Construct Forgot Password Screen UI & develop its feature													
1.4 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
1.5 Review the test result and do proper refactoring to the code and UI													
Iteration 2: Project Dashboard and Project Management													
2.1 Construct Project Dashboard UI & develop its feature													
2.2 Construct Project Management UI & develop its feature (CRUD)													
2.3 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
2.4 Review the test result and do proper refactoring to the code and UI													
Iteration 3: Product Backlog Management													

3.1 Construct Product Backlog UI & develop its implementation (CRUD)													
3.2 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
3.3 Review the test result and do proper refactoring to the code and UI													
Iteration 4: Sprint Backlog and Sprint Management													
4.1 Construct Sprint Backlog UI & develop its feature (CRUD)													
4.2 Construct Sprint UI & develop its feature (CRUD)													
4.3 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
4.4 Review the test result and do proper refactoring to the code and UI													
Iteration 5: Archive Management													
5.1 Construct Archive UI & develop its feature (CR)													
5.2 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
5.3 Review the test result and do proper refactoring to the code and UI													
Iteration 6: Customizable Scrum Board													
6.1 Construct Customizable Scrum Board UI & develop its feature (CRUD)													
6.2 Develop additional filter feature													

6.3 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
6.4 Review the test result and do proper refactoring to the code and UI													
Iteration 7: PDF and Excel Generator													
7.1 Implementation of generate PDF reports based on different data (product backlog, sprint backlog, sprint, archive)													
7.2 Implementation of generate Excel reports based on different data (product backlog, sprint backlog, sprint, archive)													
7.3 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
7.4 Review the test result and do proper refactoring to the code													
Iteration 8: Backend API s													
8.1 Implementation of Backend APIs with Firebase Admin													
8.2 Host the Backend express app on Heroku													
8.3 Test the APIs with Postman													
8.4 Review the test result and do proper refactoring to the code													
Iteration 9: Feed and notification System													
9.1 Construct Feed UI & develop its feature (CR)													
9.2 Setup OneSignal notification third party service													

9.3 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
9.4 Review the test result and do proper refactoring to the code													
2. Development of Scrum Board Web application													
Iteration 10: Login, Register and Forgot Password													
10.1 Construct Login Screen UI & develop its feature													
10.2 Register Screen UI & develop its feature													
10.3 Forgot Password Screen UI & develop its feature													
10.4 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
10.5 Review the test result and do proper refactoring to the code and UI													
Iteration 11: Project Dashboard													
11.1 Construct Project Dashboard UI & develop its feature													
11.2 View Project Details UI & develop its feature (R)													
11.3 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
11.4 Review the test result and do proper refactoring to the code and UI													
Iteration 12: User Sprint Task Management													
12.1 Construct Sprint Task Management UI & develop its feature (RU)													

12.2 Construct Unassigned Sprint Task Management UI & develop its feature (RU)													
12.3 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
12.4 Review the test result and do proper refactoring to the code and UI													
Iteration 13: Scrum Board Overview													
13.1 Construct Scrum Board Overview UI & develop its feature (R)													
13.2 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
13.3 Review the test result and do proper refactoring to the code and UI													
Iteration 14: Feed and notification system													
14.1 Construct Feed UI & develop its feature (CR)													
14.2 Setup OneSignal notification third party service													
14.3 Test the implemented functions (Unit Test, Usability Test, User Acceptance Test)													
14.4 Review the test result and do proper refactoring to the code													
Iteration 15: Backend API s													
15.1 Implementation of Backend APIs with Firebase Admin													
15.2 Test the APIs with Postman													

15.3 Review the test result and do proper refactoring to the code													

Figure 3.3: Project 2's Gantt Chart

Figure 3.2 and 3.3 gives an overview of milestones listed in Gantt Chart of Project 1 and Project 2 which can be summarised as:

3.2.1 Explanation of the Milestones of the project

In Project 1, we have

- **Project Preparatory**

  For every beginning of a project, one must know what kind of project he can do and whether he acquires the right skills and attributes. Most importantly, interest is the most important key that sustains passion and endeavour which eventually drives the success of the project. In Week 1, portion of researches need to be done onto a few of project titles that are captivating and worth the effort. Then, discussion is further carried out with the supervisor who hold the utmost responsibility of these titles. Complete all the procedure of registration of FYP title after your appointment with your supervisor has been authenticated and you are self-convinced to achieve the selected project title.

- **Project Planning and Defining**

  Define the project's main objectives, problem statement (limitations and assumptions) and its solution (strategy). The proposal also includes a brief introduction onto background of the project and provides an opportunity to express your views and ideas of current situation and future state of the project. Identify project 's critical information with the 5ws which are:

  - Who - who are the stakeholders and end users?
  - What - what is the project scope and stakeholder expectations?
  - When - when will it be done and needed to update your stakeholders?
  - Where - where will be the application deploy?
  - Why - why do you take up this project

  Nonetheless and sixth question:

  - How - How will you solve the problem of the project?

Project scheduling and organize activities by mapping out and group their task if necessary. Find each task dependencies and continuity. Make proper estimation on the time and effort it will take to complete them. Moreover, decision - making is pivotal in this duration. This whole process takes about 2 to 5 weeks. Create a draft of WBS and Gantt chart during this phase.

- **Literature Review and Analysis**

  Perform additional research to understand the project title more thoroughly and explore articles and journals related to problems stated in the project and strategies to tackle them, directly or indirectly. Analyse the resources gathered and check their reliability and validity. Conduct an online survey and interview by any means necessary. Moreover, review on existing application to get know into usage of Online Scrum Board. Reconfirm and update your supervisor frequently during the research. The whole process takes about 2 to 5 weeks and usually carried out concurrently with project scheduling to save time.

- **Requirement Analysis and Gathering**

  Gather requirements from stakeholders or through conduction of online survey and interview with end users by any means necessary within 1 week. Utilise the following weeks to execute tasks such as interpret requirements into user stories, determine functional and non - functional requirements more specifically, clarify the requirements with stakeholders and etc. If the project has contingency of having its requirements change frequently, it is suggested to elongate the duration of this phase in favour of methodology used to deliver the product.

- **Software Design**

  Requirements are the functions and expected outputs of the system. With the requirements gathered previously and researches on existing application in literature review, define the architecture or structure of the system in high level to low level view. Design the user interface of the application following Schneiderman's "Eight Golden Rules of Interaction Design" and get reference to user interface design of existing applications. Identify use cases and

workflow of the applications, then create a Use Case Diagram and a Data Flow diagram.

- **Research on Software Methodology and Development tools**

  List out how do you plan your work plan and execute it along with the methodology used in this project. Besides, explain the adoption of project methodologies and its research in the literature review section. Specify the details of how each task is performed and the tools used during the process. Discover and associate development tools that will be used in this project and give proper explanation with research in the literature review section.

- **Prototype Development**

  Develop a prototype iteratively until it has fulfilled the requirements. Revaluate the design and workflow of the system in the development process. The functions of prototype are built incrementally following the software development methodology. Acceptance Testing is performed by the stakeholders to meet project goals. Fix every bug and inconveniences of the prototype before moving on to the next function of the prototype. This usually takes the longest time, which possibly in between 7 to 12 weeks.

- **Preparation for Oral Presentation**

  Prepare the presentation slides and script within 3 weeks of time. Visualisation and number of words used in each slide of content is important. Revaluate the prototype and understand its workflow after the presentation slides were done.

In Project 2, we have

- **Iterative development**

  The project is being developed iteratively and incrementally with new feature. Each feature and its requirements are reclarified before the iteration starts. Each iteration includes the tasks of constructing the layout, develop the functional system of the feature and last but not least, revaluate and perform testing. The testing includes acceptance testing from stakeholders especially supervisor.

3.3 Adopted Development Methodology

The methodology that will be used in this project is PXP, which implement an iterative and agile to develop the Scrum Board system. These are some of the main reasons that PXP is implemented:

1. **Tight Deadline of Project 1 & Project 2**

   Implementing a prototype for autonomous developer is difficult when there are other projects going on concurrently. The other projects require lot of effort and time in planning which eventually lead to a lot of exhaustive works especially analysing research materials.

2. **Workability and lack of coding skills**

   Capability of bringing the project to success depends heavily on the workability and skills acquired for one. I do not master my programming skills and techniques very well especially when it comes to handling back end functionalities and configurations of the system. Moreover, my productivity will also be decreased because of lack of fundamental knowledge in software framework. It will be a huge waste of time in understanding and relearning the prowess of this project. Most importantly, contribution of lone developer toward the project is inadequate to meet the deadlines.

3. **Simplicity**

   PXP contrived simplicity in project planning, design and documentation while concentrating on iterative and incremental development of the final product. Spending a lot of time in complicating the documentation process is extraneous.

4. **Heavy communication and feedback**

   This progress of this project relies heavily on the stakeholder. The main stakeholder of this project holds the utmost responsibility in monitoring and providing continuous feedback to improve the Scrum Board system. An active collaboration between she and me is needed. Stand up meetings are held once a week to ensure all the requirements of the project is met and everything is running smoothly as expected. Shared decision-making will also be promoted using PXP.

3.4 Project Breakdown

The development of Scrum Board applications will be divided into 3 phases:

1. **Exploration**

- Gather user stories, requirements and perform analysis
- Create simple layout design and workflow of each system's function
- Gather insights from existing applications
- Update literature review by any necessary means

2. **Planning**

- Schedule activities and tasks
- Analyse methodology, tools used in this project
- Simple architecture design of the system
- Write complete proposal for this project

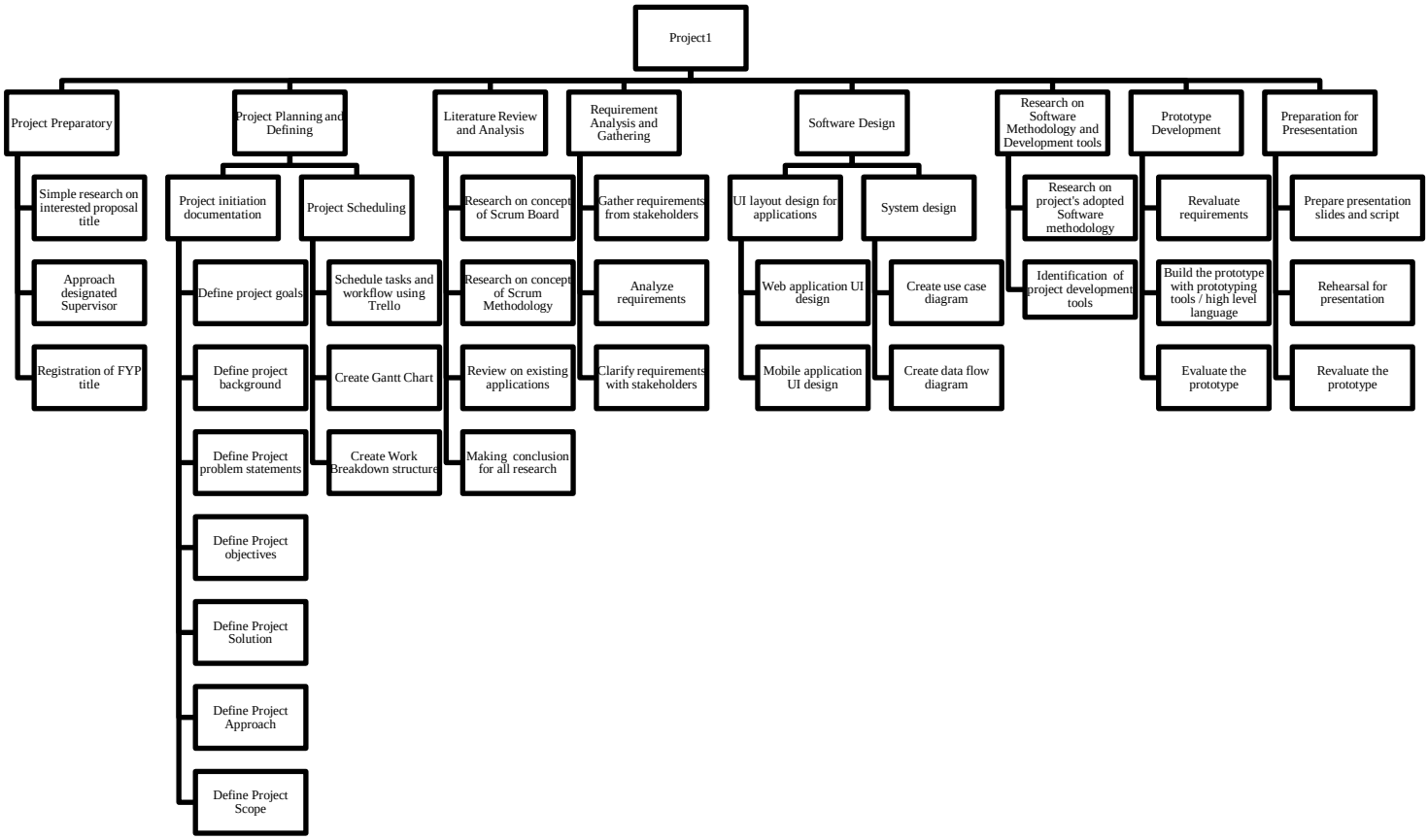
3. **Iterative Development**

- Requirement Gathering
    - Updates the latest requirements
- Design
    - Constructing the layout of application
- Implementation
    - Implement server side and client-side coding
- Testing
    - Perform user acceptance testing
    - Modify and refine the code
    - Perform system testing
- Retrospective
    - Gather feedback from stakeholder
    - List out a simple work plan and activities for the next iteration

Exploration and planning phases will be executed during Project 1. Development of prototype in Project 1 session will take at least 2 to 3 iterative development phases. Project 2 will focus more on Iterative development phase.

3.5 Work Breakdown Structure (WBS)

Figure 3.4: Project1's WBS



Project 1 tasks are break down into:

1 Project Preparatory

- 1.1 Simple research on interested proposal title
- 1.2 Approach designated Supervisor
- 1.3 Registration of FYP title

2 Project Planning and Defining

- 2.1 Project initiation documentation
  - 2.1.1 Define Project Goals
  - 2.1.2 Define Project Background
  - 2.1.3 Define Project Problem statements
  - 2.1.4 Define Project Objectives
  - 2.1.5 Define Project Solution
  - 2.1.6 Define Project Approach
  - 2.1.7 Define Project Scope
- 2.2 Project Scheduling

2.2.1 Scheduling tasks and workflow using Trello

2.2.2 Create Gantt chart

2.2.3 Create Work Breakdown Structure (WBS)

3 Literature Review and Analysis

3.1 Research on concept of Scrum Board

3.2 Research on Scrum Methodology

3.3 Review on existing applications

3.4 Making conclusion for all researches

4 Requirement Analysis and Gathering

4.1 Gather requirements from stakeholders

4.2 Analyse requirements

4.3 Clarify requirements with stakeholders

5 Software Design

5.1 UI layout design for applications

5.1.1 Web application UI design

5.1.2 Mobile application UI design

5.2 System Design

5.2.1 Create Use Case diagram

5.2.2 Create Data Flow diagram

6 Research on Software Methodology and Development tools

6.1 Research on Project's adopted software methodology

6.2 Research on Project's development tools

7 Prototype Development

7.1 Revaluate requirements

7.2 Build the prototype with prototyping tools / high level language

7.3 Evaluate the prototype

8 Preparation for Presentation

8.1 Prepare presentation slides and script

8.2 Rehearsal for presentation

8.3 Revaluate the prototype

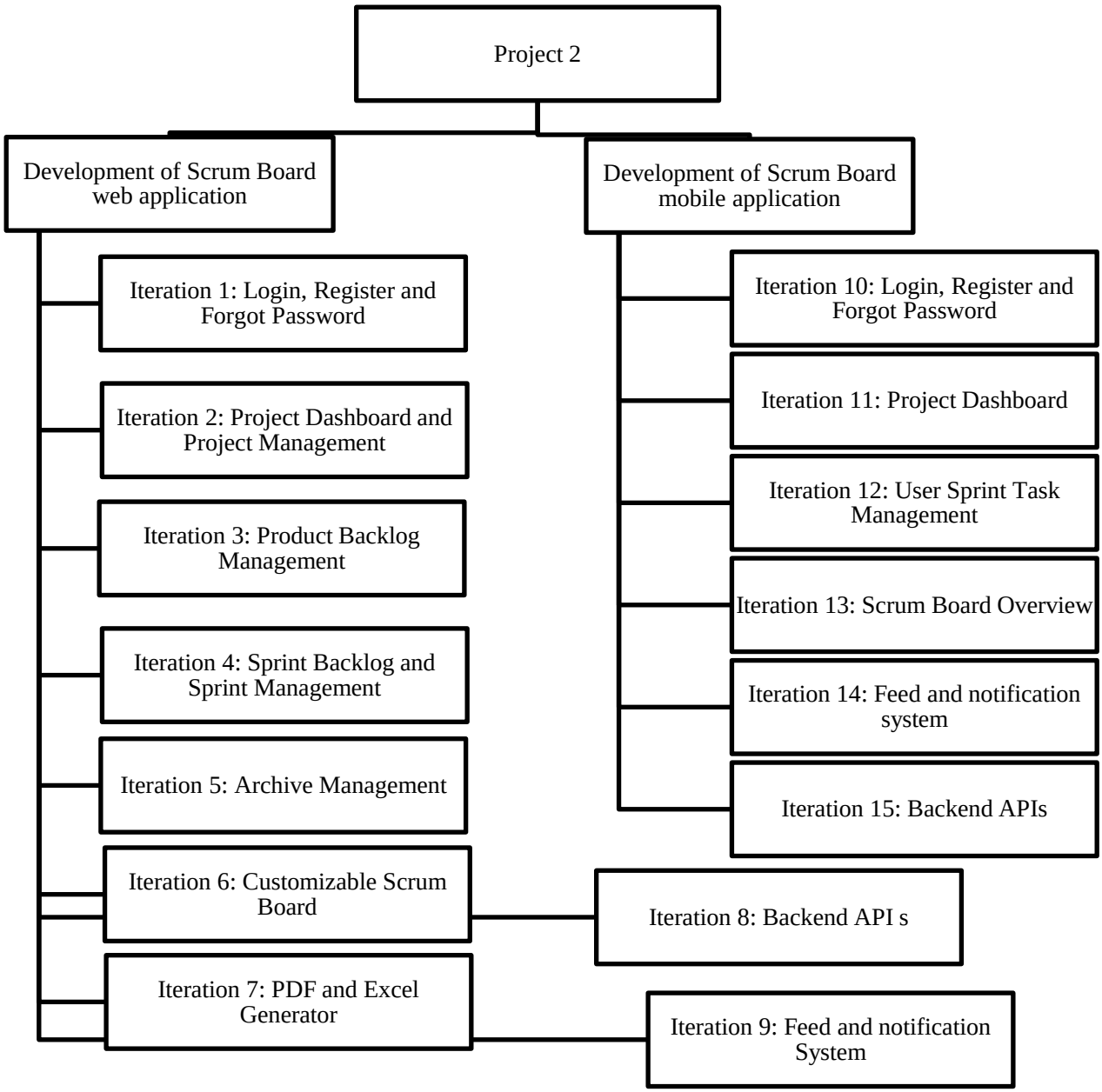


Figure 3.5: Project2’s WBS

Project 2 tasks are break down into:

1 Development of Scrum Board Web application
    1.1 Iteration 1: Login, Register and Forgot Password
    1.2 Iteration 2: Project Dashboard and Project Management
    1.3 Iteration 3: Product Backlog Management
    1.4 Iteration 4: Sprint Backlog and Sprint Management
    1.5 Iteration 5: Archive Management
    1.6 Iteration 6: Customizable Scrum Board
    1.7 Iteration 7: PDF and Excel Generator
    1.8 Iteration 8: Backend API s
    1.9 Iteration 9: Feed and notification System
2 Development of Scrum Board mobile application
    2.1 Iteration 10: Login, Register and Forgot Password
    2.2 Iteration 11: Project Dashboard
    2.3 Iteration 12: User Sprint Task Management
    2.4 Iteration 13: Scrum Board Overview
    2.5 Iteration 14: Feed and notification system
    2.6 Iteration 15: Backend APIs

3.6 Trello - a project board tool

Trello is a project management software that enables users to organize their tasks via boards, lists, cards and other productivity tools. The main advantage of Trello is its uncomplicated and user-friendly interface. In this project, Trello is used as the collaborative platform between developer and stakeholder. It delivers a simple overview of the project plan.

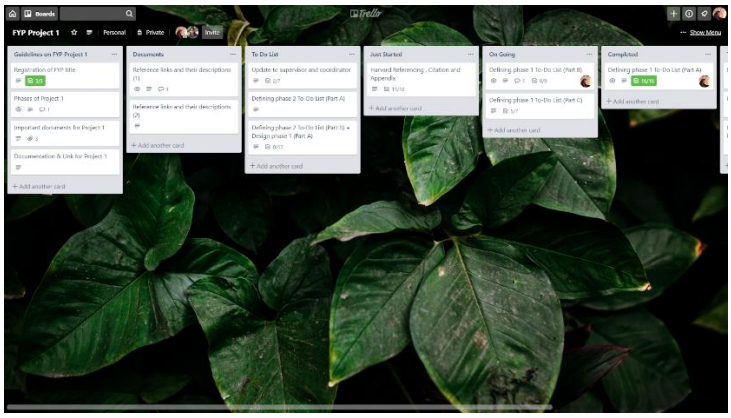


Figure 3.6: FYP Project 1 Trello Board

During Project 1, there are 7 lists. There are two lists describing the official guidelines of FYP project, references used in this project. The others are “To-Do lists”, “Just Started”, “On Going”, “Completed” which describes the progress of the project activities. Last but not least, there is a list called “To be completed” describes a list of ideas of creating new tasks to improve the final product.

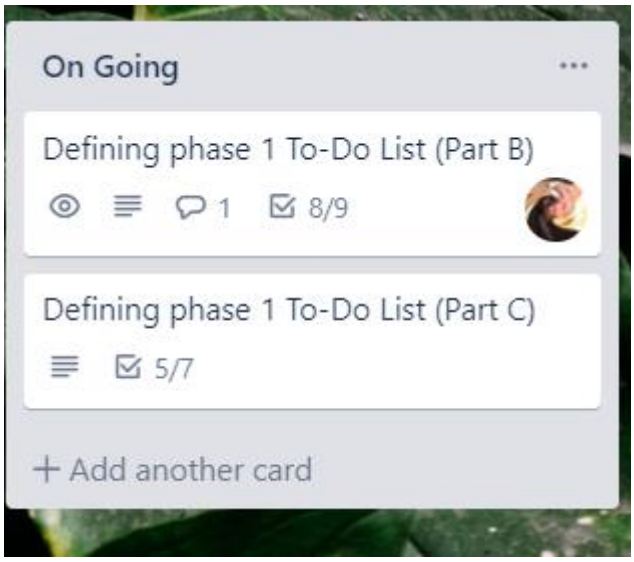


Figure 3.7: Activities on a list

Figure 3.5 sketches out several cards of activities in a list. Each card describes multiples of tasks should be done in an individual project activity as shown in Figure 3.6 below. Each card is given unique names and tasks are scheduled and grouped following WBS scheme.

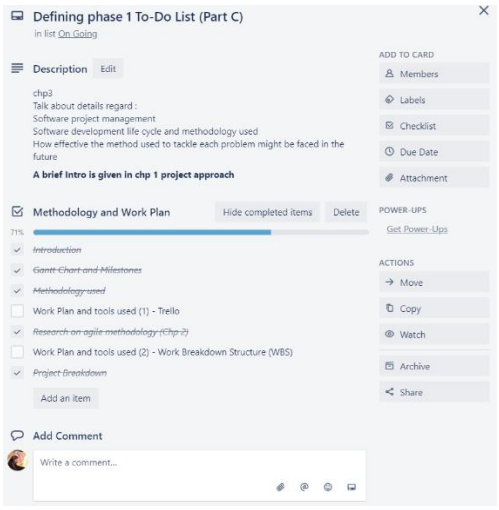


Figure 3.8: Tasks of an activity card

CHAPTER 4

PROJECT SPECIFICATION

4.1 Introduction

This chapter explains the project initial specification of this project. The methodology of this project. Section 4.1 lists out an overall analysis report on the requirement specification gathered from the stakeholders. Section 4.2 describes the use case diagram and its description. Section 4.3 illustrates the use case diagram and attach with its all the use case description. Section 4.4 illustrates the context diagram while Section 4.5 illustrates the data flow diagram. Section 4.6 explains the system architecture. Section 4.7 narrates the system database design. Section 4.8 the initial system design for the prototype.

4.2 Requirement Specifications

The requirements specifications were collected and evaluated directly from the supervisor of this project during consultation hour. Efforts were made in requirement analysis to further elaborate the functional requirements of Scrum Board application in both web and mobile platform.

4.2.1 Functional Requirements for Scrum Board web application

These are the compulsory functional requirements for Scrum Board web application:

1. Scrum Board web application should allow user to manage list of sprint backlogs item.
2. Scrum Board web application should allow user to manage list of product backlogs item.
3. Scrum Board web application should allow user to manage sprints of the project.
4. Scrum Board web application should provide an overview of Scrum Board with swim lanes of tasks of current sprint by default.
5. Scrum Board web application should allow user to view other archived sprints Scrum Board.
6. Scrum Board web application should allow user to view archive of sprint and sprint backlogs.
7. Scrum Board web application should allow user to manage multiple projects.

8. Scrum Board web application should display notifications and project feeds.
9. Scrum Board web application should allow user to generate a variety of reports regarding on sprint, backlog items or activities in excel, pdf extension.
10. Scrum Board web application should allow user to invite or remove new or existing Scrum team members to or from a project.
11. Scrum Board web application should allow existing user to login and new user to register.
12. Scrum Board mobile application should allow existing user to reset password.

Optional functional requirements:

1. Scrum Board web application shall allow user to upload documents and pictures in activity card.
2. Scrum Board web application shall allow user to post comments in activity card.
3. Scrum Board web application shall display Scrum Metric like Burndown chart.
4. Scrum Board web application shall allow user to manage profile.
5. Scrum Board web application shall display history log of updates and changes.
6. Scrum Board web application shall allow user to login or register with their GitHub account.
7. Scrum Board web application shall have filter function and search function.
8. Scrum Board web application shall provide tutorial.

4.2.2 Functional Requirements for Scrum Board mobile application

These are the compulsory functional requirements for Scrum Board mobile application:

1. Scrum Board mobile application should allow user to view all the projects that he / she is participates.
2. Scrum Board mobile application should allow user to update the status of his/ her given sprint task.
3. Scrum Board mobile application should provide an overview of Scrum Board with swim lanes of tasks of current sprint by default.
4. Scrum Board mobile application should display notifications and activity feeds.
5. Scrum Board mobile application should allow user to view all his/her sprint tasks.

6. Scrum Board mobile application should allow user to view all unassigned sprint tasks.
7. Scrum Board mobile application should allow user to get unassigned sprint tasks.
8. Scrum Board mobile application should allow existing user to login and new user to register.
9. Scrum Board mobile application should allow existing user to reset password.

Optional functional requirements:

1. Scrum Board mobile application shall allow user to upload documents and pictures in activity card.
2. Scrum Board mobile application shall allow user to post comments in activity card.
3. Scrum Board mobile application shall allow user to manage profile.
4. Scrum Board mobile application shall allow user to login or register with their GitHub account.
5. Scrum Board mobile application shall have filter function and search function.
6. Scrum Board mobile application shall provide tutorial.

4.2.3 Non-functional requirements for Scrum Board applications

These are few non-functional requirements which are meant to be achieved in the Scrum Board applications.

Security:

1. The applications shall restrict unauthorized user to access the system.
2. The applications shall be able to back up the data time to time to prevent any loss of data.

Availability

1. The applications shall be available to access at anytime and anywhere as long as user is able to connect to the internet.

Performance

1. The applications shall be able to handle concurrent requests from multiple users without crashing.

2. The applications shall be able to show the result to the user at a specific time range without delaying.

Usability

1. The applications shall be user-friendly by having a simple and consistent user interface which are able to let user to have a better experience and have the feeling of ‘ease to use’, especially towards new Scrum user.
2. The user interface of the applications shall be designed to let user to have a better control.
3. The user interfaces of the applications shall be able to guild the user to achieve their objectives.
4. The applications shall be able to display error message when the input data is irrelevant or inaccurate.

4.3 Use Case Diagram

Use case Diagrams are commonly used to describe requirements and desired functionality of software products (Grechanik, McKinley and Perry, 2007). It also describes user scenario such has how users interact with system components to perform desired operations. With the functional requirements elicited from the stakeholders, a use case diagram for each Scrum Board application was created.

4.3.1 Use Case Diagram of Scrum Board Web Application

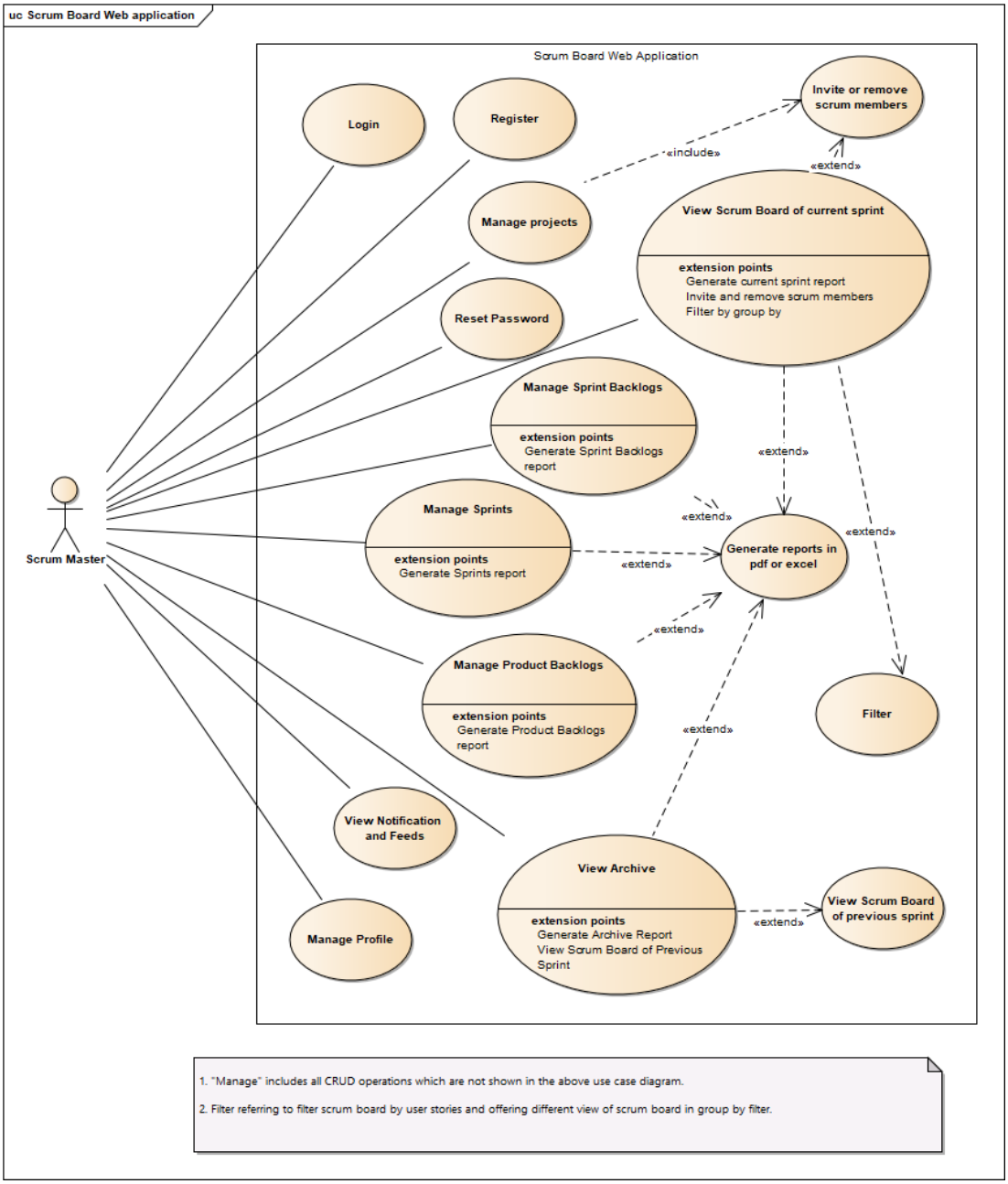


Figure 4.1: Use Case Diagram of Scrum Board Web Application

4.3.2 Use Case Description of Scrum Board Web Application

Table 4.1: Login use case description (scrum board web application)

Use case ID	1
Use case name	Login
Actor	Scrum Master
Description	Scrum Master login into Scrum Board web application using email address and password.
Preconditions	
Postconditions	
Flow of events: 1. Scrum Master enters the correct email address and password. 2. Scrum Master clicks on the Login button. 3. Scrum Master successfully login to the application and will be directed to the project dashboard.	
Alternative flow of events: 1.1 email and password are not found 1.1.1 The application displays an error message indicating the credentials entered are invalid 1.2 email or password are wrong 1.2.1 The application displays an error message indicating the credentials entered are invalid. 1.3 Wrong format of email 1.3.1 The application displays an error message indicating Scrum Master to enter a valid email-address.	

Table 4.2: Register use case description (scrum board web application)

Use case ID	2
Use case name	Register
Actors	Scrum Master
Description	Scrum Master register a new account of Scrum Board web application.
Preconditions	
Postconditions	
Flow of events: 1. Scrum Master enters a unique username, email, password. 2. Scrum Master clicks on the Register button. 3. A new account has been successfully created and the user will be directed to the project dashboard.	
Alternative flow of events: 1.1 Wrong format of email 1.1.1 The application displays an error message indicating Scrum Master to enter a valid email-address.	

Table 4.3: Reset Password use case description (scrum board web application)

Use case ID	3
Use case name	Reset Password
Actors	Scrum Master
Description	Scrum Master forgot his / her password and wish to reset password for his / her account.
Preconditions	
Postconditions	
Flow of events: 1. Scrum Master enters the email address of his / her account. 2. A recovery email will be sent into his / her inbox for further instruction to reset the password.	

Alternative flow of events: 1.1 Wrong format of email 1.1.1 The application displays an error message indicating Scrum Master to enter a valid email-address.

Table 4.4: Manage projects use case description (scrum board web application)

Use case ID	4
Use case name	Manage projects
Actors	Scrum Master
Description	Scrum Master manage multiples Scrum projects concurrently in Scrum Board web application.
Preconditions	Scrum Master login successfully to the Scrum Board web application.
Postconditions	
Flow of events: 1. Scrum Master manage multiple projects.	
Alternative flow of events: 1.1 Create New Project 1.1.1 Scrum Master clicks on the Add Project button. 1.1.2 Scrum Master enters the information of a new project includes title, caption, description, vision, start date, estimate date. 1.1.3 Scrum Master invite new scrum team members into this new project by by registering their emails. 1.1.4 Scrum Master clicks on the Create button 1.1.4 A new project is created. 1.2 View Project 1.2.1 Scrum Master select one of the projects and directed to an overview of the selected project's information. 1.3 Edit Project 1.3.1 Scrum Master select one of the projects and directed to an overview of the	

project. 1.3.2 Scrum Master clicks on the edit button will be directed to edit project page. 1.3.3 Scrum Master make changes and click on Save button to save the changes. 1.3.4 The selected project is edited and updated. 1.4 Delete Project 1.4.1 Scrum Master select one of the projects and directed to an overview of the project. 1.4.2 Scrum Master clicks on the delete button. 1.4.3 The selected project has been successfully deleted. 1.5 Close Project / Reopen Project 1.5.1 Scrum Master select one of the projects and directed to an overview of the project. 1.5.2 For a close project, Scrum Master clicks on the reopen button to activate the closed project. For an active project, Scrum Master clicks on the close button to close the project, 1.5.3 The selected project has been successfully closed / reopen depends on the action made above.

Table 4.5: Manage Product Backlogs use case description (scrum board web application)

Use case ID	5
Use case name	Manage Product Backlogs
Actors	Scrum Master
Description	Scrum Master manage product backlogs of a project in Scrum Board web application.
Preconditions	Scrum Master login successfully to the Scrum Board web application. Scrum Master select one of the projects.

	Scrum Master is in backlog management page.
Postconditions	

Flow of events:

1. Scrum Master manage list of Product Backlogs.

Alternative flow of events:

1.1 Create a Product Backlog item

1.1.1 Scrum Master clicks on the Create New Product Backlog Item button.

1.1.2 Scrum Master enters the information of a new product backlog item includes backlog code, backlog title, description, priority, type, story point, completion date, status

1.1.3 Scrum Master clicks on the Save Changes button

1.1.4 A Product Backlog item has been successfully created.

1.2 View a Product Backlog item

1.2.1 Scrum Master select one of the product backlog items and it will show an overview of the selected product backlog item's information.

1.3 Edit a Product Backlog item

1.3.1 Scrum Master select one of the product backlog items.

1.3.2 Scrum Master clicks on the edit button.

1.3.3 Scrum Master make changes and click on Save button to save the changes.

1.3.4 The selected product backlog item is edited and updated.

1.4 Delete a Product Backlog item

1.4.1 Scrum Master select one of product backlog items.

1.4.2 Scrum Master clicks on the delete button.

1.4.3 The selected product backlog item has been successfully deleted.

Table 4.6: Manage Sprint Backlogs use case description (scrum board web application)

Use case ID	6
Use case name	Manage Sprint Backlogs
Actors	Scrum Master
Description	Scrum Master manage sprint backlogs of a project in Scrum Board web application.
Preconditions	Scrum Master login successfully to the Scrum Board web application. Scrum Master select one of the projects. Scrum Master is in sprint management page.
Postconditions	
Flow of events: 1. Scrum Master manage list of Sprint Backlogs.	
Alternative flow of events: 1.1 Create a Sprint Backlog item 1.1.1 Scrum Master clicks on the Create New Sprint Backlog Item button. 1.1.2 Scrum Master enters the information of a new sprint backlog item includes sprint backlog code, sprint backlog title, priority, type, status, assignee, watchers, due date, targeted sprint. 1.1.3 Scrum Master clicks on the Save Changes button 1.1.4 A Sprint Backlog item has been successfully created. 1.2 View a Sprint Backlog item 1.2.1 Scrum Master select one of the sprint backlog items and it will show an overview of the selected sprint backlog item's information. 1.3 Edit a Sprint Backlog item 1.3.1 Scrum Master select one of the sprint backlog items. 1.3.2 Scrum Master clicks on the edit button. 1.3.3 Scrum Master make changes and click on Save button to save the changes.	

1.3.4 The selected sprint backlog item is edited and updated. 1.4 Delete a Sprint Backlog item 1.4.1 Scrum Master select one of sprint backlog items. 1.4.2 Scrum Master clicks on the delete button. 1.4.3 The selected sprint backlog item has been successfully deleted.

Table 4.7: Manage Sprints use case description (scrum board web application)

Use case ID	7
Use case name	Manage Sprints
Actors	Scrum Master
Description	Scrum Master manage sprints of a project in Scrum Board web application.
Preconditions	Scrum Master login successfully to the Scrum Board web application. Scrum Master select one of the projects. Scrum Master is in sprint management page.
Postconditions	
Flow of events: 1. Scrum Master manage list of Sprints.	
Alternative flow of events: 1.1 Create a Sprint 1.1.1 Scrum Master clicks on the Create New Sprint button. 1.1.2 Scrum Master enters the information of a new sprint includes Sprint length, start date, sprint goal. 1.1.3 Scrum Master clicks on the Save Changes button 1.1.4 A Sprint has been successfully created. 1.2 View a Sprint 1.2.1 Scrum Master select one of the sprints and it will show an overview of the selected sprint's information.	

1.3 Edit a Sprint

1.3.1 Scrum Master select one of the open sprints.

1.3.2 Scrum Master clicks on the edit button.

1.3.3 Scrum Master make changes and click on Save button to save the changes.

1.3.4 The selected sprint is edited and updated.

1.4 Delete a Sprint

1.4.1 Scrum Master select one of the open sprints.

1.4.2 Scrum Master clicks on the delete button.

1.4.3 The selected sprint has been successfully deleted.

1.5 Start sprint

1.5.1 Scrum Master select one of the open sprints.

1.5.2 Scrum Master clicks on the start button.

1.5.3 The system will check if the selected sprint is consecutive sprint with correct sprint name and number.

1.5.4 The system will check if the selected sprint ‘s estimate due date is not expired.

1.5.5 The system will check if there is an ongoing sprint. If so, it will terminate the ongoing sprint.

1.5.6 The selected sprint is successfully started.

1.6 Close sprint

1.5.1 Scrum Master select the ongoing sprint.

1.5.2 Scrum Master clicks on the close button.

1.5.3 The selected sprint is successfully started.

Table 4.8: Generate reports in pdf or excel (scrum board web application)

Use case ID	8
Use case name	Generate reports in pdf or excel
Actors	Scrum Master
Description	Scrum Master can generate report for stand-up meetings or project management.
Preconditions	Scrum Master login successfully to the Scrum Board web application. Scrum Master select one of the projects. Scrum Master is in sprint management page / scrum board page / backlog management page / archive page.
Postconditions	
Flow of events: 1. Scrum Master clicks on the print button. 2. The application will show a drop-down list of items which displays the format of the report which are pdf and excel format. 3. Scrum Master selects one of the drop-down list items. 4. A report will be downloaded automatically.	
Alternative flow of events:	

Table 4.9: View Scrum Board of current sprint use case description (scrum board web application)

Use case ID	9
Use case name	View Scrum Board of current sprint
Actors	Scrum Master
Description	The Scrum Board web application displays the Scrum Board of the active sprint.
Preconditions	Scrum Master login successfully to the Scrum Board web application.

	Scrum Master select one of the projects. Scrum Master is in scrum board page.
Postconditions	
Flow of events: 1. The application displays scrum boards of the current sprint.	
Alternative flow of events: 1.1 Invite or remove scrum members 1.1.1 Scrum Master clicks on the Add Member button. 1.1.2 Scrum Master register or unregister scrum team member email. 1.1.3 Scrum Master clicks on the Save Changes button. 1.1.4 The scrum team members of the project is updated. 1.2 Filter 1.2.1 Scrum Master clicks on the Group By status to display swim lanes of sprint tasks in toDo , ongoing , complete , can be further filter with user stories. 1.2.2 Scrum Master clicks on the Group By story to display list of user stories with sprint tasks	

Table 4.10: View archive use case description (scrum board web application)

Use case ID	10
Use case name	View Archive
Actors	Scrum Master
Description	The Scrum Board web application displays list of closed sprints in the archive.
Preconditions	Scrum Master login successfully to the Scrum Board web application. Scrum Master select one of the projects. Scrum Master is in archive page.
Postconditions	
Flow of events: 1. The application displays list of archived sprints.	

Alternative flow of events: 1.1 View Scrum Board of previous sprint 1.1.1 Scrum Master select one of the archived sprints. 1.1.2 Scrum Master click on View Scrum Board button. 1.1.3 The system will display the scrum board of the selected archived sprint.

Table 4.11: View notifications and feeds use case description (scrum board web application)

Use case ID	11
Use case name	View notifications and feeds
Actors	Scrum Master
Description	Scrum Master will receive notifications on the updates from the Scrum Board mobile application and application displays them in a feed list.
Preconditions	Scrum Master login successfully to the Scrum Board web application.
Postconditions	
Flow of events: 1. The application will display notifications in the feed list.	
Alternative flow of events:	

Table 4.12: Manage Profile (scrum board web application)

Use case ID	12
Use case name	Manage Profile
Actors	Scrum Master
Description	Scrum Master manage the profile of his / her own account in Scrum Board web application.
Preconditions	Scrum Master login successfully to the Scrum Board web application. Scrum Master is in profile page.
Postconditions	
Flow of events: 1. Scrum Master manage the profile of his / her account in Scrum Board web application.	
Alternative flow of events: 1.1 Change Username 1.2 Change Profile Picture 1.3 Sign Out	

4.3.3 Use Case Diagram of Scrum Board Mobile Application

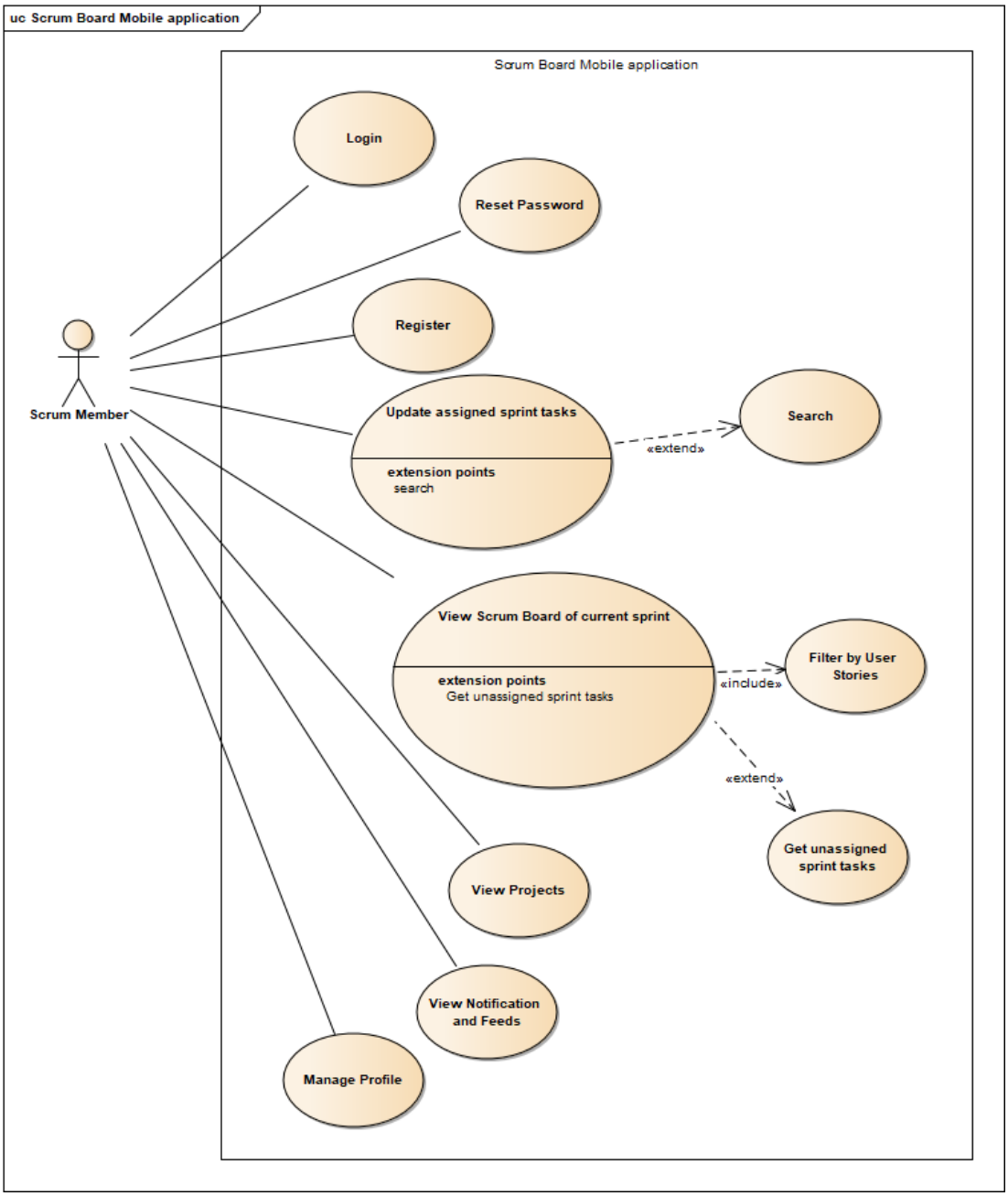


Figure 4.2: Use Case Diagram of Scrum Board Mobile Application

4.3.4 Use Case Description of Scrum Board Mobile Application

Table 4.13: Login use case description (scrum board mobile application)

Use case ID	1
Use case name	Login
Actor	Scrum Member
Description	Scrum Member login into Scrum Board mobile application using email address and password.
Preconditions	
Postconditions	
Flow of events: 1. Scrum Member enters the correct email address and password. 2. Scrum Member clicks on the Login button. 3.Scrum Member successfully login to the application and will be directed to the project dashboard.	
Alternative flow of events: 1.1Email and password are not found 1.1.1 The application displays an error message indicating the credentials entered are invalid 1.2 Email or password are wrong 1.2.1 The application displays an error message indicating the credentials entered are invalid. 1.3 Wrong format of email 1.3.1 The application displays an error message indicating Scrum Member to enter a valid email-address.	

Table 4.14: Register use case description (scrum board mobile application)

Use case ID	2
Use case name	Register
Actors	Scrum Member
Description	Scrum Member register a new account of Scrum Board mobile application.
Preconditions	
Postconditions	
Flow of events: 1. Scrum Member enters a unique username, email, password. 2. Scrum Member clicks on the Register button. 3. A new account has been successfully created and the user will be directed to the project dashboard.	
Alternative flow of events: 1.1 Wrong format of email 1.1.1 The application displays an error message indicating Scrum Member to enter a valid email-address.	

Table 4.15: Reset Password use case description (scrum board mobile application)

Use case ID	3
Use case name	Reset Password
Actors	Scrum Member
Description	Scrum Member forgot his / her password and wish to reset password for his / her account.
Preconditions	
Postconditions	
Flow of events: 1. Scrum Member enters the email address of his / her account. 2. A recovery email will be sent into his / her inbox for further instruction to reset the password.	

Alternative flow of events: 1.1 Wrong format of email 1.1.1 The application displays an error message indicating Scrum Member to enter a valid email-address.

Table 4.16: Update assigned sprint tasks use case description (scrum board mobile application)

Use case ID	4
Use case name	Update assigned sprint tasks
Actors	Scrum Member
Description	Scrum Member update the status of Sprint tasks that assigned by Scrum Master from Scrum Board web application.
Preconditions	Scrum Member has successfully login into Scrum Board mobile application. Scrum Member select one of the projects.
Postconditions	
Flow of events: 1. The application displays a list of currently assigned Sprint tasks of the active sprint of the selected project. 2. Scrum Member selects one of the sprint tasks. 3. Scrum Team member update the status of Sprint task.	
Alternative flow of events: 1.1 Search 1.1.1 Scrum Member can search for a specific assigned sprint task.	

Table 4.17: View notifications and feeds use case description (scrum board mobile application)

Use case ID	5
Use case name	View notifications and feeds
Actors	Scrum Member
Description	Scrum Member will receive notifications on the updates from the Scrum Board mobile application and application displays them in a feed list.
Preconditions	Scrum Member login successfully to the Scrum Board mobile application.
Postconditions	
Flow of events: 1. The application will display notifications in the feed list.	
Alternative flow of events:	

Table 4.18: Manage Profile (scrum board mobile application)

Use case ID	6
Use case name	Manage Profile
Actors	Scrum Member
Description	Scrum Member manage the profile of his / her own account in Scrum Board mobile application.
Preconditions	Scrum Member login successfully to the Scrum Board mobile application. Scrum Member is in profile page.
Postconditions	
Flow of events: 1. Scrum Member manage the profile of his / her account in Scrum Board web application.	

Alternative flow of events: 1.1 Change Username 1.2 Change Profile Picture 1.3 Sign Out

Table 4.19: View Scrum Board use case description (scrum board mobile application)

Use case ID	7
Use case name	View Scrum Board
Actors	Scrum Member
Description	Scrum Member can see an overview of swim lanes of sprint tasks of a sprint of the selected project in Scrum Board mobile application.
Preconditions	Scrum Member has successfully login into Scrum Board mobile application Scrum Member select one of the projects.
Postconditions	
Flow of events: 1. The application displays swim lanes of sprint backlog items (To-Do, In Progress, Completed) in a Scrum Board of the active sprint of selected project.	
Alternative flow of events: 1.1 Filter by User stories 1.1.1 Scrum Member click on the dropdown box to choose the user story. 1.1.2 The scrum board will be filtered according to the selected user story. 1.2 Get unassigned Sprint tasks 1.2.1 Scrum Member click on sprint task which do not have assignee. 1.2.2 Scrum Member click on Assign to Yourself button. 1.2.2 Scrum Member has been successfully assigned to the selected available task.	

4.4 Context Diagram

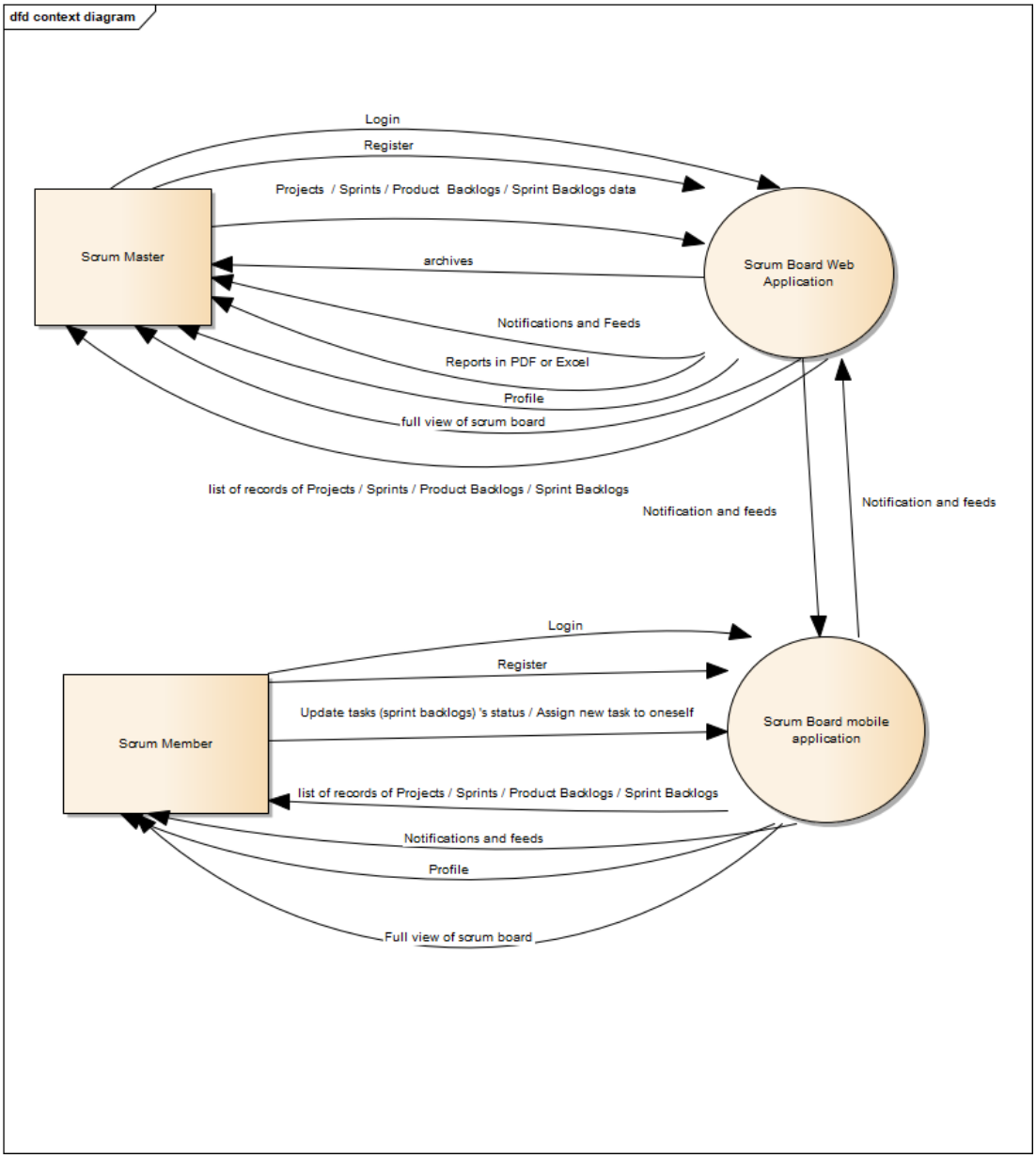


Figure 4.3: Context Diagram of Scrum Board Mobile Application and Scrum Board Web Application

Basically, scrum master performs all the CRUD operations in managing all sorts of data including projects, product backlogs, sprint backlogs and sprints on the scrum board web application. These feeds and notifications are pushed to the scrum board mobile application should that the scrum member can receive immediate updates. On the other hand, all actions that were done by scrum member on the scrum board mobile

application will generate feeds and notifications and pushed to the scrum board web application so that the scrum master is aware of it.

4.5 Data Flow Diagram

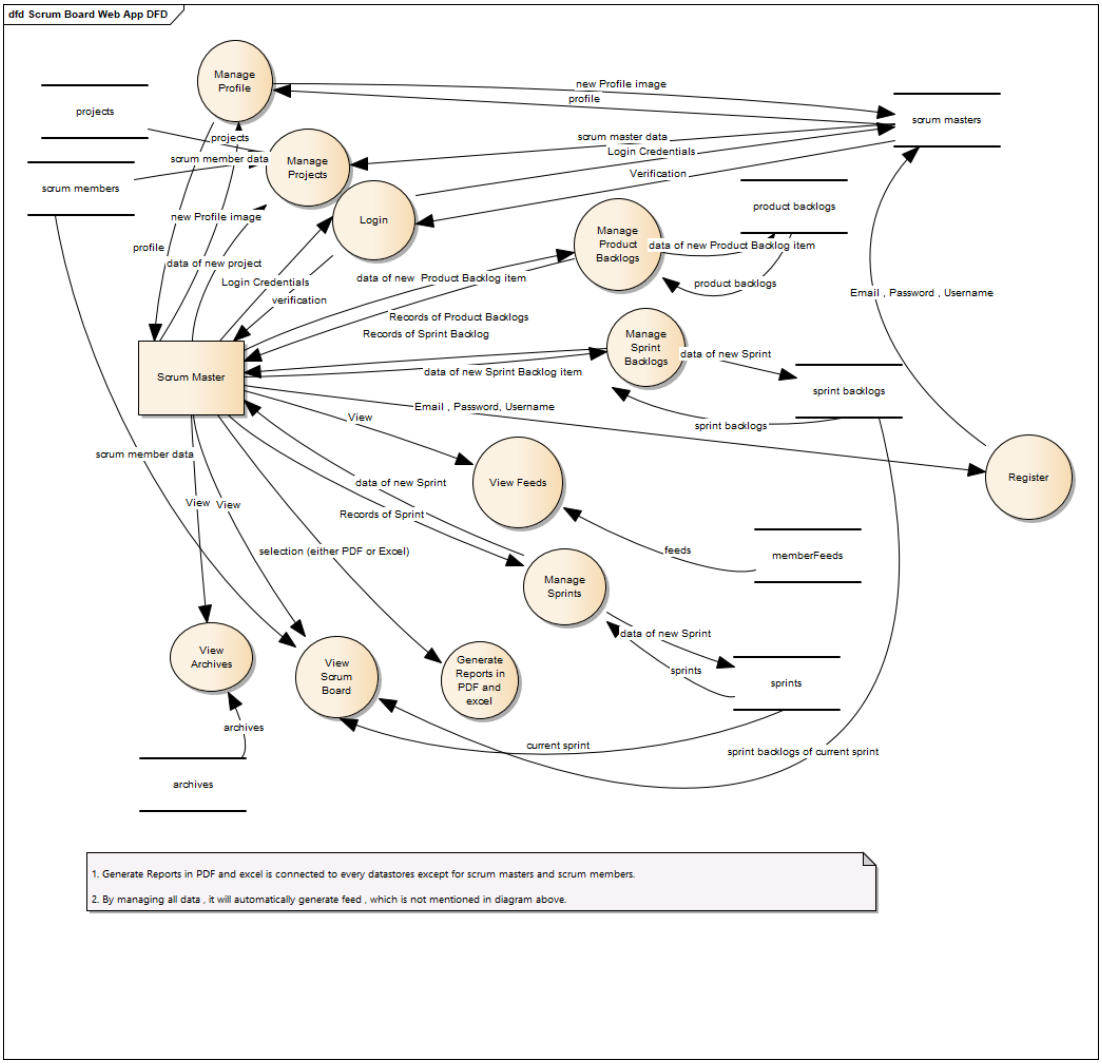


Figure 4.4: Data flow diagram of Scrum Board Web Application

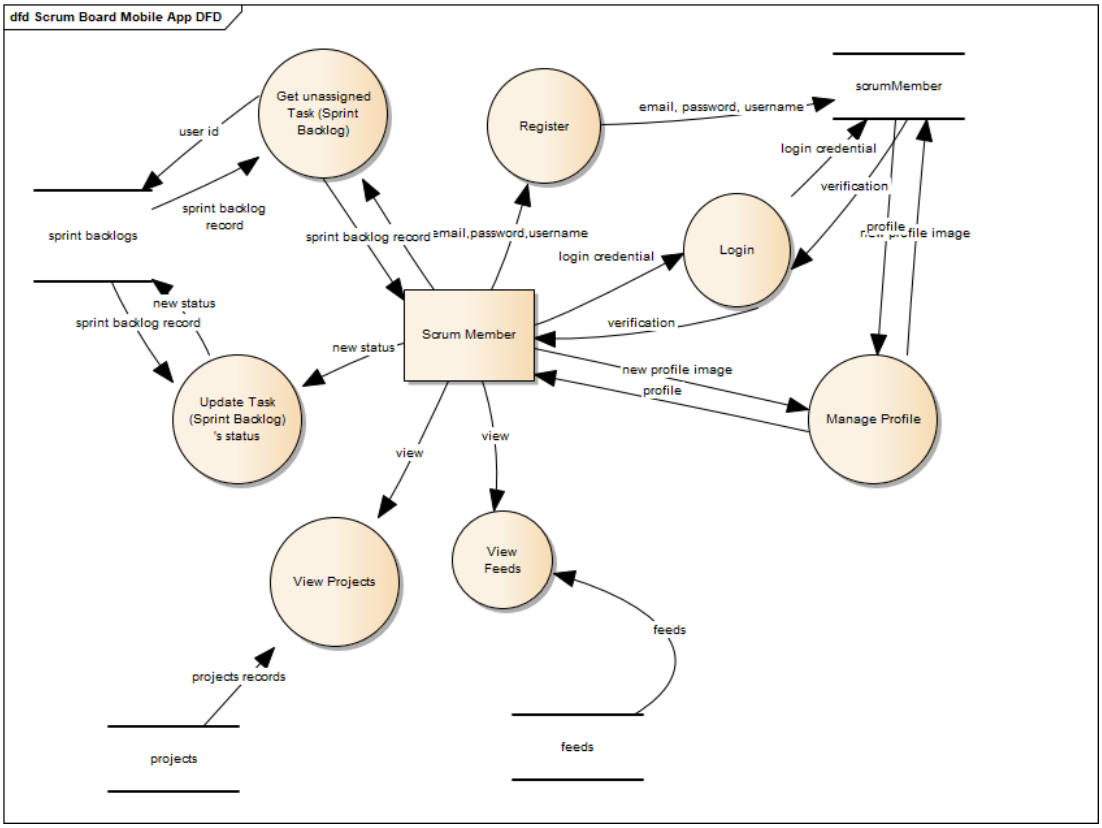


Figure 4.5: Data flow diagram of Scrum Board Mobile Application

4.6 System Architecture

The ideal system architecture that should be adopted in this project was shown in the Figure below, which is a 3-tier architecture. In the back end, Firebase authentication was used for authentication for register, login and reset password. For the database, Firebase cloud firestore was used to store the collection of the data needed such as sprint backlogs, projects and etc. Firebase cloud storage was used to store images and documents such as profile picture of users. Restful APIs server were created with Node.js and Express and implemented to increase reusability and maintainability of backend code. It was hosted in Heroku, a freemium cloud application platform.

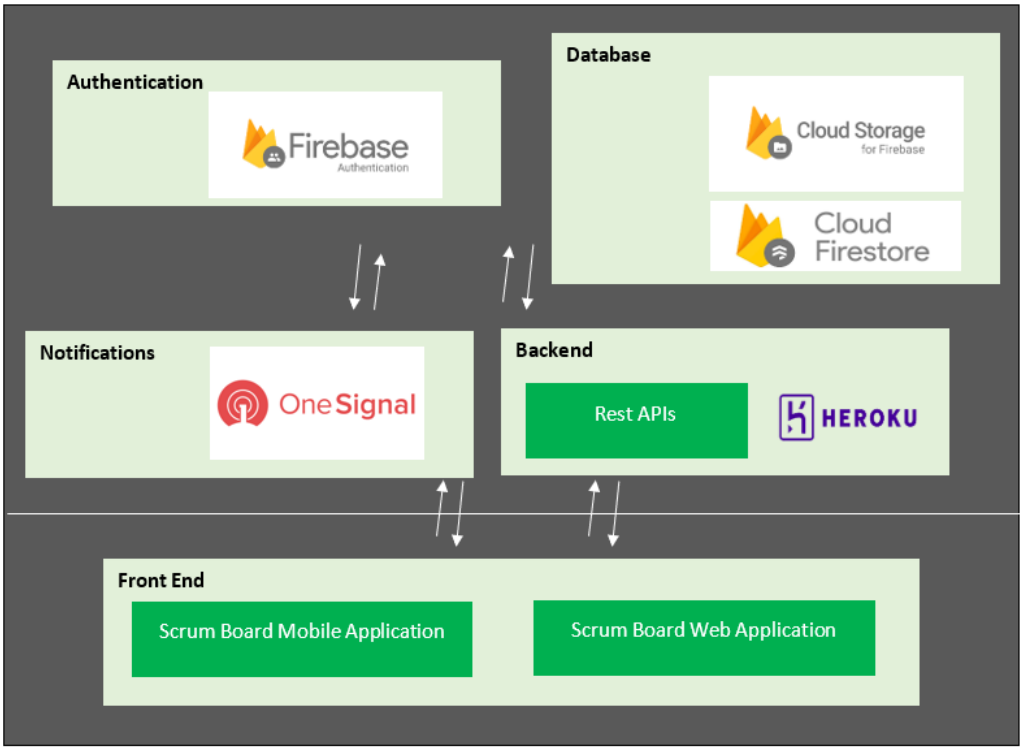


Figure 4.6: The ideal reactive core architecture

However, the realistic system architecture that was implemented in the both applications were quite limited due to short timeline, insufficient knowledge and skills in creating customised APIs using expressJs and nodeJs. The backend was done together with the front end, which meant that the no restful APIs are purposely created during the implementation. The applications communicate directly to the database.

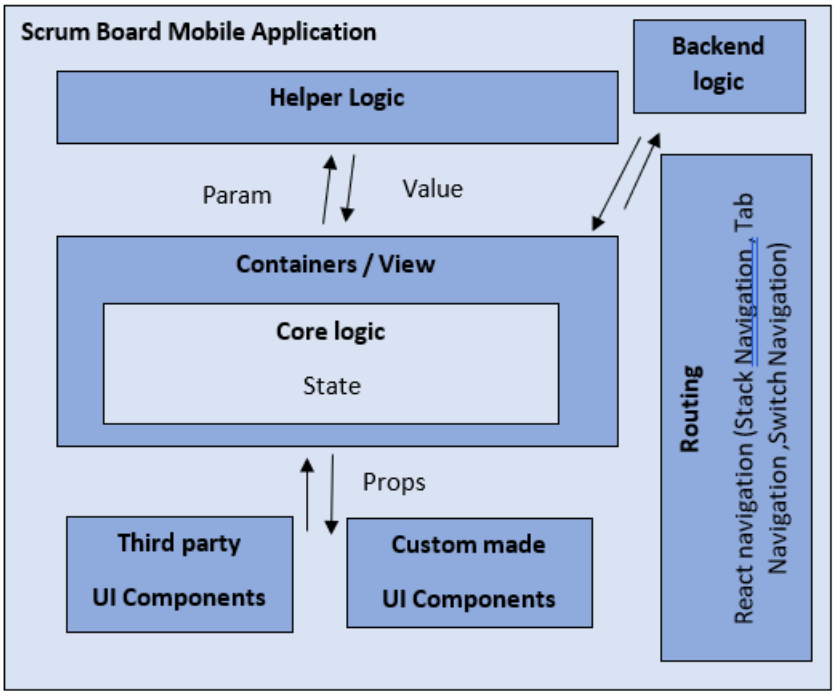


Figure 4.7: Scrum Board Mobile application system architecture

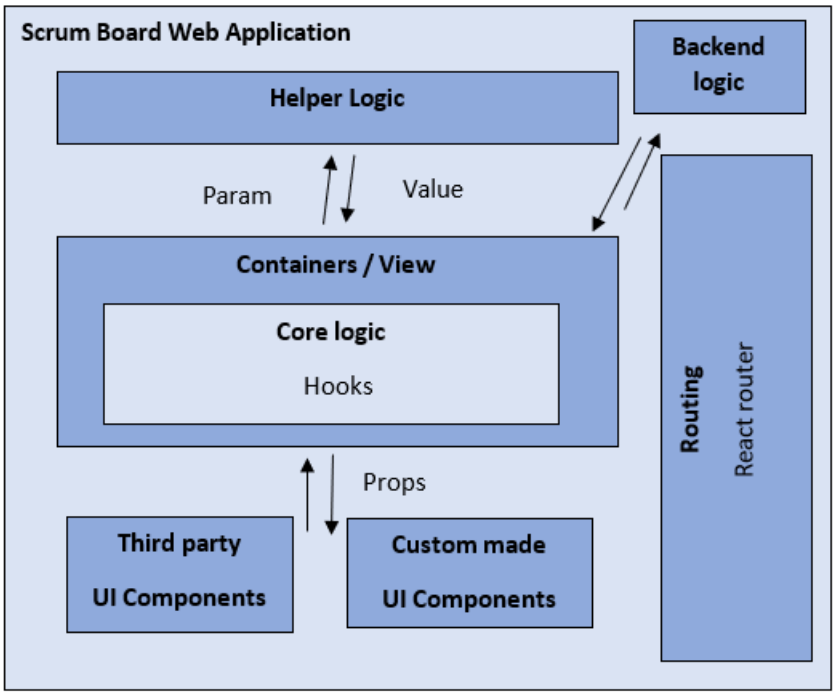


Figure 4.8: Scrum Board Web application system architecture

There were some similarities between the system architecture in Figure 4.4 and Figure 4.5. First of all, both of these system architectures were focus on Front End development. Secondly, UI components (third party UI libraries or custom made) were used to build up a user-friendly user interface. Each application had its core logic sustained inside the view. To illustrate this point, image there is a specific page which displays all projects scrum master / scrum member participated in it. First, it had to fetch all the project data and present them in a way that the user wanted to see. The redundant helper logics were stored individually in separate file so it could be reused in other view. An example of helper logics was to get the priority value of the sprint activity and return the background colour of the badge to display the priority value. Different badge colour shown distinction and allowed user to easily recognise the sprint task ‘s priority without clicking in for more details. On the other hand, React and React Native framework had their own routing and navigation setup. The major difference in between these two-applications system architectures was the technique they used to maintain their core logic. In the Scrum Board Mobile application , class component and lifecycle methods were used to change states to make it reusable and encapsulate. In the Scrum Board Web application , hooks were used so that it could use state without the needs of using class. Hooks was more suggestable because it made the code easier to be read and understand. Morever, using hooks shorten the codes and increased the performance.

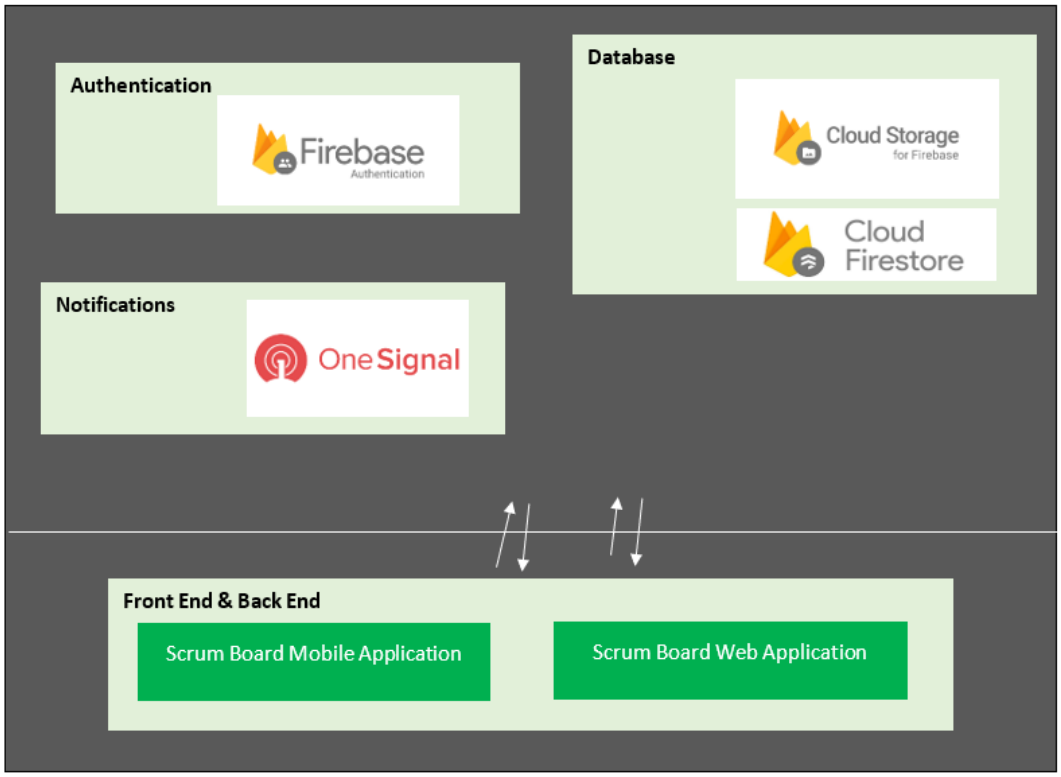


Figure 4.9: an overview of the overall system architecture of both applications.

4.7 Database Design

Firebase cloud firestore is a NoSQL, document-oriented database. Unlike SQL, it doesn't have table or rows. Data is stored in documents, which are organized into collections. Each document contains a set of key value pairs and it also might contain sub collections of small documents.

4.7.1 Logical Entity Relationship Diagram

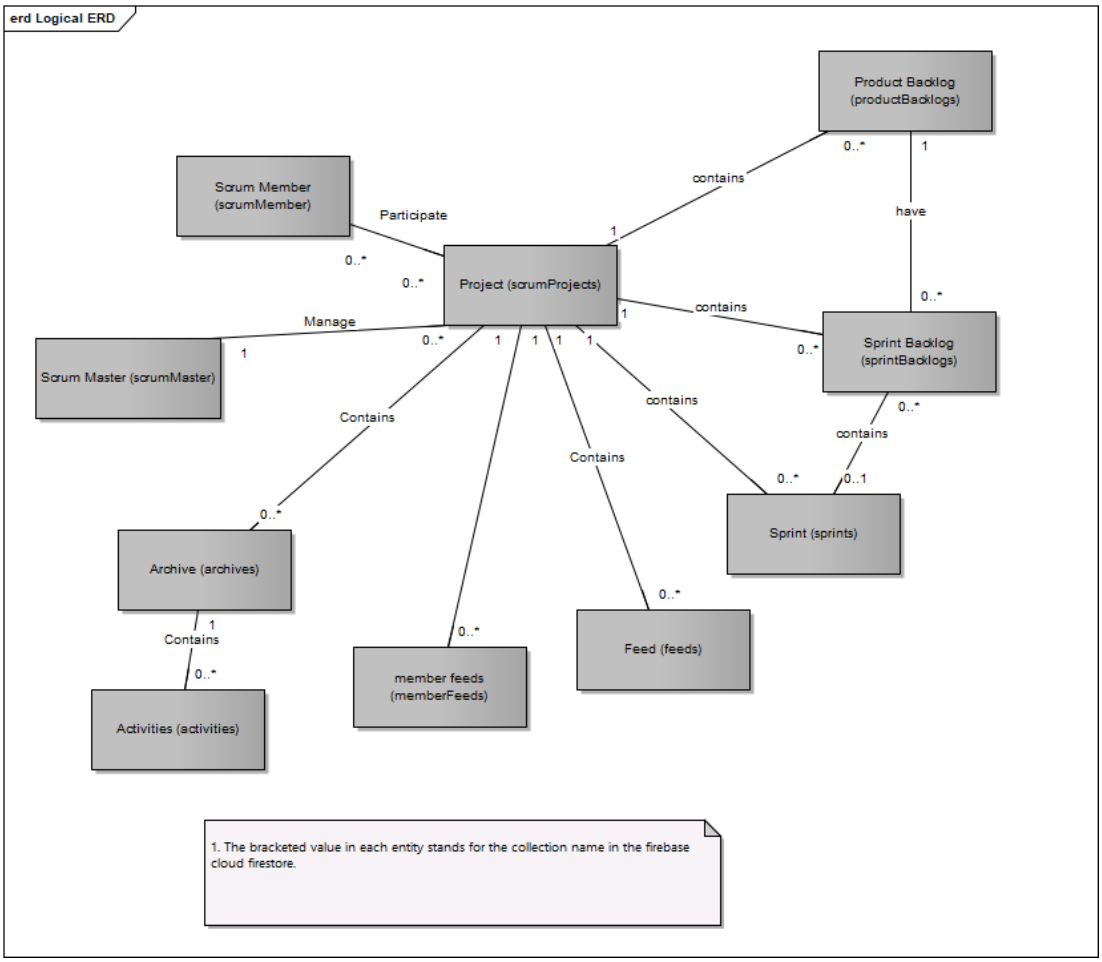


Figure 4.10: Logical entity relationship diagram

There were currently 10 collection of documents store inside the Firebase cloud firestore. They were:

Table 4.20: Database Collections

Collection	**Description**
sprints	contains every sprint document
sprintBacklogs	contain every sprint backlog document
productBacklogs	contain every product backlog document
scrumProjects	contain every project document
scrumMaster	contain every scrum master document
scrumMember	contain every scrum member document
Archives	contain every archive (closed sprint) document

Activities (sub collection of archives)	contain every sprint task documents of the particular archive
feeds	contain every feed document created by Scrum Master
memberFeeds	Contain every feed document created by Scrum Member

4.7.2 Physical Entity Relationship Diagram

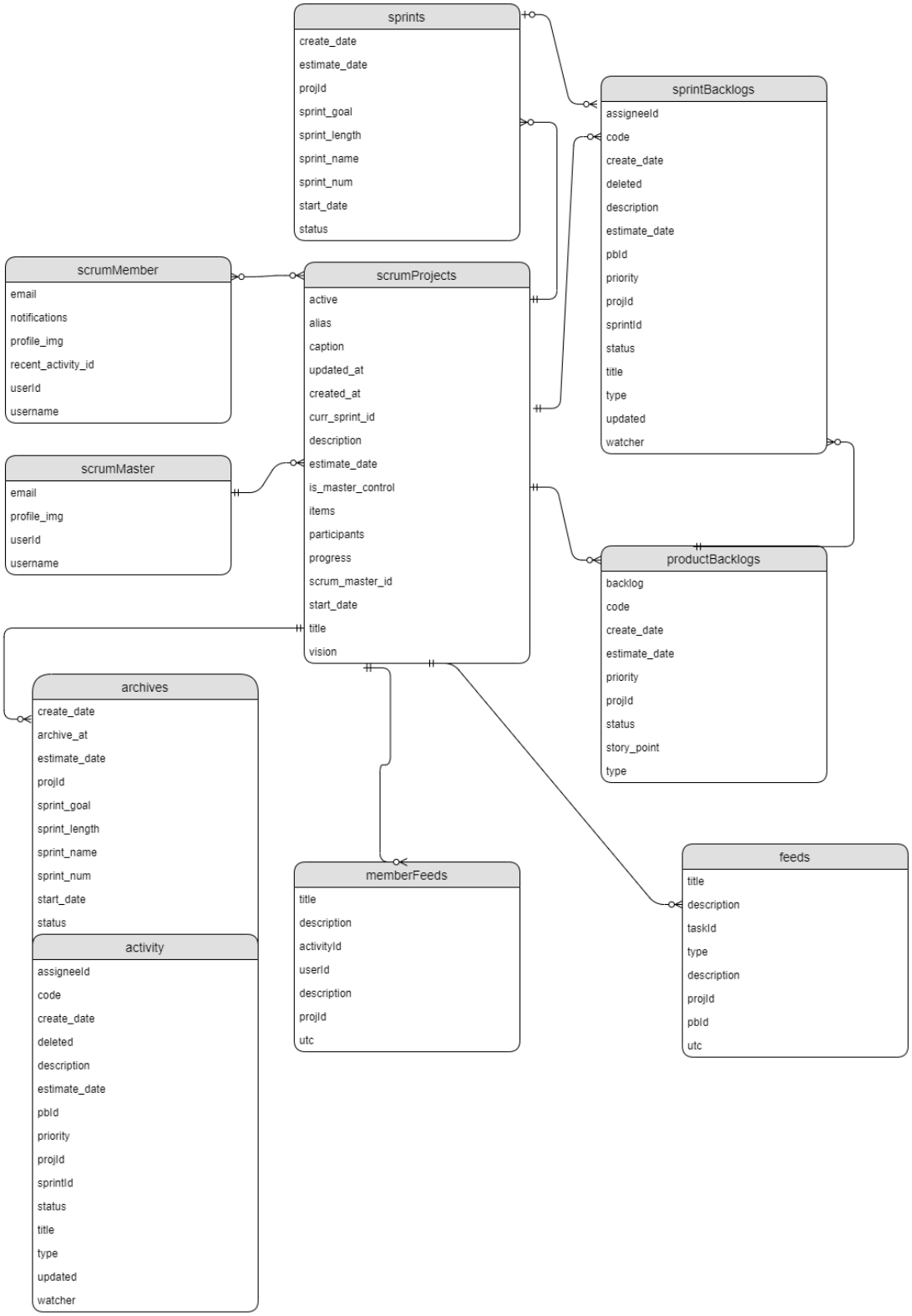


Figure 4.11: Physical entity relationship diagram

Table 4.21: scrumMember document

Field	**Type**	**Description**
email	string	The email of the scrum member.
notifications	boolean	Check whether the scrum member wants to turn on the notification for updated changes from scrum board web application.
Profile_img	string	The profile image of the scrum member.
Recent_activity_id	string	The sprint task id that the scrum member recently updated. (Reference to sprintBacklogs collection).
userId	string	The user id gets from firebase authentication.
Username	string	Contains the username of the scrum member.

Table 4.22: scrumMaster document

Field	**Type**	**Description**
Email	string	The email of the scrum master.
Profile_img	string	The profile image of the scrum master.
userId	string	The user id gets from firebase authentication.
Username	string	The username of the scrum master.

Table 4.23: scrumProjects document

Field	Type	Description
active	boolean	Check if the scrum project is active or closed
alias	string	An abbreviation of the scrum project title.
caption	string	A short description of the scrum project.
Updated_at	array	The last updated date of the scrum project
Created_at	string	The creation date of the scrum project
Curr_sprint_id	string	The id of the current ongoing sprint (Reference to sprint collection)
description	string	A long description of the scrum project.
Estimate_date	string	The due date of the scrum project.
Is_master_control	boolean	Check if the whole scrum board can only be edited and updated by Scrum Master.
items	number	The total number of product backlogs item of the project.
participants	array	An array of scrum member id to indicate which scrum member participates in this scrum project (Reference to scrumMember collection).
Start_date	string	The start date of the scrum project.
title	string	The title of the scrum project.
Vision	String	The project vision of the scrum project.

Table 4.24: productBacklogs document

Field	**Type**	**Description**
Backlog	string	Check if the scrum project is active or closed
Code	string	A unique code for the product backlog item.
Create_date	string	The creation date of the product backlog item.
Estimate_date	string	The due date of the product backlog item.
Priority	string	The priority of the product backlog item.
projId	string	The id of the scrum project which the product backlog item belongs to (Reference to scrumProjects collection)
Status	string	The status of the product backlog item (complete, open, incomplete, close).
story_point	string	The efforts and resources needed to complete the product backlog item.
Type	String	The type of the product backlog item.

Table 4.25: sprintBacklogs document

Field	Type	Description
assigneeId	string	The id of the assignee to the sprint backlog item (Reference to scrumMember collection)
code	string	A unique code for the sprint backlog item.
Create_date	string	The creation date of the sprint backlog item.
deleted	boolean	Check if the sprint backlog item is soft deleted
description	string	A description of the sprint backlog item.
Estimate_date	string	The due date of the sprint backlog item.
pbId	string	The parent id product backlog item of the sprint backlog item (Reference to productBacklogs collection).
priority	string	The priority of the product backlog item.
projId	string	The id of the scrum project which the sprint backlog item belongs to (Reference to scrumProjects collection).
sprintId	string	The id of the selected sprint the sprint backlog item allocated to (Reference to sprint collection).
status	string	The status of the sprint backlog item (toDo, ongoing, complete).
title	string	The title of the sprint backlog item.
type	String	The type of the sprint backlog item.

Updated	boolean	Check if the sprint backlog item has been updated by scrum member from scrum board mobile application.
Watcher	array	An array of the scrum member id to indicate which scrum member is watching the sprint backlog item (Reference to scrumMember collection)

Activities (sub collection inside archives document) document had the similar layout with sprintBacklogs document.

Table 4.26: sprints document

Field	**Type**	**Description**
Create_date	string	The creation date of the sprint.
Estimate_date	string	The due date of the sprint.
projId	string	The id of the scrum project which the sprint belongs to (Reference to scrumProjects collection)
Sprint_goal	string	The sprint goal of the sprint
Sprint_length	string	The total duration of the sprint in days
Sprint_name	string	The name of the sprint, created by the system
start_date	String	The start date of the sprint.
status	string	The status of the sprint (open, ongoing, close).

Archives document had the similar layout with sprints document but with an additional sub collection named activities.

Table 4.27: feeds document

Field	Type	Description
description	string	The description of the feed.
projId	string	The id of the scrum project which the feed belongs to (Reference to scrumProjects collection)
taskId	string	Id of newly create sprint backlog item or updated sprint backlog item (Reference to sprintBacklogs collection).
title	string	The title of the feed.
type	String	The type of the feed.
utc	string	The date of the feed is created.

Table 4.28: memberFeeds document

Field	Type	Description
description	string	The description of the feed.
projId	string	The id of the scrum project which the feed belongs to (Reference to scrumProjects collection)
activityId	string	Id of updated sprint backlog item (Reference to sprintBacklogs collection).
title	string	The title of the feed.
userId	String	The ScrumMember who create this feed (Reference to scrumMembers collection).
utc	string	The date of the feed is created.

4.8 Initial System Design (Prototype)

This section provided a set of illustrations of system design of the prototype of scrum web and mobile application.

4.8.1 Scrum mobile application (Prototype)

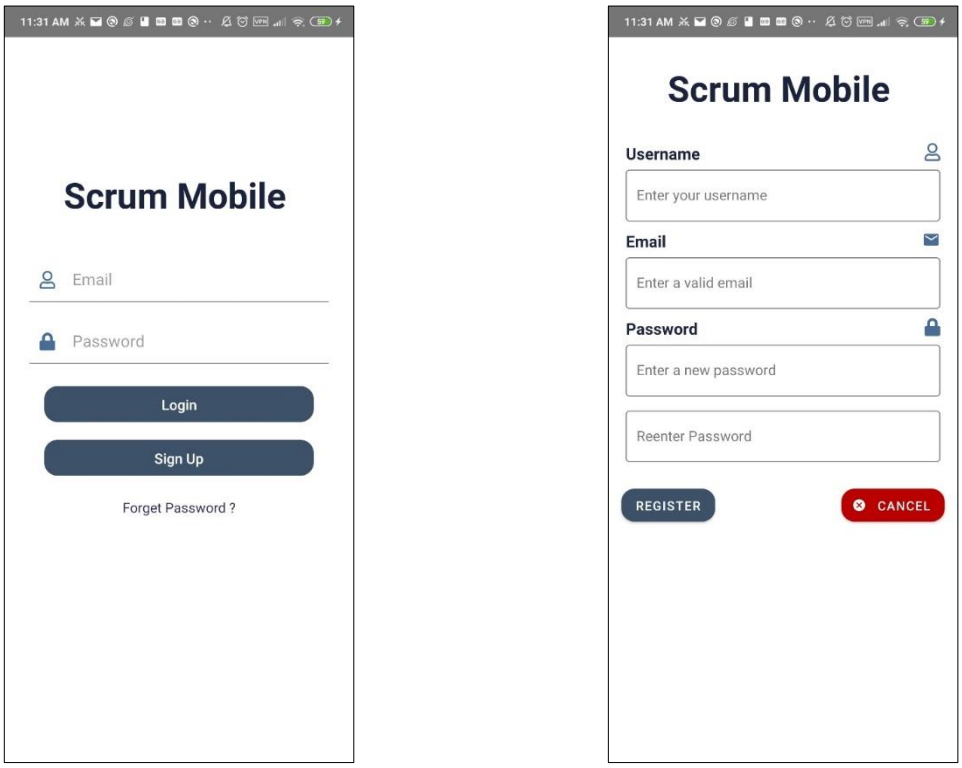


Figure 4.12: Login screen (left) and Register screen (right)

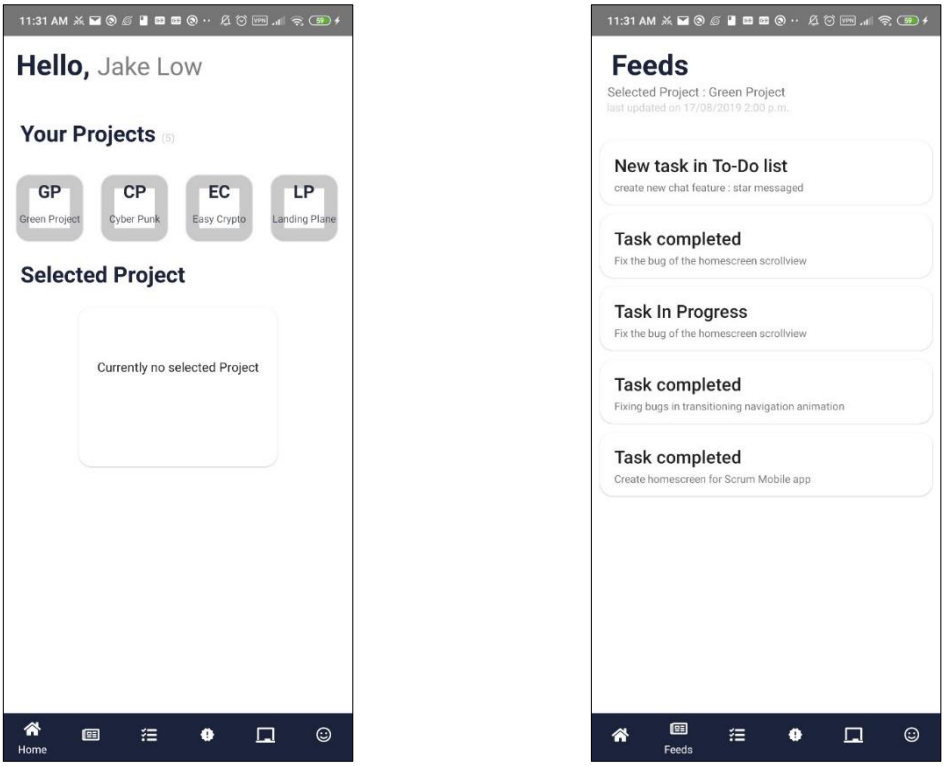


Figure 4.13: Project Dashboard screen (left) and Feed screen (right)

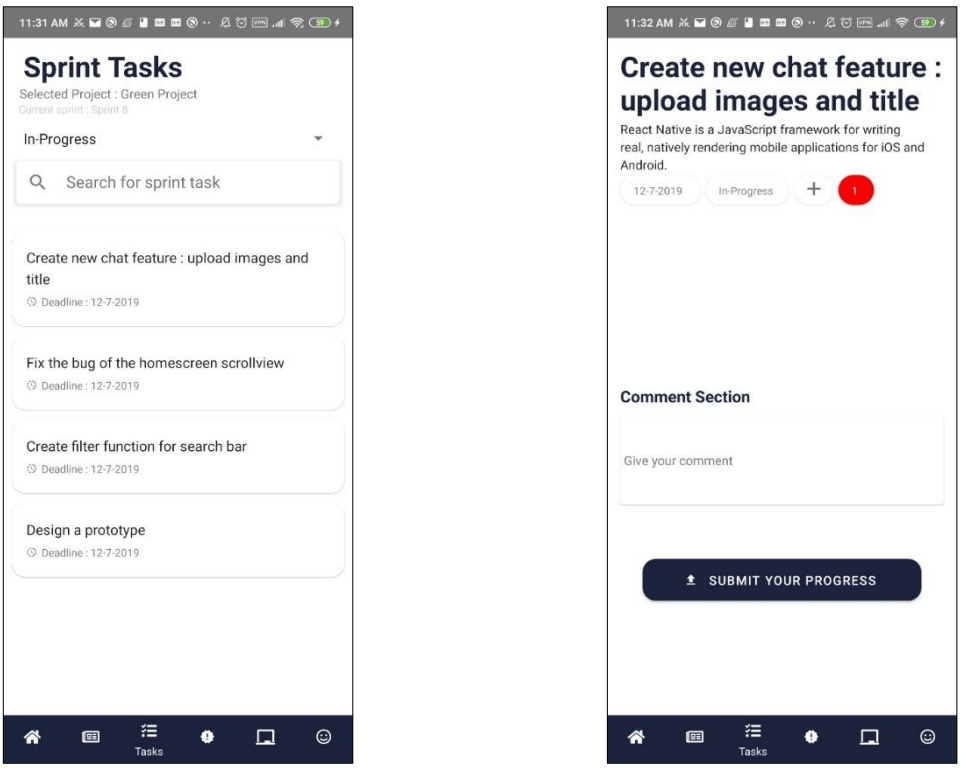


Figure 4.14: List of user sprint tasks screen (left) and Sprint Task ‘s detail screen (right)

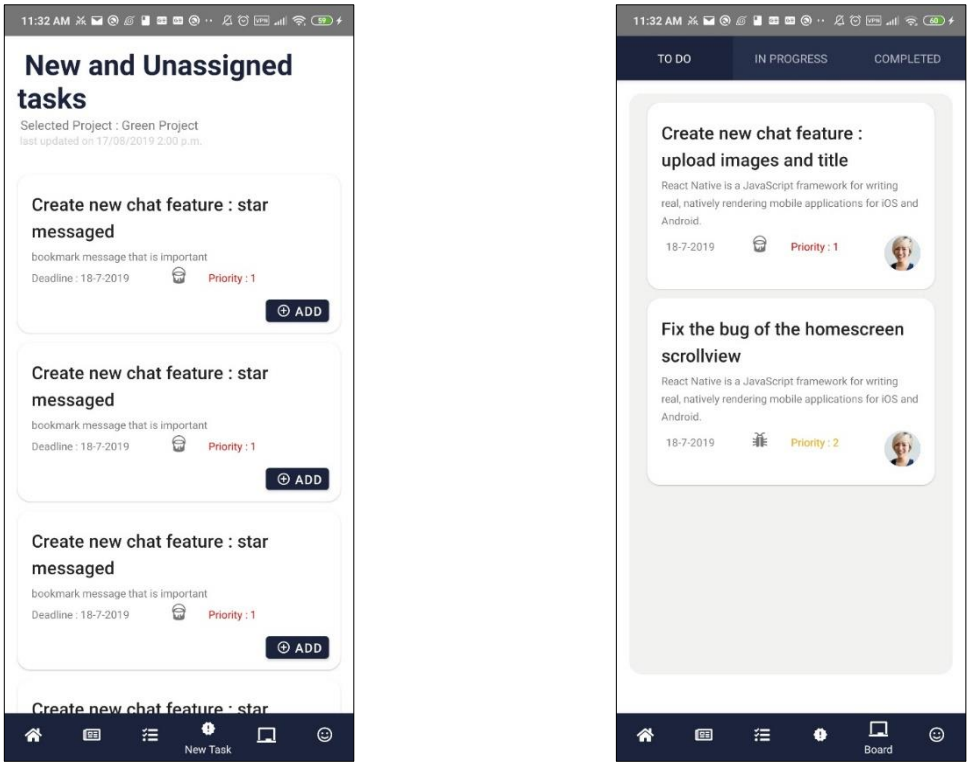


Figure 4.15: List of unassigned sprint tasks screen (left) and Scrum board screen (right)

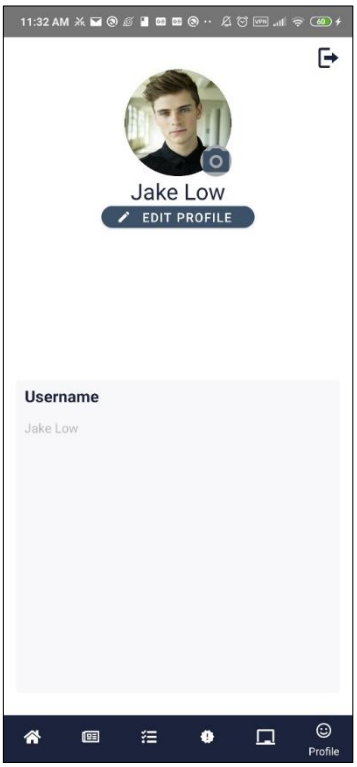


Figure 4.16: Profile screen

4.8.2 Scrum Web Application (Prototype)

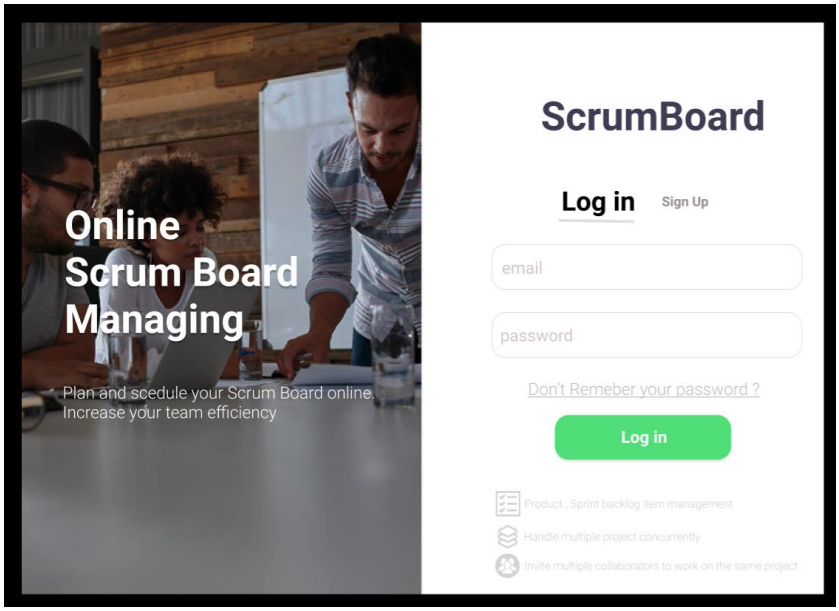


Figure 4.17: Login Screen

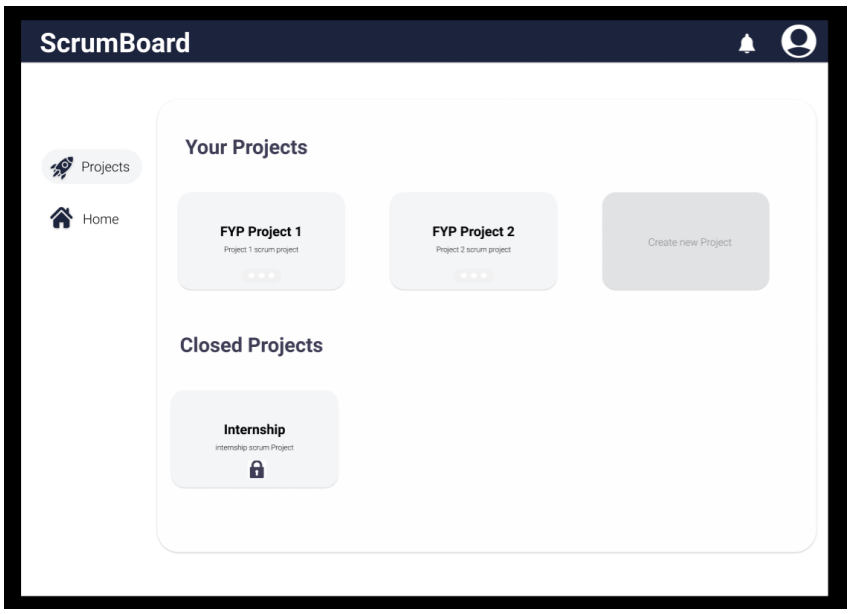


Figure 4.18: Project Dashboard screen

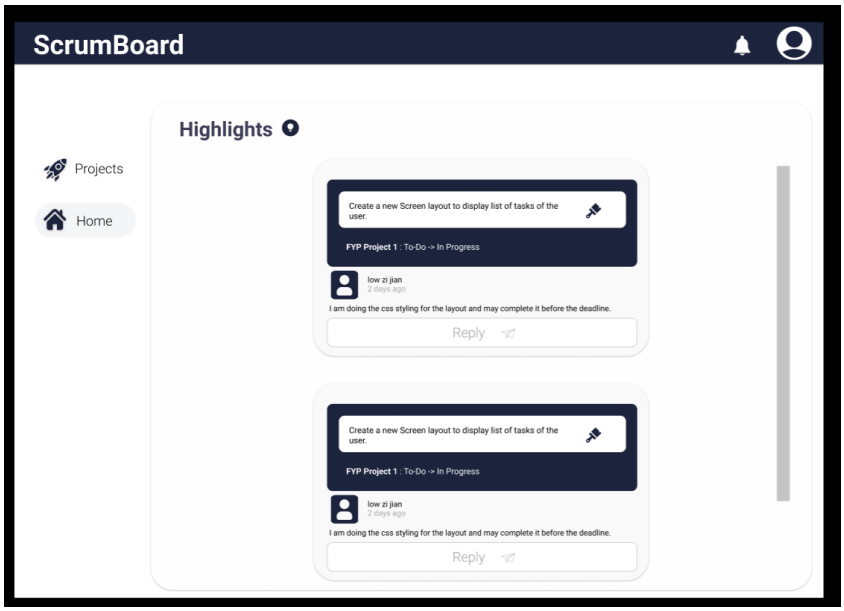


Figure 4.19: Project Feed screen

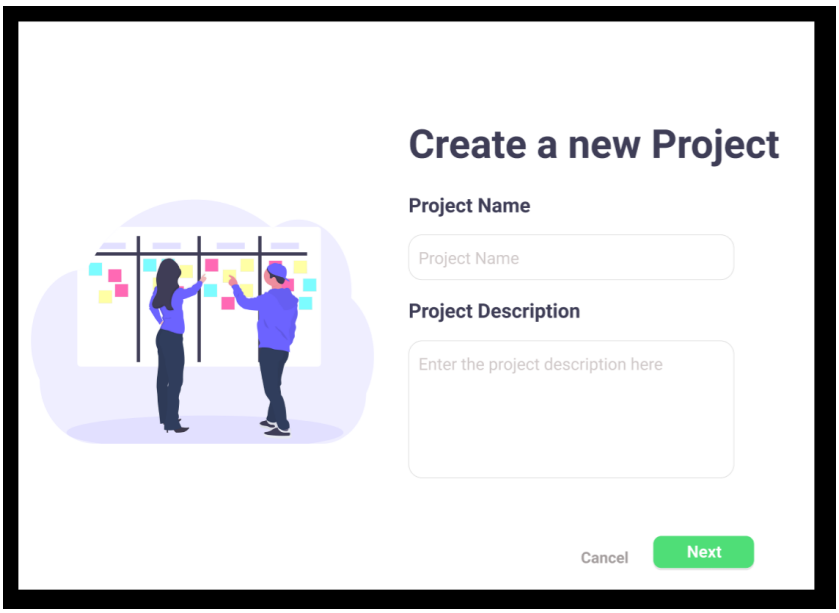


Figure 4.20: Create new project screen

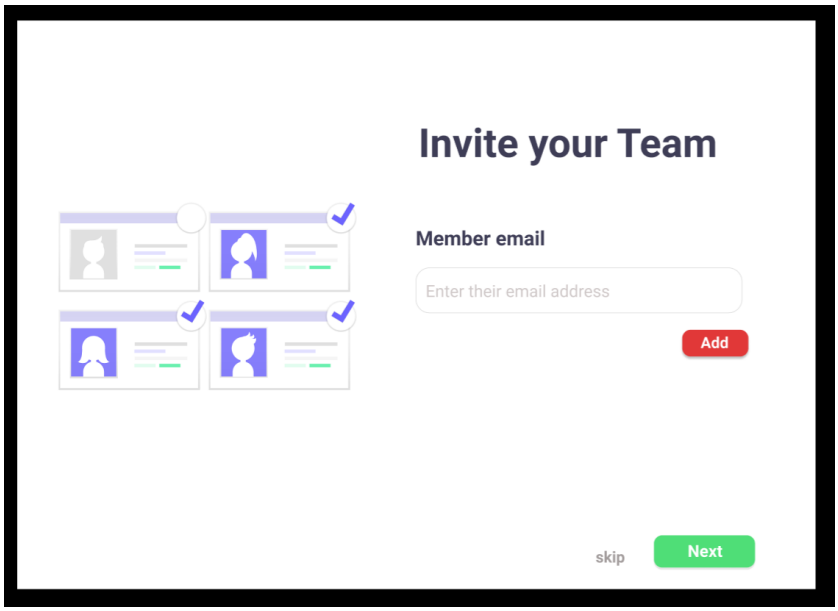


Figure 4.21: Invite you team screen

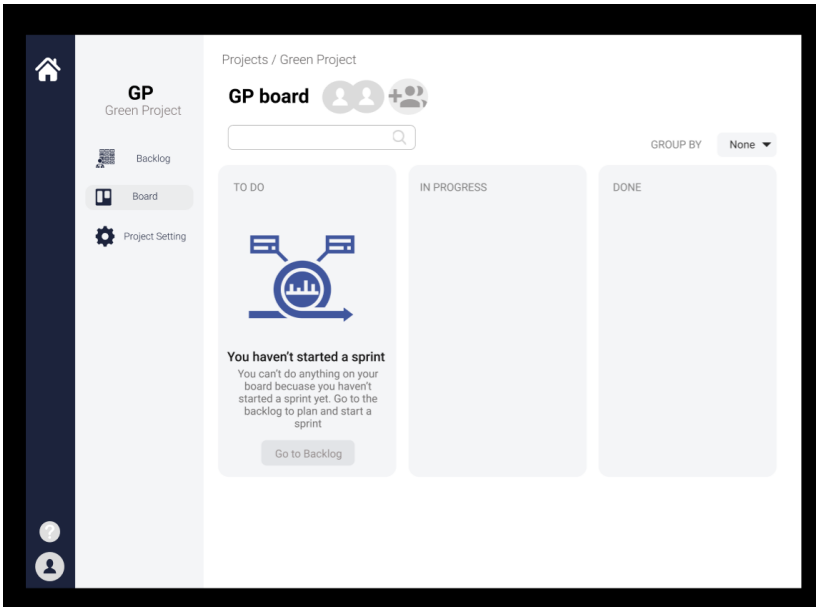


Figure 4.22: Empty scrum board screen

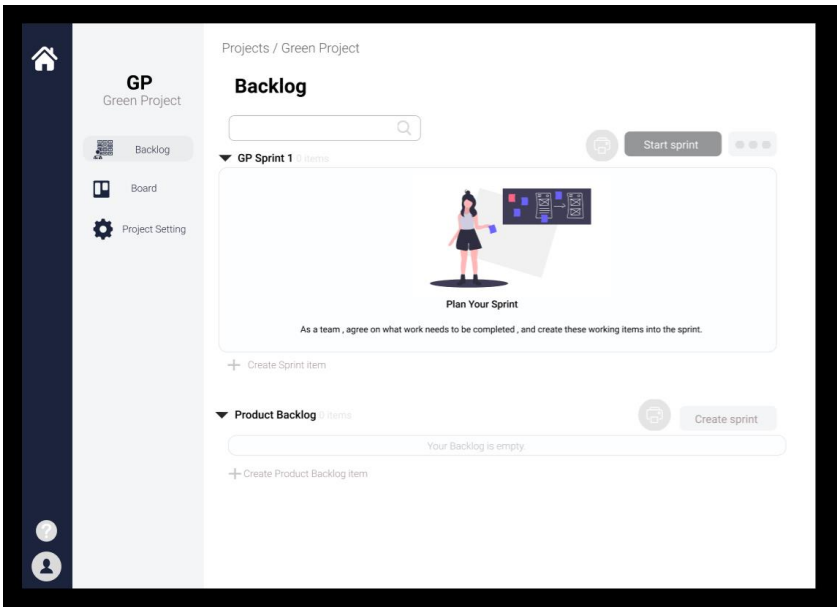


Figure 4.23: Sprint Backlog and Product Backlog screen

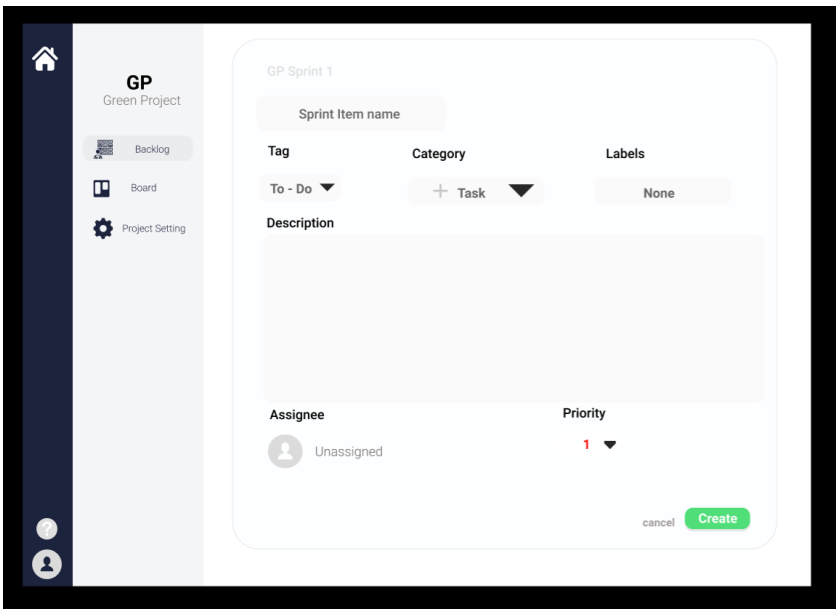


Figure 4.24: Create Sprint Backlog screen

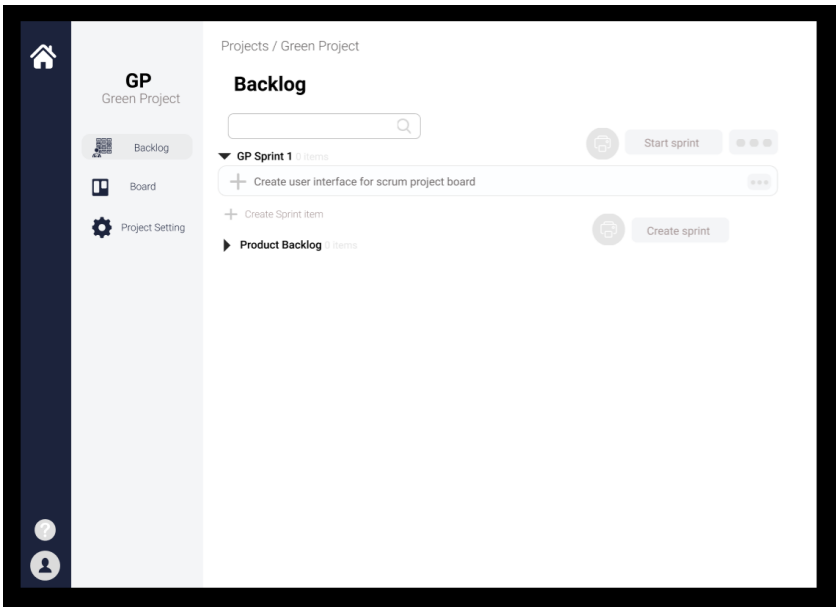


Figure 4.25: Sprint backlog is created screen

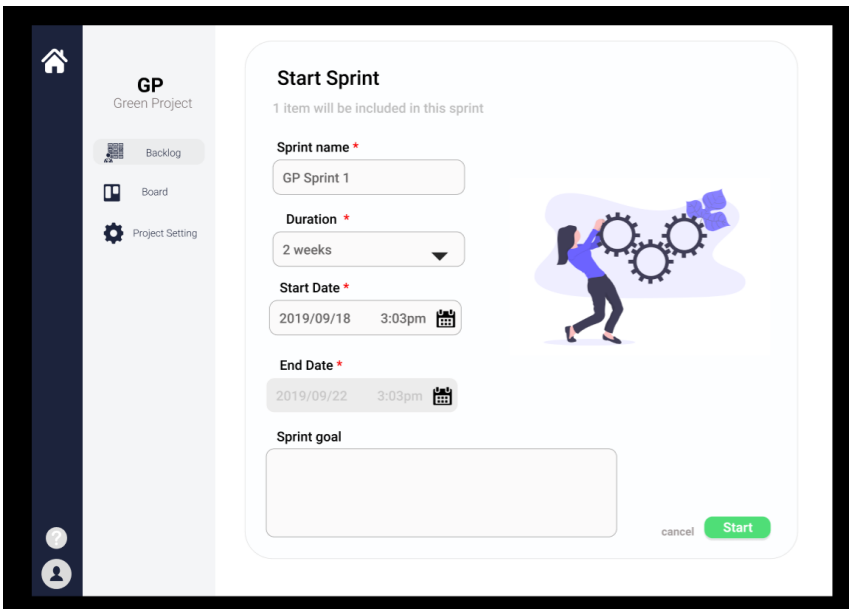


Figure 4.26: Create new sprint sample screen

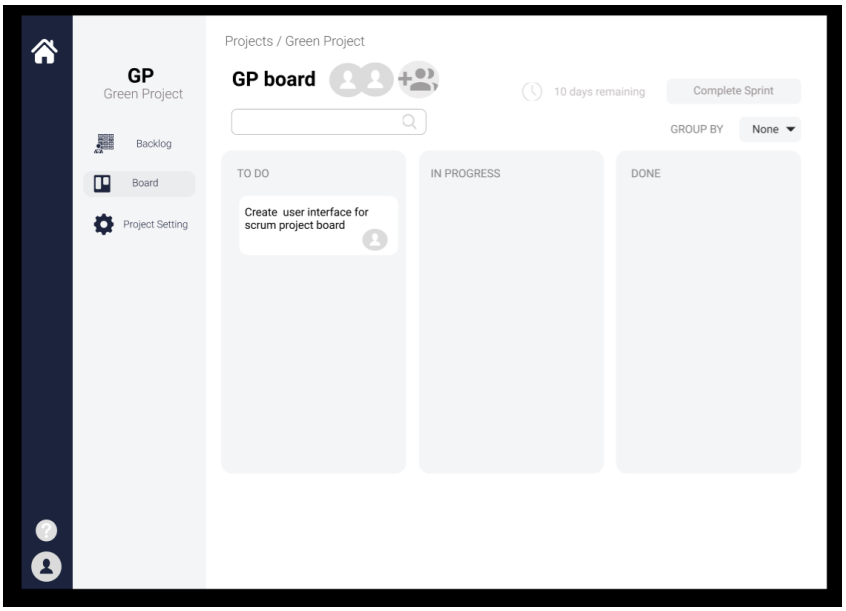


Figure 4.27: Scrum board with sprint tasks screen

CHAPTER 5

SYSTEM IMPLEMENTATION

5.1 Introduction

The overall system possesses two types of users: scrum master and scrum member. The scrum master uses the scrum board web application to manage the scrum process of the project including sprint and backlogs management. The scrum members use the scrum board mobile application to keep track of their sprint tasks that being assigned from the scrum master through the web application. Both users have their respective modules for accessing different functionalities. The modules for each user are displayed as below:

Table 5.1: Modules for respective users

Type of Users	Application	Modules
Scrum Master	Scrum Board web application	Login
		Register
		Forgot Password / Reset Password
		Project Management
		Product Backlog Management
		Sprint Backlog Management
		Sprint Management
		View Archive
		View Scrum board
		Generate Reports
		Profile Management
		View Notification and feeds
Scrum Member	Scrum Board mobile application	Login
		Register

		Forgot Password / Reset password
		View Projects
		Update assigned sprint tasks
		View Scrum Board of current sprint
		Get unassigned sprint tasks
		View Notifications and Feeds
		Profile Management

Section 5.2 explains the modules of scrum board web application and section 5.3 explains the modules of scrum board mobile application. Section 5.4 describes user authentication and how route protection is implemented in scrum board web application. Section 5.5 explains how form validation is done. Section 5.6 shows an overview of libraries that used in the projects. Section 5.7 explains how the notification system works in the Scrum Board applications using the OneSignal API. Section 5.8 shows an overview of database structure of Cloud Firestore that is being used in the Scrum Board applications.

5.2 Modules for scrum board web application

5.2.1 Login Module

The module serves as the initial page of the scrum board web application. Scrum Master enters the valid credentials and the firebase authentication will help to check if the credentials entered matched with the user database. If the credential entered are compliance with the data in the database, it will automatically redirect Scrum Master to his / her project dashboard or else, an error message will be shown. In Figure 5.2, “The password is invalid, or the user does not have a password” is shown when the Scrum Master enter invalid password. Figure 5.3 shows the code segment of the login verification by firebase authentication.

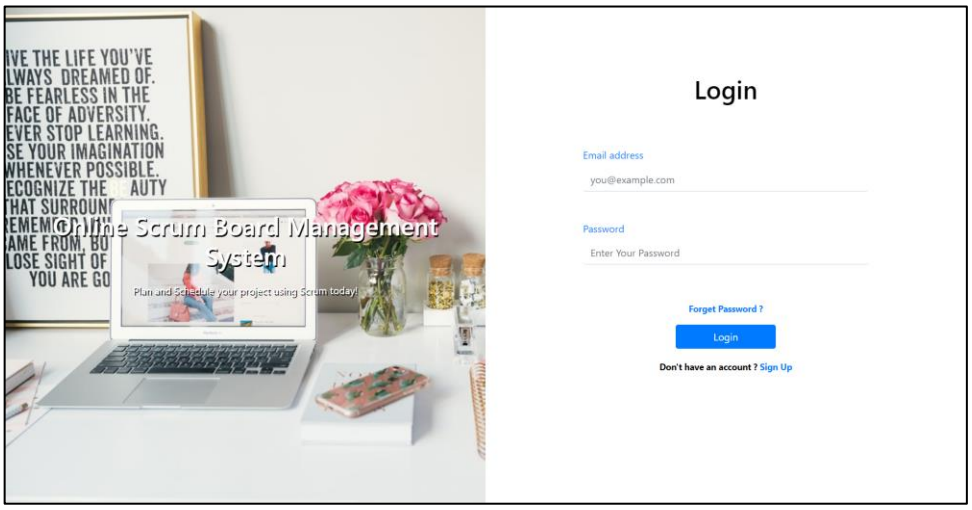


Figure 5.1: Login screen

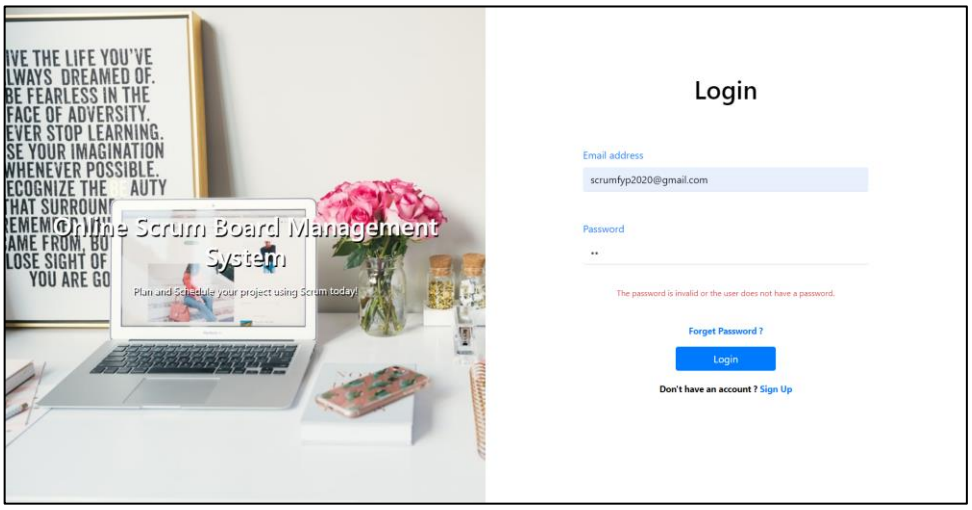
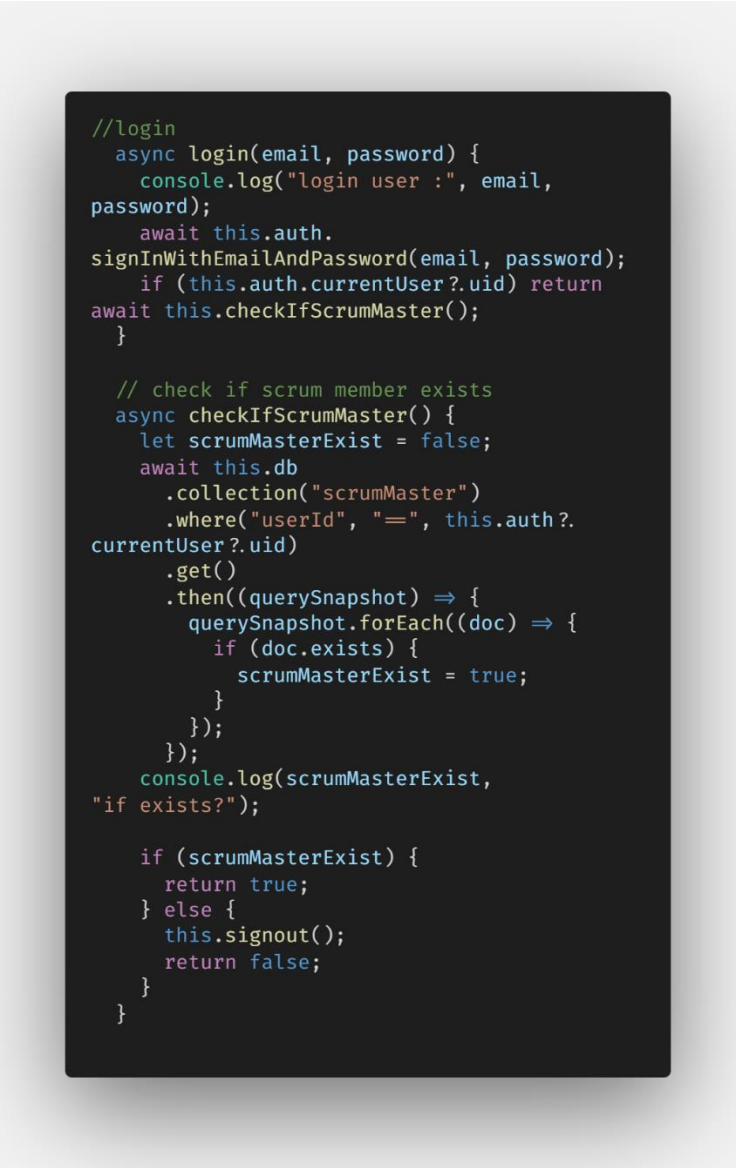


Figure 5.2: Login screen when user entered the wrong password



```
//login
  async login(email, password) {
    console.log("login user :", email,
password);
    await this.auth.
signInWithEmailAndPassword(email, password);
    if (this.auth.currentUser?.uid) return
await this.checkIfScrumMaster();
  }

  // check if scrum member exists
  async checkIfScrumMaster() {
    let scrumMasterExist = false;
    await this.db
      .collection("scrumMaster")
      .where("userId", "==", this.auth?.
currentUser?.uid)
      .get()
      .then((querySnapshot) => {
        querySnapshot.forEach((doc) => {
          if (doc.exists) {
            scrumMasterExist = true;
          }
        });
      });
    console.log(scrumMasterExist,
"if exists?");

    if (scrumMasterExist) {
      return true;
    } else {
      this.signout();
      return false;
    }
  }
```

Figure 5.3 Code segment of the login verification by firebase authentication

5.2.2 Register Module

The module is to allow Scrum Master to create a new account with a unique email and password, along with the intended username to be used and viewed by his scrum members. The Scrum Master can navigate to this page by pressing the Sign-Up link button in the Login page.

How does the Scrum Master create a new account? Scrum Master has to enters valid information to the input fields given and clicks on the Register button. Figure 5.5 shows the code segment of register a new Scrum Master. The information will be sent to the firebase authentication and new user will be registered. Once it was done, a new

Scrum Master profile will be automatically created as document inside the scrumMaster collection. Then, Scrum Master will be redirected back to the login screen.

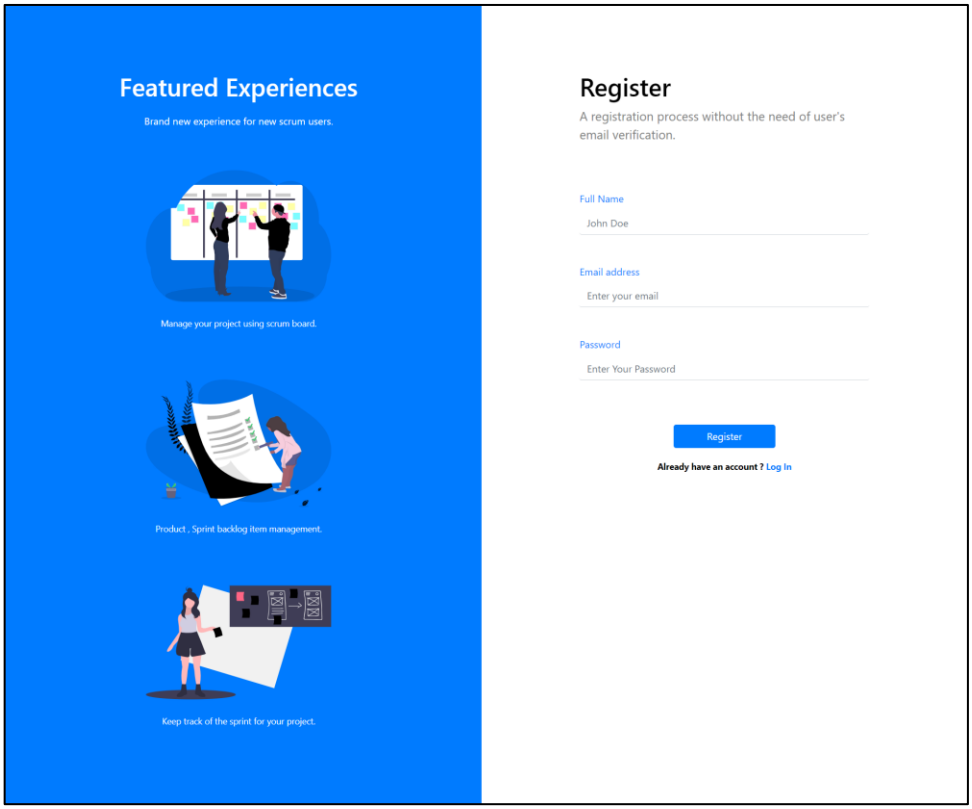


Figure 5.4: Register screen

```
//register
register(username, email, password) {
  console.log("registered user :", username,
email, password);
  this.auth.createUserWithEmailAndPassword(email
, password).then(() => {
    this.auth.currentUser
      .updateProfile({
        displayName: username
      })
      .then(() => {
        this.createNewScrumMaster();
      });
  });
}

//register → create new scrum master
createNewScrumMaster() {
  const user = this.auth.currentUser;
  console.log(user);
  this.db.collection("scrumMaster").add({
    username: user.displayName,
    userId: user.uid,
    email: user.email,
    profile_img: ""
  });
}
```

Figure 5.5: Code segment of the register by firebase authentication

5.2.3 Forgot Password / Reset Password Module

The module shown in Figure 5.6 is to allow Scrum Master to reset the password for his / her account when he / she forgot it. Scrum Master enters the recovery email of his / her account to the given input field and press the Reset Password button. It will send an email to the recovery account and provide further information on how to reset the password. Figure 5.7 shows the code segment of reset password by firebase authentication.

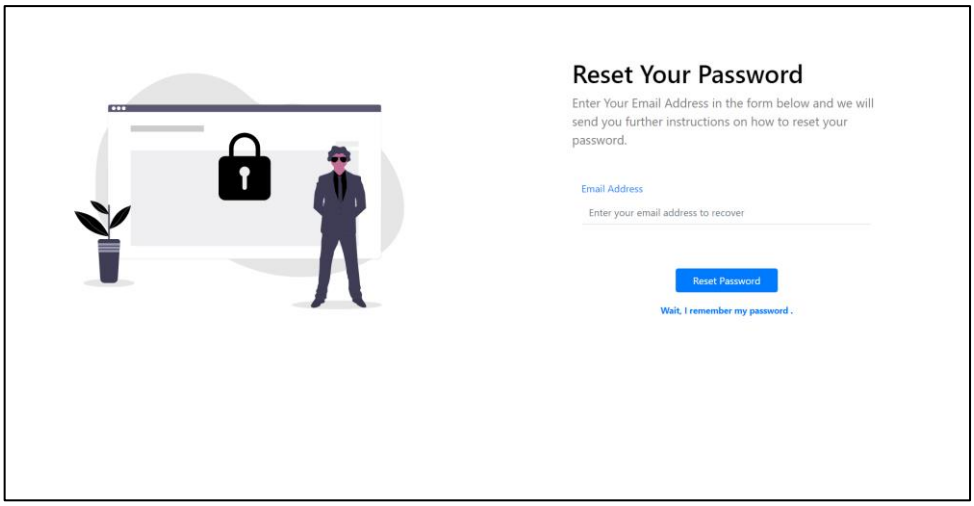


Figure 5.6: Reset Password screen

```
//reset password
 resetPassword(resetEmail) {
   console.log(resetEmail);
   return this.auth.sendPasswordResetEmail(
resetEmail);
 }
```

Figure 5.7: Code segment of the reset password by firebase authentication

5.2.4 Project Management Module

5.2.4.1 Project Dashboard

This module basically displays all projects that belongs to the Scrum Master. Figure 5.9 shows the code segment of retrieving all projects created by the Scrum Master.

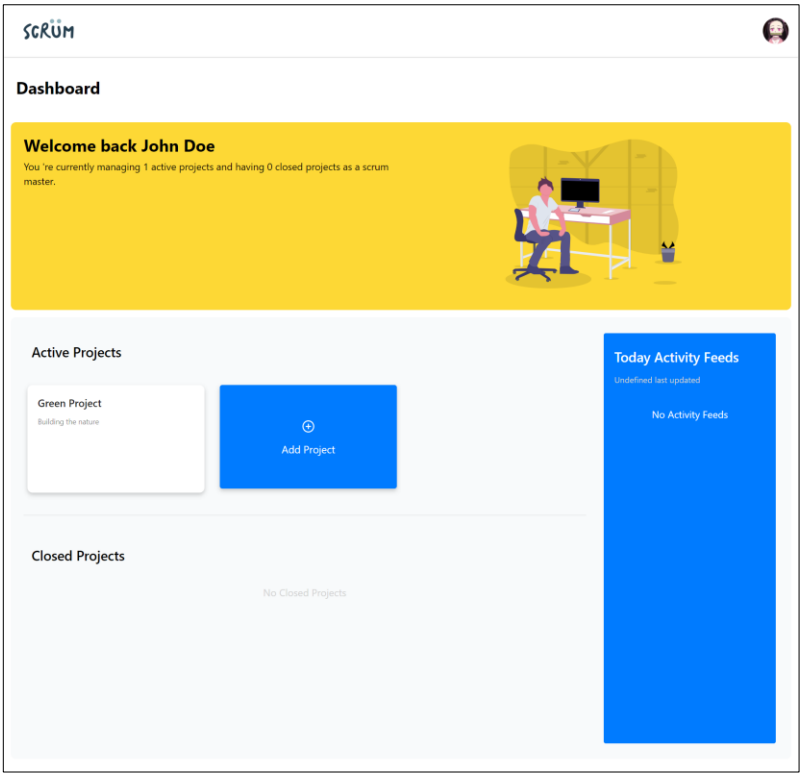


Figure 5.8: Project Dashboard screen

```
//get all projects by userId
 async getProjects(userId) {
   return await this.db
     .collection("scrumProjects")
     .where("scrum_master_id", "==", userId)
     .get()
     .then(querySnapshot => {
       let activeProject = [],
         closedProject = [];
       querySnapshot.forEach(doc => {
         let projectData = {};
         projectData = doc.data();
         projectData.projId = doc.id;
         if (projectData.active) {
           activeProject.push(projectData);
         } else {
           closedProject.push(projectData);
         }
       });
       return { active: activeProject, close:
closedProject };
     });
 }
```

Figure 5.9: Code segment of fetching all projects

5.2.4.2 Create Project

This module allows Scrum Master to create new project. How does the Scrum Master navigate to this page? He / She clicks on the Add Project button in the project dashboard. Scrum Master enters information of a new project such as title, caption, description, project vision, start date and completion date of the project. Invitation of Scrum members are optional for every creation of the project. Scrum Master need to input the scrum member ‘s email before pressing the Add button. The email is sent to firestore and return any compliance scrum member info could be found in the scrumMember collection as shown in Figure 5.12. Then, the scrum member info is added into the members array which used to keep track of all added member. Figure 5.11 shows that the scrum member has been added to the project. Figure 5.13 shows the logic of scrum member info can removed from members array. Figure 5.14 shows the code segment of creating a new project.

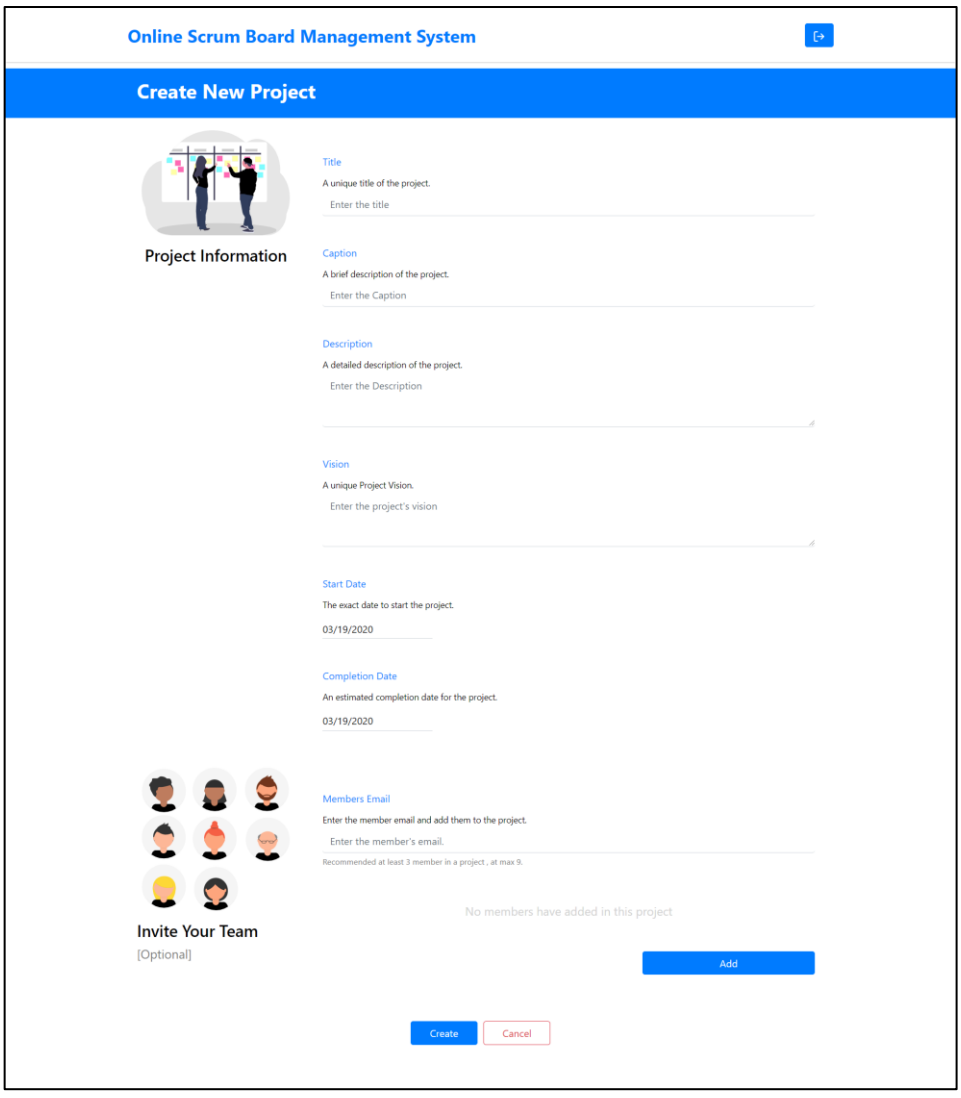


Figure 5.10: Create Project screen

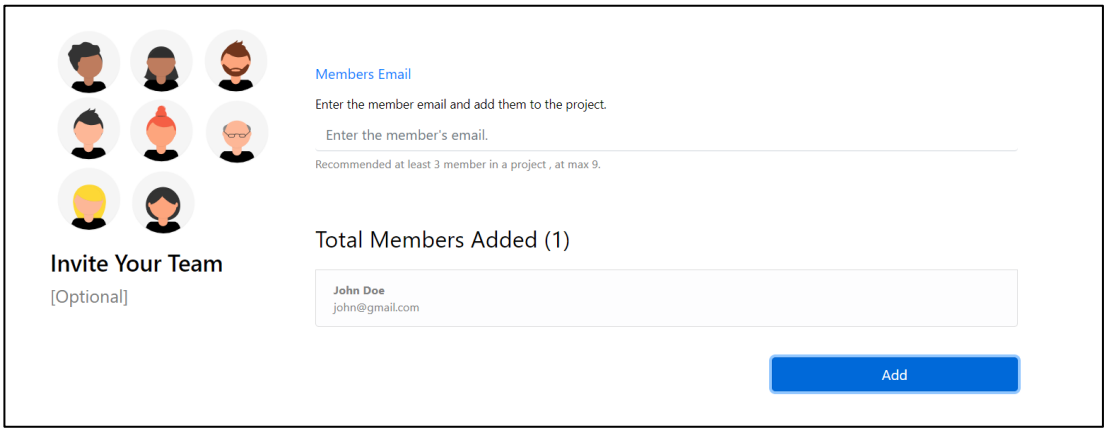


Figure 5.11: Invite New Scrum Team Member section screen

```
//get member using email  , to be used to add ne
w project member
 async addNewProjectMember(memberEmail) {
   return await this.db
     .collection("scrumMember")
     .where("email", "==", memberEmail)
     .get()
     .then(querySnapshot => {
       let memberData = null;
       querySnapshot.forEach(doc => {
         memberData = doc.data();
         memberData.docId = doc.id;
       });
       return {
         userId: memberData.userId,
         username: memberData.username,
         email: memberData.email
       };
     });
 }
```

Figure 5.12: Code segment of invite new scrum member

Figure 5.13: Code segment of removing a scrum member

Figure 5.14: Code segment of creating a new scrum Project

5.2.4.3 View Project

Scrum Master clicks on one of the project cards in the project dashboard to access to specific overview of the selected project. Figure 5.15 shows an overview of selected active project. Figure 5.16 shows an overview of selected closed project. Only active projects can be edited. To close the project, Scrum Master clicks on the Close Project on bottom. To Reopen the project, Scrum Master clicks on the Reopen Project. Figure 5.17 shows code segment of how the project data was fetched. Besides, Scrum Master can edit the project by clicking on button with the pencil icon.

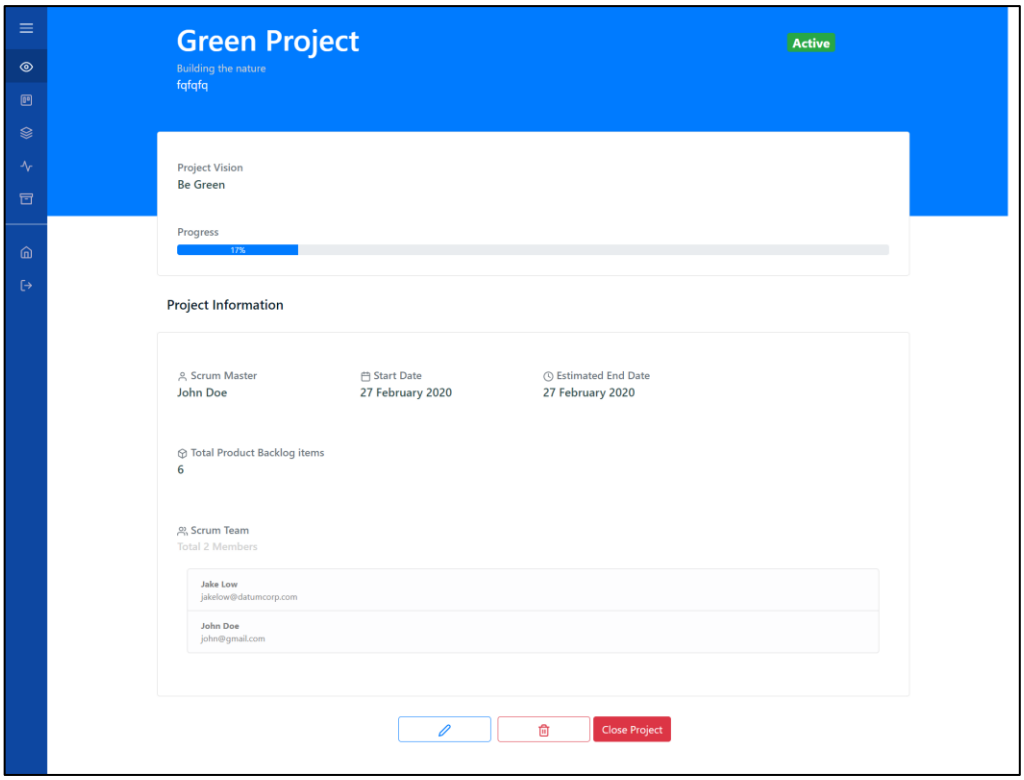


Figure 5.15: View Project screen of an active project

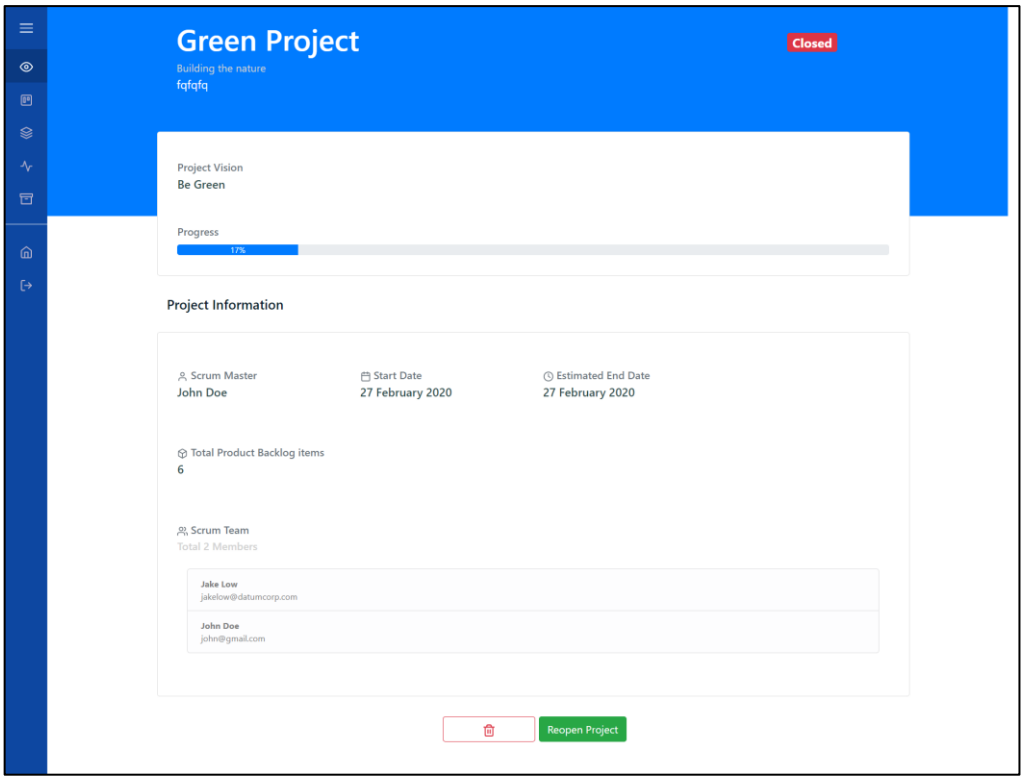


Figure 5.16: View Project screen of a closed project

```
//get specific project using proj id
 async getProjectByProjId(projId) {
   let project = await this.db
     .collection("scrumProjects")
     .doc(projId)
     .get()
     .then(doc => {
       let projectDs = doc.data();
       projectDs.projId = doc.id;
       return projectDs;
     });

   let scrumMasterData = await this.db
     .collection("scrumMaster")
     .where("userId", "==", project?.
scrum_master_id)
     .get()
     .then(querySnapshot => {
       let masterData = null;
       querySnapshot.forEach(doc => {
         masterData = doc.data();
       });
       return masterData;
     });

   let scrumMemberData = [];
   if (project && project?.participants &&
project?.participants.length > 0) {
     await this.db
       .collection("scrumMember")
       .where("userId", "in", project?.
participants)
       .get()
       .then(querySnapshot => {
         querySnapshot.forEach(doc => {
           let user = {
             email: doc.data().email,
             username: doc.data().username,
             userId: doc.data().userId
           };
           scrumMemberData.push(user);
         });
       });
   }

   project.scrumMember = scrumMemberData;
   project.scrumMaster = scrumMasterData;

   return project;
 }
```

Figure 5.17 : Code segment of retrieving a project

5.2.4.4 Edit Project

This module allows user to edit the project. First, all the projects data is being fetched as shown in the code segment in Figure 5.17. Scrum Master updated the field value and clicks on Save Changes button. Figure 5.19 shows the code segment of edit project. Subsequently, all sprint name and number will be reordered and reset automatically by the application to ensure consistency. Figure 5.20 shows the code segment of reset

sprint order and name. After that, the Scrum Master is redirected to the view project page as shown in Figure 5.15 to see immediate changes.

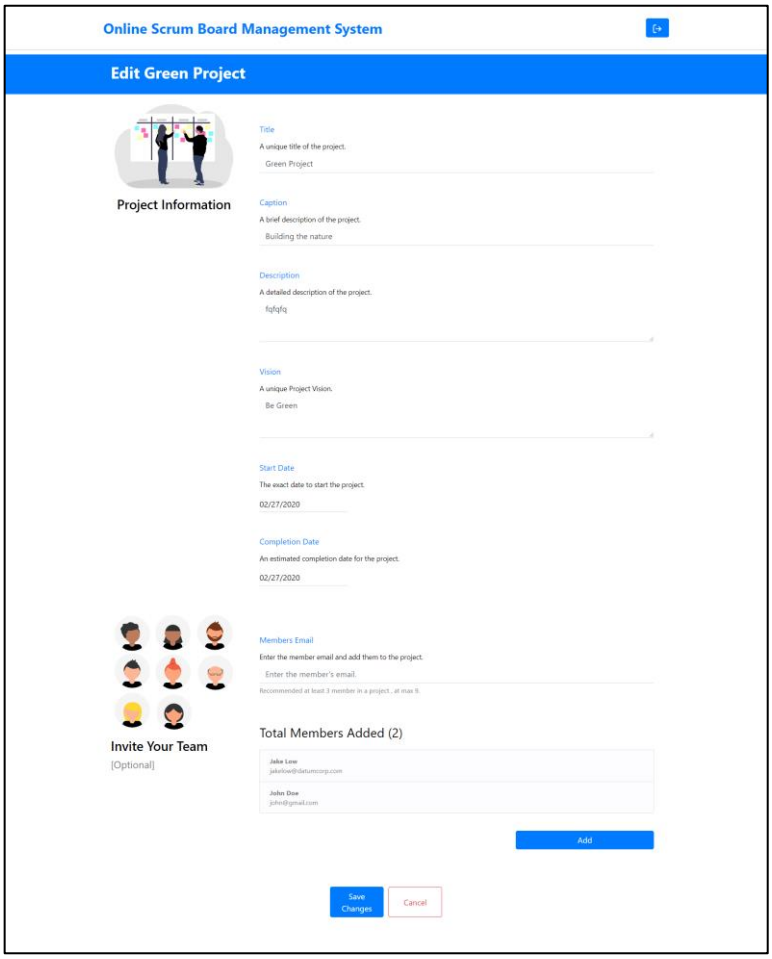


Figure 5.18: Edit Project screen

```
await this.db
    .collection("scrumProjects")
    .doc(projId)
    .update(updatedProject);
```

Figure 5.19: Code segment of updating an existing project

```
//reset sprint Num & sprintName
 async resetExistingSprintOrder(projId, projName
) {
   let db = firebase.firestore();
   let sprints = await this.db
     .collection("sprints")
     .where("projId", "==", projId)
     .get()
     .then(function(querySnapshot) {
       let sprints = [];
       querySnapshot.forEach(doc => {
         let item = {
           sprint_num: doc.data().sprint_num,
           sprint_name: doc.data().sprint_name,
           ref: doc.ref
         };
         sprints.push(item);
       });
       return sprints;
     });

   sprints.sort((a, b) => a.sprint_num - b.
sprint_num);
   var batch = db.batch();

   sprints.forEach((sprint, index) => {
     batch.update(sprint.ref, {
       sprint_name: `${projName.match(/\b(\w)/g
).join("")}-Sprint${index + 1}`,
       sprint_num: index + 1
     });
   });
   return batch.commit();
 }
```

Figure 5.20: Code segment of resetting existing sprint order and name

5.2.4.5 Delete Project

To delete the selected project, Scrum Master clicks on trash button icon on View Project Screen as shown in Figure 5.15. It will delete everything includes the sprints, product backlogs, sprint backlogs and feeds related to the selected project. Figure 5.21 shows the code segment of delete the selected project and it related items.

```
// delete project
  async deleteProject(projId, projName) {
    let db = firebase.firestore();

    //delete all sprint backlogs
    await this.db
      .collection("sprintBacklogs")
      .where("projId", "==", projId)
      .get()
      .then(function(querySnapshot) {
        var batch = db.batch();
        querySnapshot.forEach(function(doc) {
          batch.delete(doc.ref);
        });
        return batch.commit();
      });

    //delete all product backlogs
    await this.db
      .collection("productBacklogs")
      .where("projId", "==", projId)
      .get()
      .then(function(querySnapshot) {
        var batch = db.batch();
        querySnapshot.forEach(function(doc) {
          batch.delete(doc.ref);
        });
        return batch.commit();
      });

    //delete all sprints
    await this.db
      .collection("sprints")
      .where("projId", "==", projId)
      .get()
      .then(function(querySnapshot) {
        var batch = db.batch();
        querySnapshot.forEach(function(doc) {
          batch.delete(doc.ref);
        });
        return batch.commit();
      });

    //delete all sprints
    await this.db
      .collection("archives")
      .where("projId", "==", projId)
      .get()
      .then(function(querySnapshot) {
        var batch = db.batch();
        querySnapshot.forEach(function(doc) {
          batch.delete(doc.ref);
        });
        return batch.commit();
      });

    //delete all feeds
    await this.db
      .collection("feeds")
      .where("projId", "==", projId)
      .get()
      .then(function(querySnapshot) {
        var batch = db.batch();
        querySnapshot.forEach(function(doc) {
          batch.delete(doc.ref);
        });
        return batch.commit();
      });

    await this.db
      .collection("scrumProjects")
      .doc(projId)
      .delete();
```

Figure 5.21: Code segment of deleting an existing project

5.2.4.6 Open and Close project

Every project has an active status to determine whether it is an ongoing, active project or closed project. Scrum Master can easily change the active status by clicking on the open / close project Button on the bottom of the view project screen. Figure 5.22 shows the code segment of updating the project status.

```
async updateProjectStatus(projId, projName,
status) {
  await this.db.collection("scrumProjects").
doc(projId).update({
    active: !status,
  });
```

Figure 5.22: Code segment of updating the status of the project.

5.2.5 Product Backlog Management Module

5.2.5.1 View Product Backlog

This module allows Scrum Master to view all product backlogs created in the selected project as shown in Figure 5.23. By clicking into one of the items from the list, Scrum Master can view the full detail of the selected product backlog item as shown in Figure 5.24. Figure 5.25 shows the code segment of fetching all product backlogs.

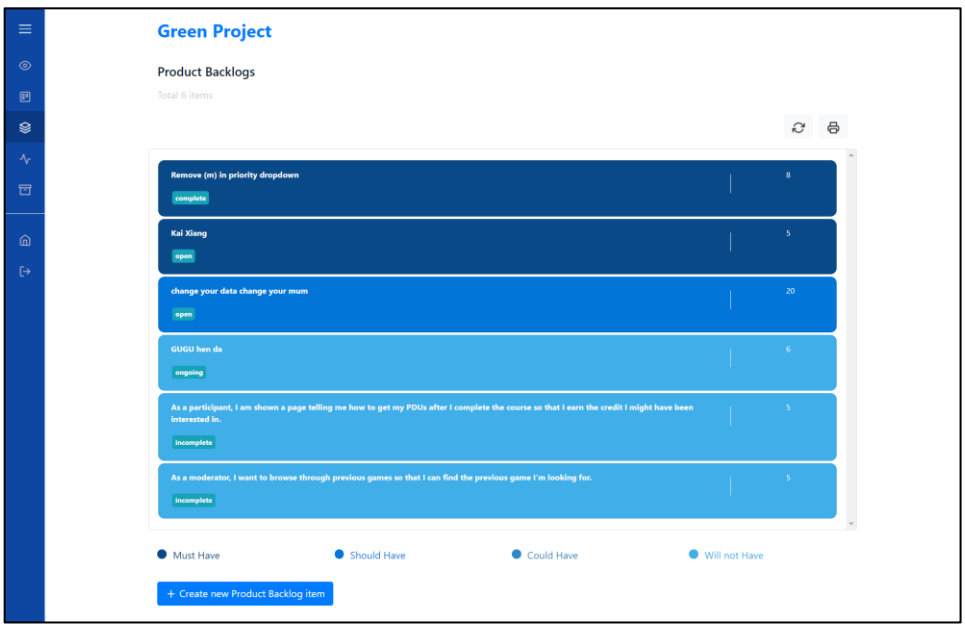


Figure 5.23: View Product Backlogs screen

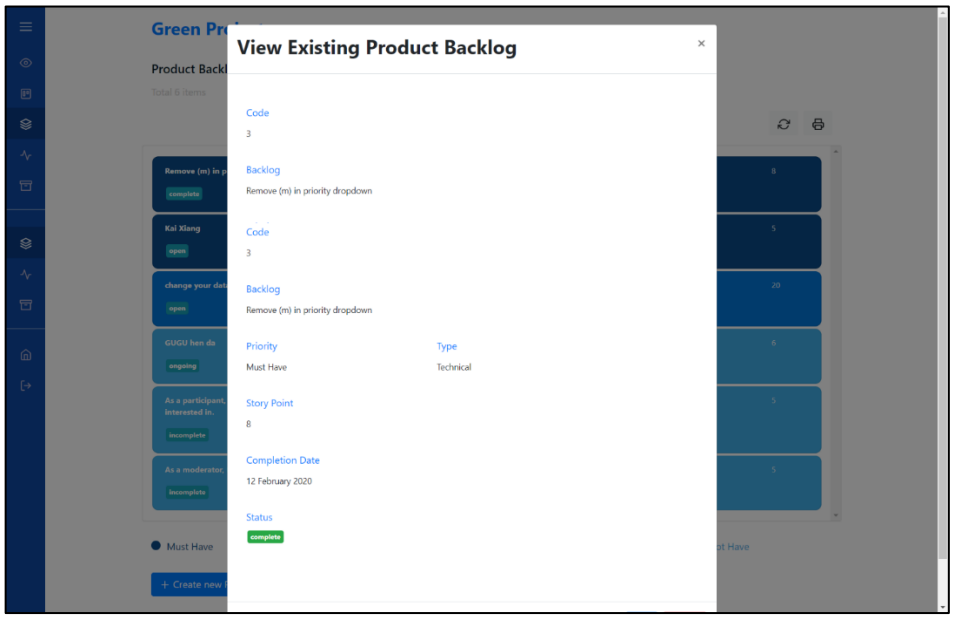


Figure 5.24: View selected Product Backlog item screen

```
//get all product backlog item by projId
 async getProductBacklogsByProjId(projId) {
   //moscow prioritization
   const moscowSort = ["M", "S", "C", "W"];

   return await this.db
     .collection("productBacklogs")
     .where("projId", "==", projId)
     .get()
     .then(querySnapshot => {
       let productBacklogs = [];
       querySnapshot.forEach(doc => {
         let backlog = doc.data();
         backlog.backlogId = doc.id;
         productBacklogs.push(backlog);
       });


//sort by story point number in descending order
       productBacklogs.sort((a, b) => b.
story_point - a.story_point);

       // custom sort using priorization method
       this.customSort({
         data: productBacklogs,
         sortBy: moscowSort,
         sortField: "priority"
       });
       return {
         productBacklogs: productBacklogs,
         total: productBacklogs.length
       };
     });
 }
```

Figure 5.25: Code segment of fetching all product backlogs

5.2.5.2 Create Product Backlog

By clicking on the Create new Product Backlog button, the application will show a modal with the form of creating new product backlog as shown in Figure 5.26. Scrum Master filled in every input fields given and clicks on Save Changes button. Figure 5.27 shows the code segment of create a new product backlog item.

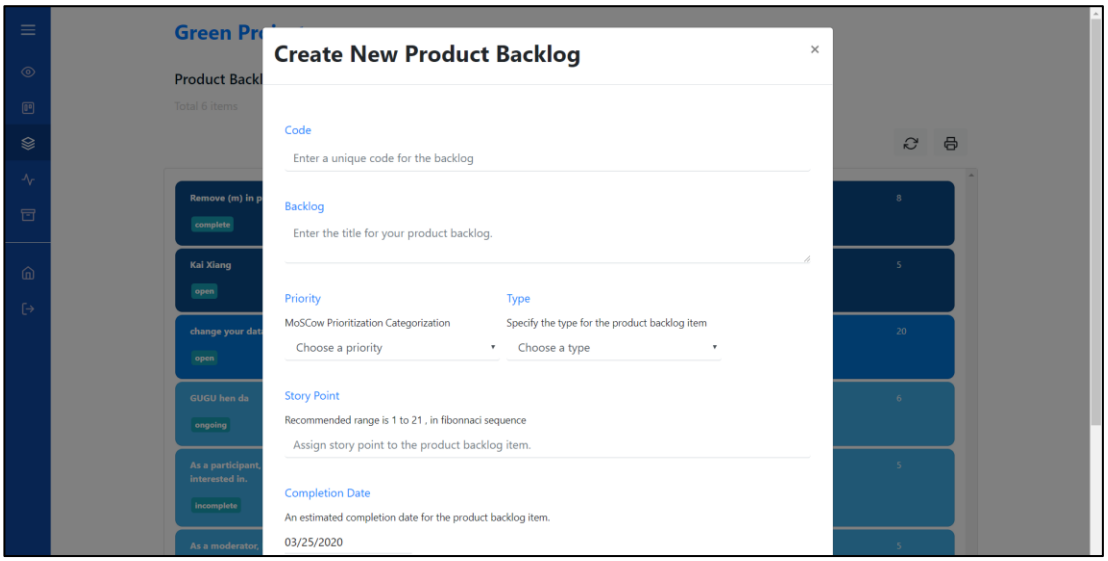


Figure 5.26: Create new Product Backlog item screen

```
await this.db.collection("productBacklogs").add(
newBacklog);
```

Figure 5.27: Code segment of create new Product Backlog item.

5.2.5.3 Edit Product Backlog

After Scrum Master select one of the many product backlog items from the list, Scrum Master clicks on the Edit button on the bottom of modal. The modal will be refreshed

and display the form where Scrum Master could make the changes as shown in Figure 5.28. Scrum Master save the changes made in the form by clicking on Save Changes button. Figure 5.29 shows the code segment of edit an existing product backlog item.

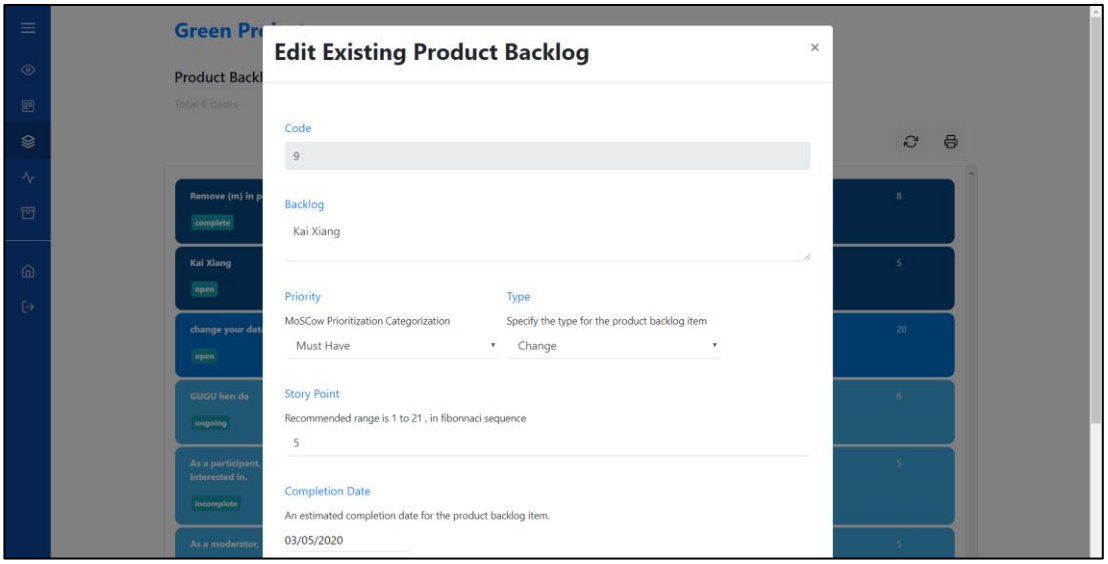


Figure 5.28: Edit existing Product Backlog item screen

```
await this.db
    .collection("productBacklogs")
    .doc(backlogId)
    .set(updatedProductBacklog, { merge: true });
```

Figure 5.29: Code Segment of edit an existing Product Backlog item.

5.2.5.4 Delete Product Backlog

After Scrum Master select one of the many product backlog items from the list, Scrum Master clicks on the Delete button on the bottom of modal. The selected product backlog item will be deleted. Figure 5.30 shows the code segment of delete an existing product backlog item.

```
await this.db
  .collection("productBacklogs")
  .doc(backlogId)
  .delete();
```

Figure 5.30: Code Segment of delete an existing Product Backlog item.

5.2.6 Sprint Backlog Management Module

5.2.6.1 View Sprint Backlog

This module allows Scrum Master to view all sprint backlogs created in the selected project as shown in Figure 5.31. By clicking into one of the items from the list, Scrum Master can view the full detail of the selected sprint backlog item as shown in Figure 5.32. Figure 5.33 shows the code segment of fetching all sprint backlogs.

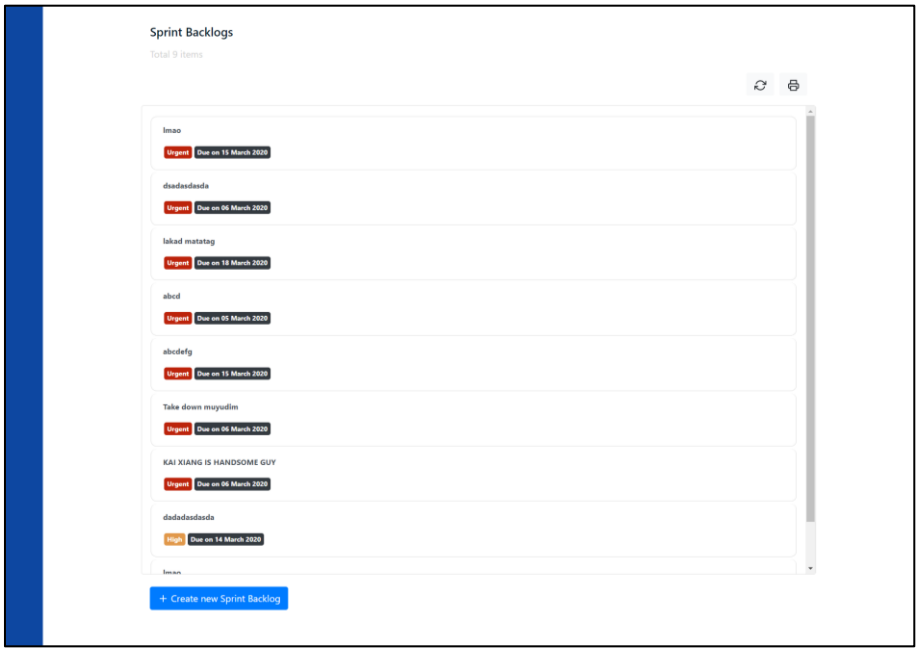


Figure 5.31 : View Sprint Backlogs screen

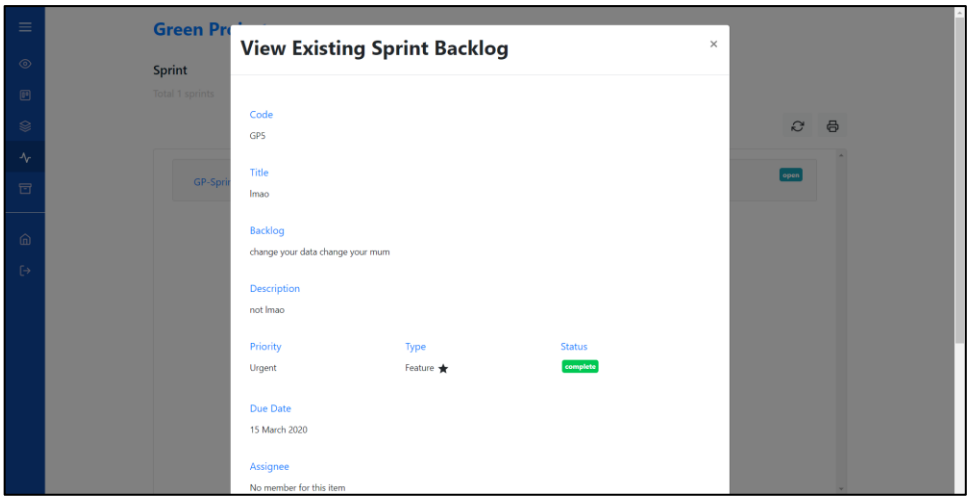
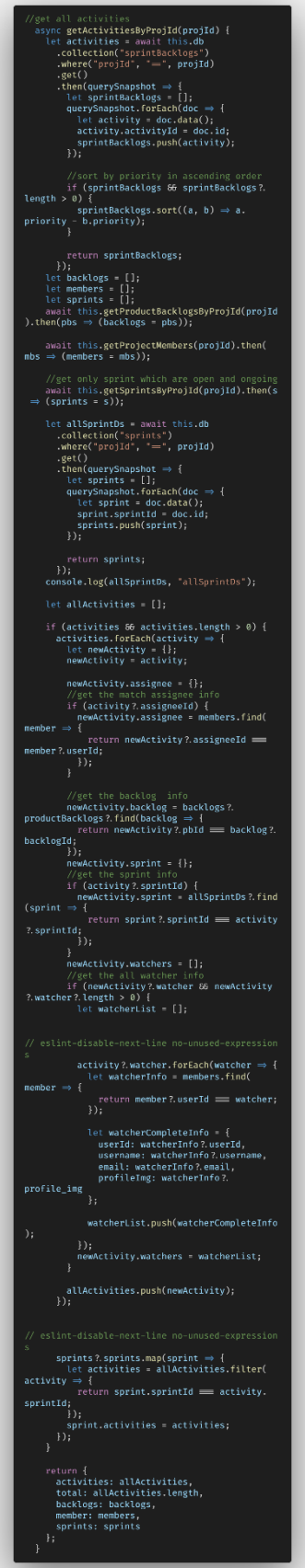


Figure 5.32: View selected Sprint Backlog item screen



```
//get all activities
  async getActivitiesByProjId(projId) {
    let activities = await this.db
      .collection("sprintBacklogs")
      .where("projId", "==", projId)
      .get()
      .then(querySnapshot => {
        let sprintBacklogs = [];
        querySnapshot.forEach(doc => {
          let activity = doc.data();
          activity.activityId = doc.id;
          sprintBacklogs.push(activity);
        });

        //sort by priority in ascending order
        if (sprintBacklogs && sprintBacklogs?.
length > 0) {
          sprintBacklogs.sort((a, b) => a.
priority - b.priority);
        }

        return sprintBacklogs;
      });
    let backlogs = [];
    let members = [];
    let sprints = [];
    await this.getProductBacklogsByProjId(projId
).then(pbs => (backlogs = pbs));

    await this.getProjectMembers(projId).then(
mbs => (members = mbs));

    //get only sprint which are open and ongoing
    await this.getSprintsByProjId(projId).then(s
 => (sprints = s));

    let allSprintDs = await this.db
      .collection("sprints")
      .where("projId", "==", projId)
      .get()
      .then(querySnapshot => {
        let sprints = [];
        querySnapshot.forEach(doc => {
          let sprint = doc.data();
          sprint.sprintId = doc.id;
          sprints.push(sprint);
        });

        return sprints;
      });
    console.log(allSprintDs, "allSprintDs");

    let allActivities = [];

    if (activities && activities.length > 0) {
      activities.forEach(activity => {
        let newActivity = {};
        newActivity = activity;

        newActivity.assignee = {};
        //get the match assignee info
        if (activity?.assigneeId) {
          newActivity.assignee = members.find(
member => {
            return newActivity?.assigneeId ===
member?.userId;
          });
        }

        //get the backlog  info
        newActivity.backlog = backlogs?.
productBacklogs?.find(backlog => {
          return newActivity?.pbId === backlog?.
backlogId;
        });
        newActivity.sprint = {};
        //get the sprint info
        if (activity?.sprintId) {
          newActivity.sprint = allSprintDs?.find
(sprint => {
            return sprint?.sprintId === activity
?.sprintId;
          });
        }
        newActivity.watchers = [];
        //get the all watcher info
        if (newActivity?.watcher && newActivity
?.watcher?.length > 0) {
          let watcherList = [];


// eslint-disable-next-line no-unused-expression
s
          activity?.watcher.forEach(watcher => {
            let watcherInfo = members.find(
member => {
              return member?.userId === watcher;
            });

            let watcherCompleteInfo = {
              userId: watcherInfo?.userId,
              username: watcherInfo?.username,
              email: watcherInfo?.email,
              profileImg: watcherInfo?.
profile_img
            };

            watcherList.push(watcherCompleteInfo
);
          });
          newActivity.watchers = watcherList;
        }

        allActivities.push(newActivity);
      });


// eslint-disable-next-line no-unused-expression
s
      sprints?.sprints.map(sprint => {
        let activities = allActivities.filter(
activity => {
          return sprint.sprintId === activity.
sprintId;
        });
        sprint.activities = activities;
      });
    }

    return {
      activities: allActivities,
      total: allActivities.length,
      backlogs: backlogs,
      member: members,
      sprints: sprints
    };
  }
```

Figure 5.33: Code segment of fetching all sprint backlogs and sprints

5.2.6.2 Create Sprint Backlog

By clicking on the “create new Sprint Backlog” button, the application will show a modal with the form of creating new sprint backlog as shown in Figure 5.34. Scrum Master filled in every input fields given and clicks on Save Changes button. Figure 5.35 shows the code segment of create a new sprint backlog item.

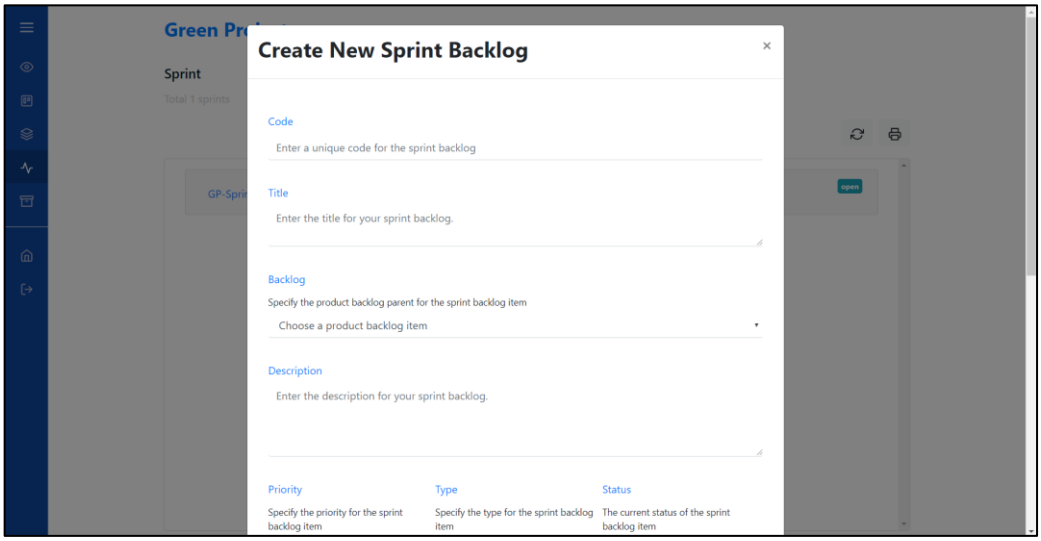


Figure 5.34: Create new Sprint Backlog item screen

```
await this.db
      .collection("sprintBacklogs")
      .add(newActivity)
      .then(doc => {
        return doc.id;
      });
```

Figure 5.35: Code segment of create new Sprint Backlog item.

5.2.6.3 Edit Sprint Backlog

After Scrum Master select one of the many sprint backlog items from the list, Scrum Master clicks on the Edit button on the bottom of modal. The modal will be refreshed and display the form where Scrum Master could make the changes as shown in Figure 5.36. Scrum Master save the changes made in the form by clicking on Save Changes button. Figure 5.37 shows the code segment of edit an existing sprint backlog item.

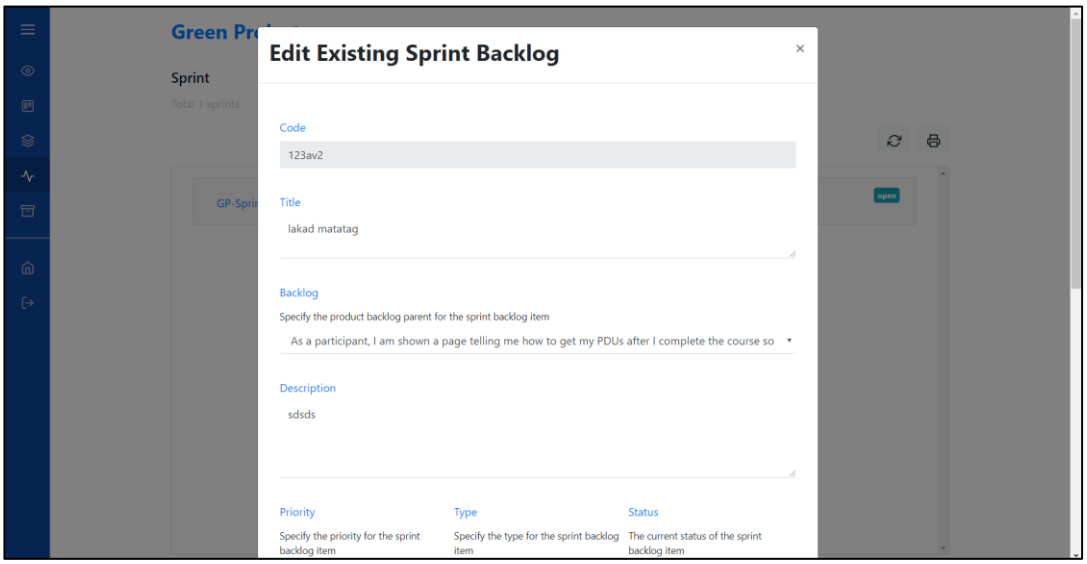


Figure 5.36: Edit existing Sprint Backlog item screen

```
await this.db
      .collection("sprintBacklogs")
      .doc(activityId)
      .set(updatedActivity, { merge: true });
```

Figure 5.37: Code Segment of edit an existing Sprint Backlog item.

5.2.6.4 Delete Sprint Backlog

After Scrum Master select one of the many sprint backlog items from the list, Scrum Master clicks on the Delete button on the bottom of modal. The selected sprint backlog item will be deleted. Figure 5.38 shows the code segment of delete an existing sprint backlog item.

```
await this.db
      .collection("sprintBacklogs")
      .doc(activityId)
      .delete();
```

Figure 5.38: Code Segment of delete an existing Sprint Backlog item.

5.2.7 Sprint Management Module

5.2.7.1 View Sprint

This module allows Scrum Master to view all sprints (only open or ongoing) created in the selected project. By clicking into one of the sprint name from the list, the application will list out all sprint backlogs involves in the sprint (shown in Figure 5.39). By clicking on the accordion header where the sprint name on, Scrum Master can view the full detail of the selected sprint which shown in Figure 5.40. Figure 5.33 shows the code segment of fetching all sprints data.

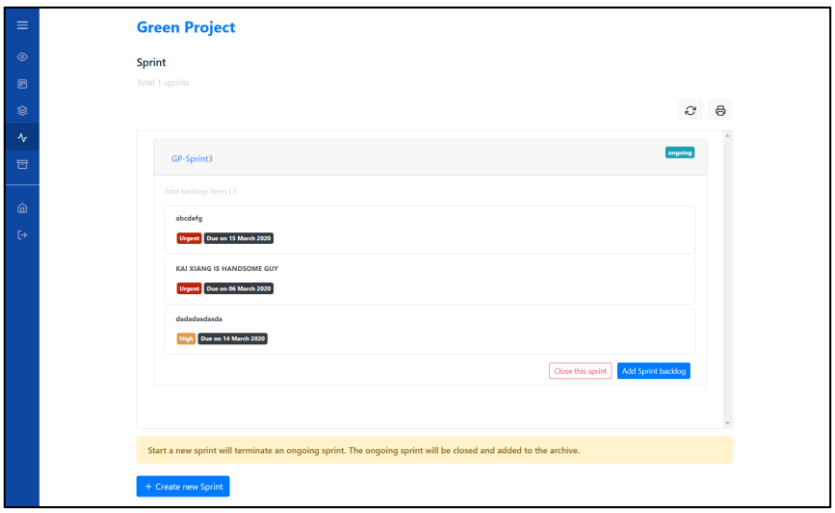


Figure 5.39: View Sprints screen

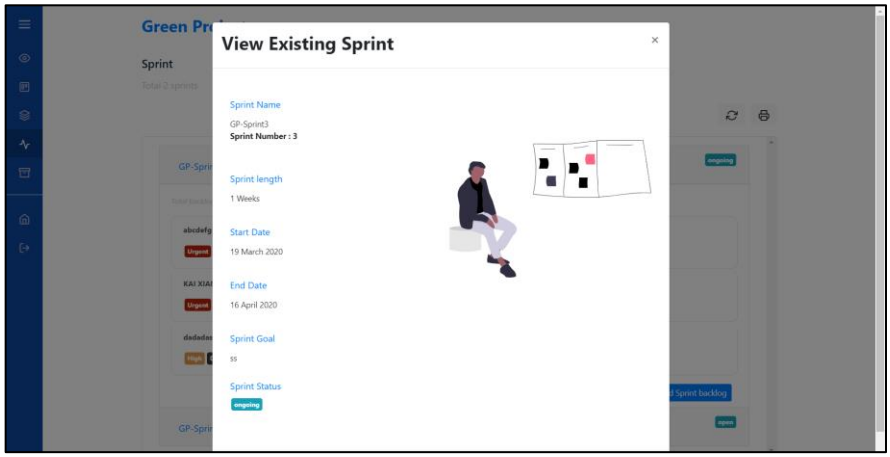


Figure 5.40: View selected Sprint screen

5.2.7.2 Create Sprint

By clicking on the "create new Sprint" button, the application will show a modal with the form of creating new sprint as shown in Figure 5.41. Scrum Master filled in every input fields given and clicks on Save Changes button. Figure 5.42 shows the code segment of create a new sprint.

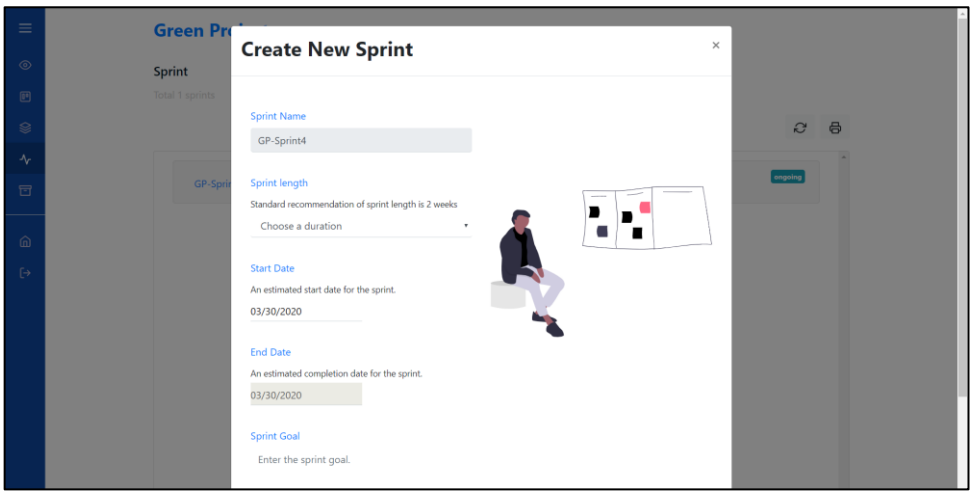


Figure 5.41: Create new Sprint screen

```
await this.db.collection("sprints").add(newSprint);
```

Figure 5.42: Code segment of create new sprint.

5.2.7.3 Edit Sprint

Scrum Master could only edit an open sprint which is a sprint that has not been started. After clicking on the accordion header where the sprint name on, Scrum Master clicks on the Edit button on the bottom of modal. The modal will be refreshed and display the form where Scrum Master could make the changes as shown in Figure 5.43. Scrum Master save the changes made in the form by clicking on Save Changes button. Figure 5.44 shows the code segment of edit an existing sprint backlog item.

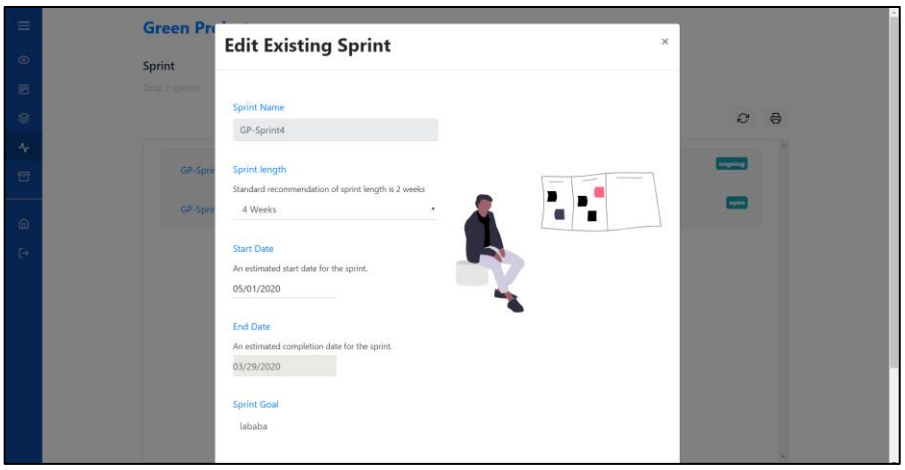


Figure 5.43: Edit existing Sprint screen

```
this.db
      .collection("sprints")
      .doc(sprintId)
      .set(updatedSprint, { merge: true });
```

Figure 5.44: Code Segment of edit an existing Sprint.

5.2.7.4 Delete Sprint

There are two delete options which is either delete the sprint itself or delete the sprint and all its sprint backlog items. After Scrum Master select on one of the accordion headers with sprint name on it, Scrum Master clicks on the Delete button on the bottom of modal to delete the sprint itself only. Scrum Master can also click on the Delete Everything button on the bottom of modal to delete everything not only the sprint itself

and all its sprint backlog items. Figure 5.45 shows the code segment of delete an existing sprint.

```
async deleteExistingSprint(sprintId, isDeleteAll) {
  let db = firebase.firestore();
  await this.db
    .collection("sprints")
    .doc(sprintId)
    .delete();

  if (isDeleteAll) {
    await this.db
      .collection("sprintBacklogs")
      .where("sprintId", "==", sprintId)
      .get()
      .then(function(querySnapshot) {
        var batch = db.batch();
        querySnapshot.forEach(function(doc) {
          batch.delete(doc.ref);
        });
        return batch.commit();
      });
  }
}
```

Figure 5.45: Code Segment of delete an existing Sprint.

5.2.7.5 Start and Terminate Sprint

This module allows Scrum Master to start sprint and close sprint. Scrum Master can start a sprint which has an “open” status as shown in Figure 5.46. However, the application will automatically terminate the ongoing sprint if there is one. Starting a new sprint is applicable when it meets the following conditions:

- It is the sprint with the subsequent sprint number (which is a good practice to start a sprint accordingly for new scrum practitioner)
- The sprint start date should not be behind today date.

The terminated sprint will be changed into “close” status and documented as an archive. Moreover, Scrum Master can terminate sprint which has an “ongoing” status as shown in Figure 5.47. Figure 5.48 shows the code segment of starting a new sprint. Figure 5.49 shows the code segment of terminating an ongoing sprint. Figure 5.50 shows the code segment of archiving a recent terminated sprint.

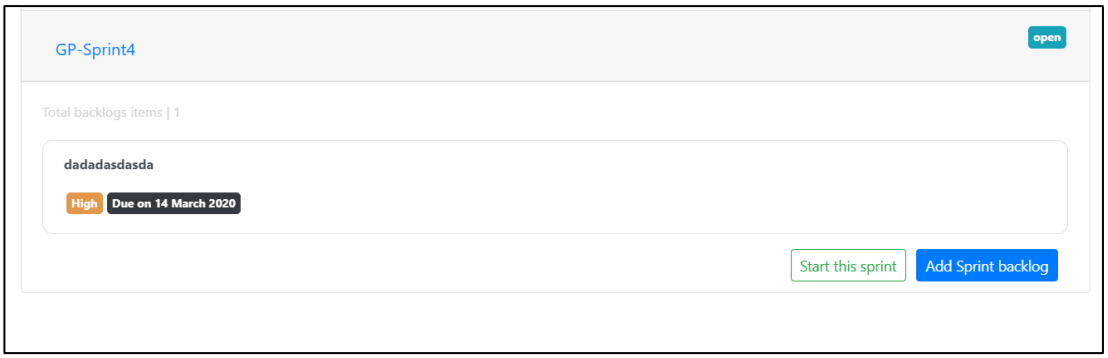


Figure 5.46: Start a new sprint screen

Figure 5.47: Terminate an ongoing sprint screen

```
async startProjectSprint(latestSprintId, projId, sprintName) {
   // update status of the sprint
   await this.db
     .collection("sprints")
     .doc(latestSprintId)
     .update({
       status: "ongoing"
     });

   //add latest sprint id to the project
   await this.db
     .collection("scrumProjects")
     .doc(projId)
     .set(
       {
         curr_sprint_id: latestSprintId
       },
       { merge: true }
     );
```

Figure 5.48: Code Segment of starting a new Sprint.

```
async closeProjectSprint(closeSprintId, projId, sprintName) {
  // update status of the sprint
  await this.db
    .collection("sprints")
    .doc(closeSprintId)
    .update({
      status: "close"
    });

  await this.db
    .collection("scrumProjects")
    .doc(projId)
    .update({
      curr_sprint_id: ""
    });
```

Figure 5.49: Code Segment of terminating an ongoing Sprint.

```
async createArchivefromCloseSprint(closeSprint) {
    let archiveActivities = closeSprint.activities;
    delete closeSprint.activities;
    delete closeSprint.status;
    let archive = { ...closeSprint, archive_at: new Date().toISOString() };

    let completeArchiveActivities = [];
    if (archiveActivities && archiveActivities?.length > 0) {
      archiveActivities.map(activity => {
        let archiveActivityItem = activity;
        delete archiveActivityItem?.sprint;
        completeArchiveActivities.push(archiveActivityItem);
      });
    }
    console.log("add in archive activities success");
    console.log(archive);

    let archiveDocRefId = await this.db
      .collection("archives")
      .add(archive)
      .then(docRef => {
        return docRef.id;
      });

    console.log(archiveDocRefId, "add in archive doc ref");

    const batch = this.db.batch();

    completeArchiveActivities.map(archiveItemActivity => {
      var subActivityCollection = this.db
        .collection("archives")
        .doc(archiveDocRefId)
        .collection("activities")
        .doc();

      console.log(archiveItemActivity);
      batch.set(subActivityCollection, archiveItemActivity);
    });

    return batch.commit();
  }
```

Figure 5.50: Code Segment of archiving a recent terminated Sprint.

5.2.8 View Archive Module

Archives are the collection of previous sprint records and sprint backlogs documents. In Figure 5.52, Scrum Master can view the detail of a closed sprint by clicking on one of many sprint names provided in the list. When one the sprint names is selected, the jumbotron above will show the details of the sprint, provides the option to generate reports as PDF or Excel and move all incomplete (ongoing and toDo) sprint backlog items to the next sprint. Most importantly, Scrum Master can also view the scrum board of the previous sprint as shown in Figure 5.53. Moreover , Scrum Master could view on the details of the sprint backlog items of an archive by click on one of the sprint backlog item in list of the selected archive as shown in Figure 5.54 .Figure 5.55 shows the code segment of fetching all archives documents and Figure 5.57 shows the code segment of moving the all incomplete sprint backlog items to the next sprint.

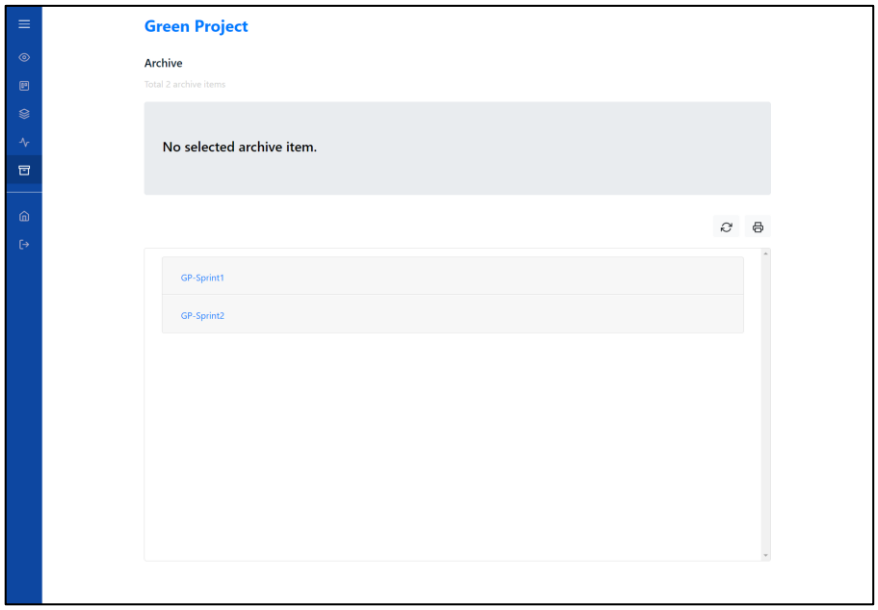


Figure 5.51: View archives screen

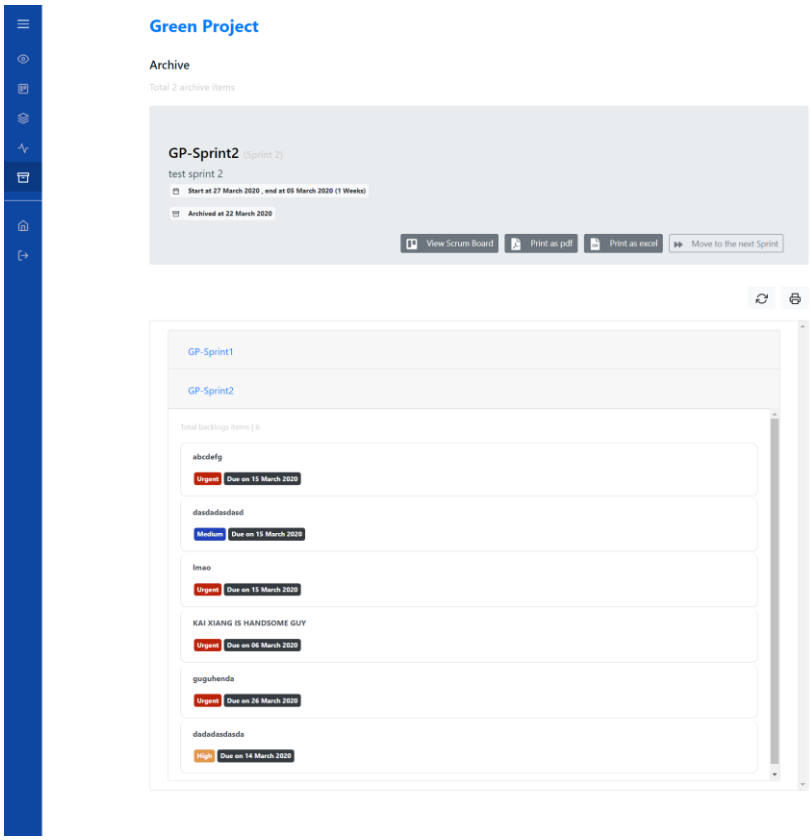


Figure 5.52: View archives screen when one of the sprint name title has been clicked on.

Figure 5.53: View Scrum Board of previous sprint screen

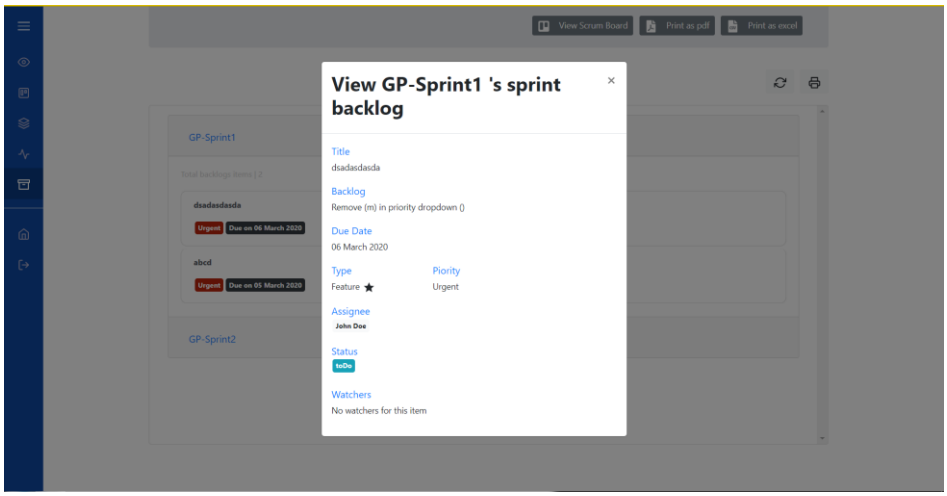


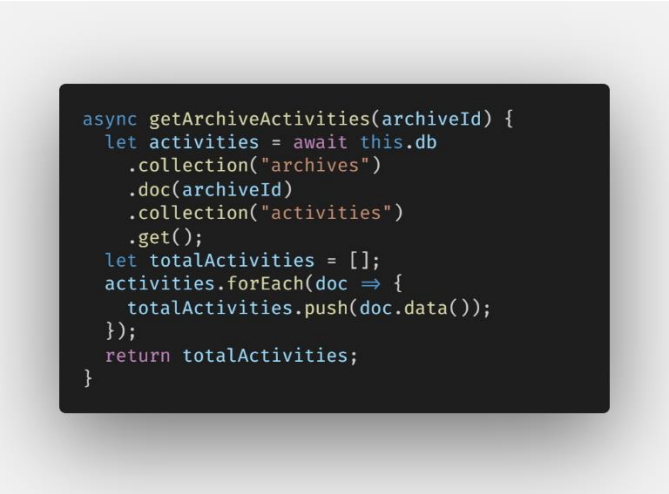
Figure 5.54: View Selected sprint backlog items of previous sprint screen.

```
/**
  *  Get Archive
  */
 async getArchive(projId) {
   let archives = await this.db
     .collection("archives")
     .where("projId", "==", projId)
     .get();
   let archiveDs = [];
   archives.forEach(doc => {
     let archiveItem = doc.data();
     archiveItem.archiveId = doc.id;
     archiveDs.push(archiveItem);
   });
   var latestSprintRef = await this.db.
collection("sprints");
   latestSprintRef = latestSprintRef.where(
"projId", "==", projId);
   latestSprintRef = latestSprintRef.where(
     "sprint_num",
     "==",
     archiveDs?.length + 1
   );
   let latestSprintId = null;
   let latestSprintName = null;

   await latestSprintRef.get().then(
querySnapshot => {
     querySnapshot.forEach(doc => {
       latestSprintName = doc.data()?.
sprint_name;
       latestSprintId = doc.id;
     });
   });
   for (let i = 0; i < archiveDs.length; i
++) {
     let activities = await this.
getArchiveActivities(archiveDs[i].archiveId);
     archiveDs[i].activities = activities;
   }

   return {
     archives: archiveDs,
     total: archiveDs.length,
     latestSprintId: latestSprintId,
     latestSprintName: latestSprintName
   };
 }
```

Figure 5.55: Code Segment of fetching all archives data



```
async getArchiveActivities(archiveId) {
  let activities = await this.db
    .collection("archives")
    .doc(archiveId)
    .collection("activities")
    .get();
  let totalActivities = [];
  activities.forEach(doc ⇒ {
    totalActivities.push(doc.data());
  });
  return totalActivities;
}
```

Figure 5.56: Code Segment of fetching all archive's activities

```
async moveArchivedActivitiesToLatestSprint(
    archivedActivities,
    latestSprintId
  ) {
    console.log(archivedActivities,
latestSprintId);

// eslint-disable-next-line array-callback-re
turn

    if (archivedActivities &&
archivedActivities?.length > 0) {
      var batch = this.db.batch();

      await Promise.all(
        archivedActivities.map(async activity
 ⇒ {
          var activityRef = await firebase
            .firestore()
            .collection("sprintBacklogs")
            .doc(activity);

          await activityRef.get().then(doc ⇒
 {
            if (doc.exists) {
              batch.update(activityRef, {
sprintId: latestSprintId });
              console.log(doc.exists,
"check if exists");
            }
          });
        })
      );
      return batch.commit();
    }
  }
```

Figure 5.57: Code Segment of moving incomplete sprint backlog items of an archive to the next sprint.

5.2.9 View Scrum Board Module

This Module allows Scrum Master to view the Scrum Board of ongoing sprint and its tasks / activities (sprint backlog items). There are total two different view of scrum board which are scrum board that display sprint's tasks that group by status and scrum board that display sprint's tasks that group by product backlogs which also known as user stories at here. Figure 5.58 shows the board view of the scrum board which the sprint tasks are assigned to swim lanes of toDo, Ongoing and complete. Figure 5.59 shows the exact same thing but in list view. On the other hand, figure 5.60 shows the list view of the scrum board which the sprint tasks are grouped by unique user story. Figure 5.61 shows the code segment of fetching the current scrum board data. Moreover, Scrum Master can change the status of the task by dragging the item and dropping the item to swim lane where it intended to be.

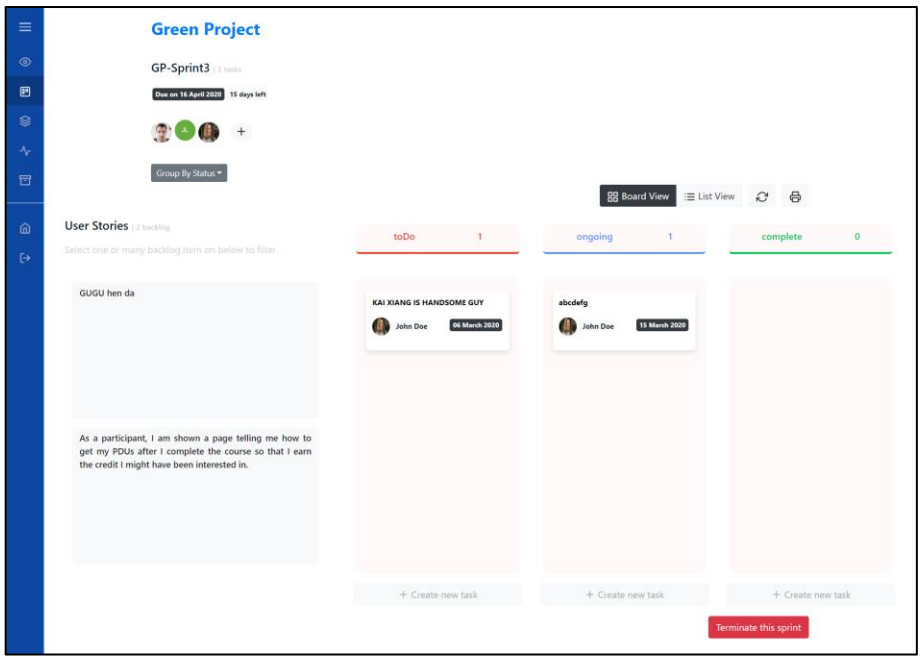


Figure 5.58: Scrum Board of the current Sprint (group by status) screen.

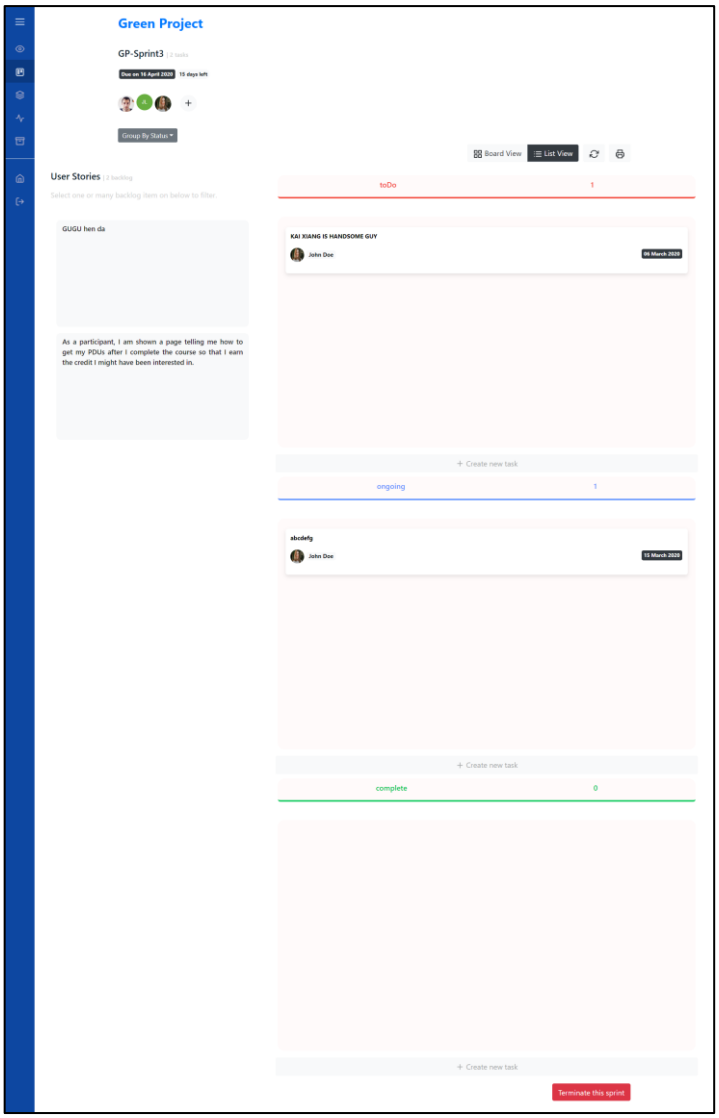


Figure 5.59: Scrum Board of the current Sprint (group by status) screen.

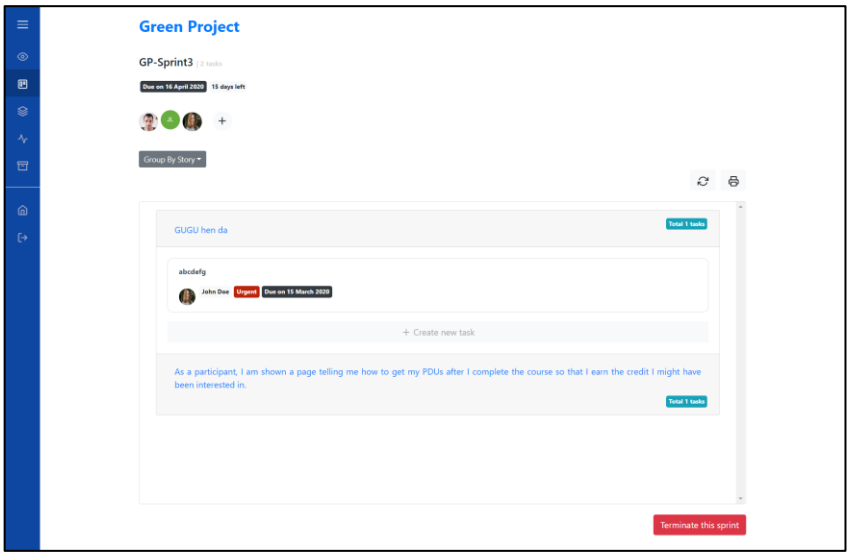


Figure 5.60: Scrum Board of the current Sprint (group by story) screen.

```
// Get current sprint of project
async getCurrentScrumBoard(projId) {
  let project = await this.db
    .collection("scrumProjects")
    .doc(projId)
    .get()
    .then(docRef => {
      return docRef.data();
    });
  let backlogs = await this.getProductBacklogsByProjId(projId);

  let currentSprintId = project?.curr_sprint_id;

  let participants = [];
  if (project?.participants && project?.participants?.length > 0) {
    participants = await this.db
      .collection("scrumMember")
      .where("userId", "in", project?.participants)
      .get()
      .then(querySnapshot => {
        let memberData = [];
        querySnapshot.forEach(doc => {
          let user = {
            email: doc.data().email,
            username: doc.data().username,
            userId: doc.data().userId,
            profileImg: doc.data().profile_img
          };
          memberData.push(user);
        });
        return memberData;
      });
  }

  if (currentSprintId) {
    let currSprint = await this.db
      .collection("sprints")
      .doc(currentSprintId)
      .get()
      .then(docRef => {
        let currSprint = docRef.data();
        currSprint.sprintId = currentSprintId;
        return currSprint;
      });

    let currSprintActivities = await this.db
      .collection("sprintBacklogs")
      .where("sprintId", "==", currentSprintId)
      .get()
      .then(querySnapshot => {
        let sprintBacklogs = [];
        querySnapshot.forEach(doc => {
          let activity = doc.data();
          activity.activityId = doc.id;
          activity.assignee = {};
          if (activity?.assigneeId) {
            activity.assignee = participants.find(member => {
              return member.userId === activity?.assigneeId;
            });
          }
          activity.watchers = [];
          if (activity?.watcher && activity?.watcher?.length > 0) {
            activity.watchers = activity.watcher.map(watcherId => {
              let watching = participants.find(member => {
                return member?.userId === watcherId;
              });
              return watching;
            });
          }
          activity.backlog = backlogs?.productBacklogs?.find(backlog => {
            return activity?.pbId === backlog?.backlogId;
          });
          sprintBacklogs.push(activity);
        });
        return sprintBacklogs;
      });

    let allSprintBacklogs = await this.db
      .collection("sprintBacklogs")
      .where("projId", "==", projId)
      .get()
      .then(querySnapshot => {
        let allSprintBacklogs = [];
        querySnapshot.forEach(doc => {
          let activity = doc.data();
          activity.activityId = doc.id;
          allSprintBacklogs.push(activity);
        });
        return allSprintBacklogs;
      });

    return {
      currSprint: currSprint,
      currActivities: currSprintActivities,
      participants: participants,
      backlogs: backlogs,
      sprintBacklogs: allSprintBacklogs
    };
  } else return {};
}
```

Figure 5.61: Code Segment of fetching the current Scrum Board data

5.2.9.1 Remove Activity from the sprint

Inside the scrum board, Scrum Master also allow to remove the sprint backlog item from the ongoing sprint by clicking on the “Remove from Sprint name” button on the

bottom of the modal as shown in Figure 5.62. Figure 5.63 shows the code segment of removing the sprint backlog item from the ongoing sprint.

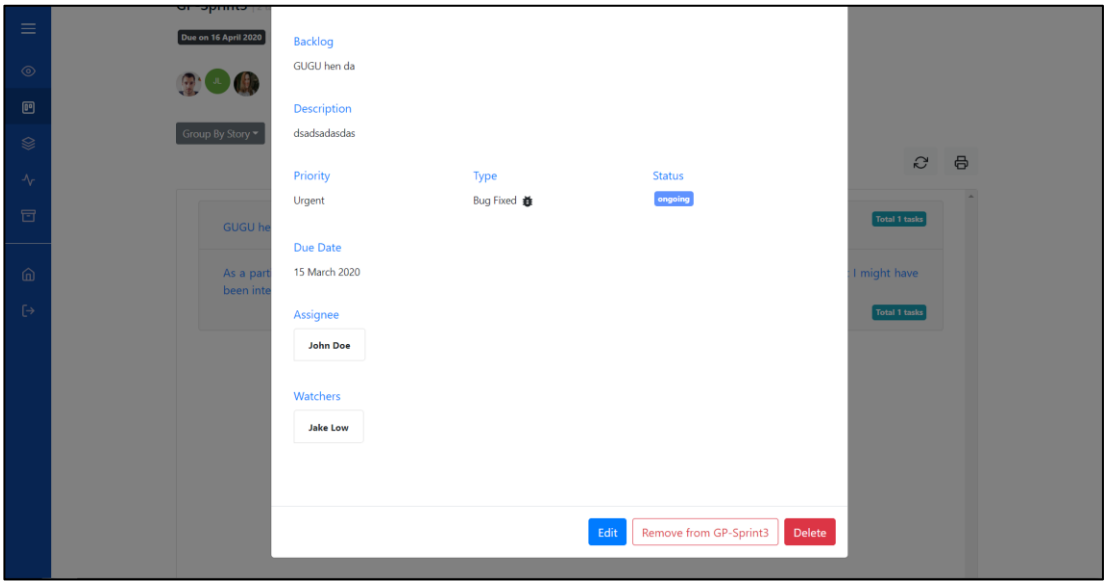


Figure 5.62: Remove Sprint Backlog item from the current sprint screen.

```
// remove activity from the sprint
  async removeActivityFromSprintByActivityId(activityId) {
    await this.db
      .collection("sprintBacklogs")
      .doc(activityId)
      .update({
        sprintId: ""
      });
  }
```

Figure 5.63: Code Segment of removing the Sprint Backlog item from the current sprint.

5.2.9.2 Invite or Remove Scrum Member

This module allows Scrum Master to invite and remove scrum members. On the top of the scrum board, it shows a list of scrum members who are participating in the project, which is shown in Figure 5.64. By clicking on the plus button, the application pops out a modal as shown in Figure 5.65. It works similar as what Scrum Master has done in creating a new project screen as shown in Figure 5.11.

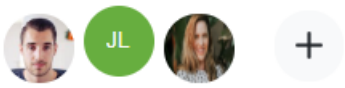


Figure 5.64: List of scrum members in the project.

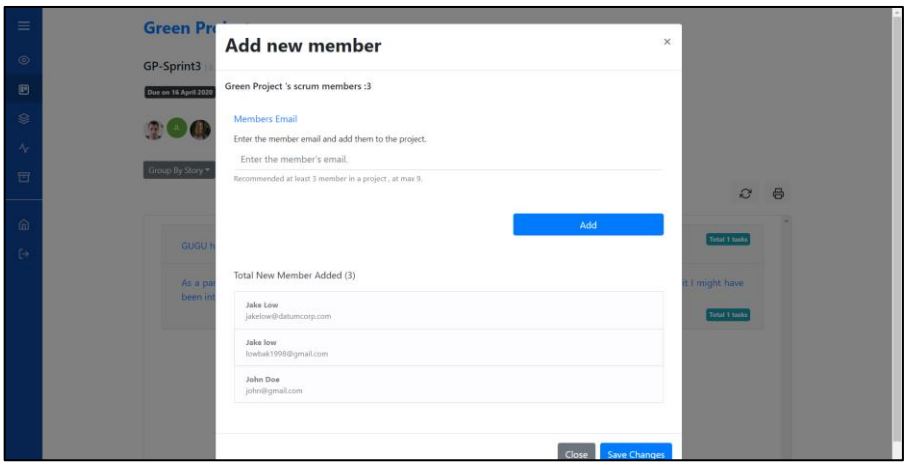


Figure 5.65: Invite new scrum members or remove existing scrum member modal.

5.2.10 Generate Reports Module

Scrum Master able to generate reports by clicking on the printer icon as shown in Figure 5.66. There are two types of file extension can be exported which are PDF and excel. Figure 5.67 shows a sample pdf product backlogs report. Figure 5.68 shows a sample excel product backlogs report.

Figure 5.66: Printer Icon.

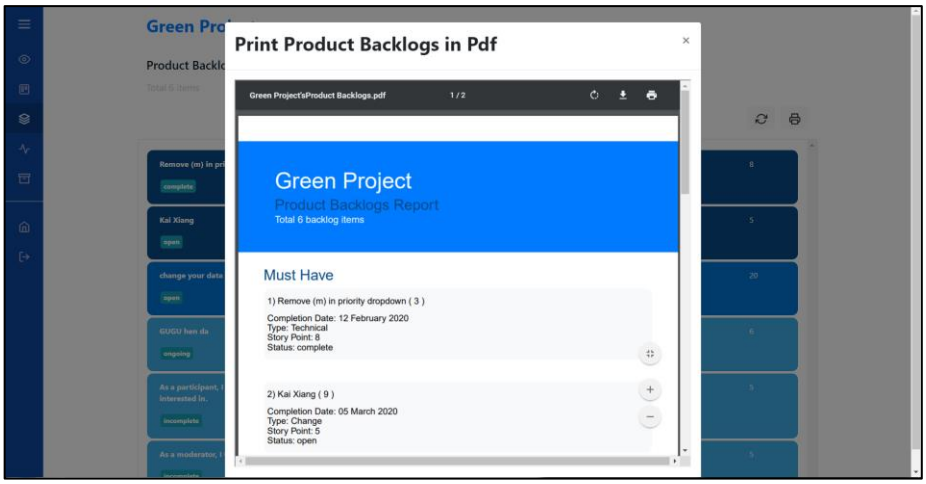


Figure 5.67: Pdf report sample.

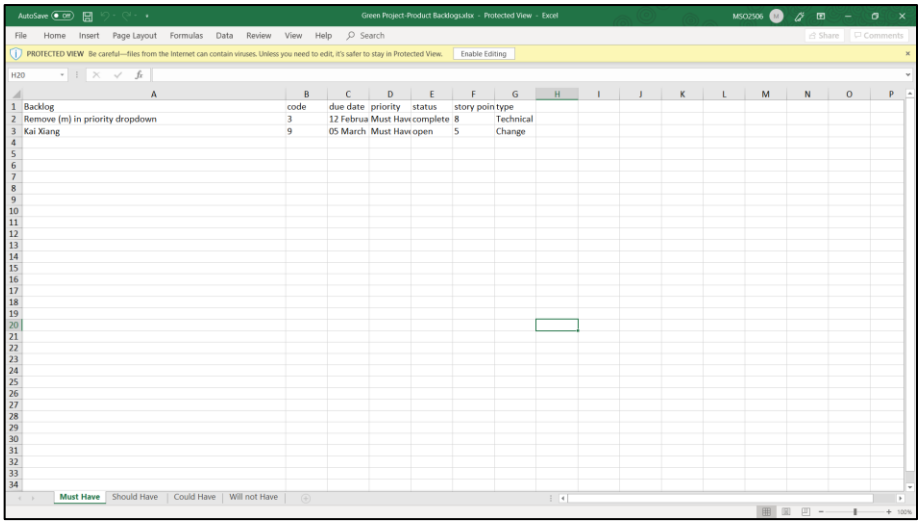


Figure 5.68: Excel report sample.

5.2.11 Profile Management Module

In the project dashboard, Scrum Master can edit his/ her profile image by clicking on the avatar at the end of the navigation bar. When the avatar is clicked on, a modal will pop out, showing all the profile information of the Scrum Master. Scrum Master clicks on “Change Profile Picture” and select the image he/she wants. Finally, clicks on the “Upload” button to update his/her profile. Figure 5.71 shows the code segment of uploading new profile image.

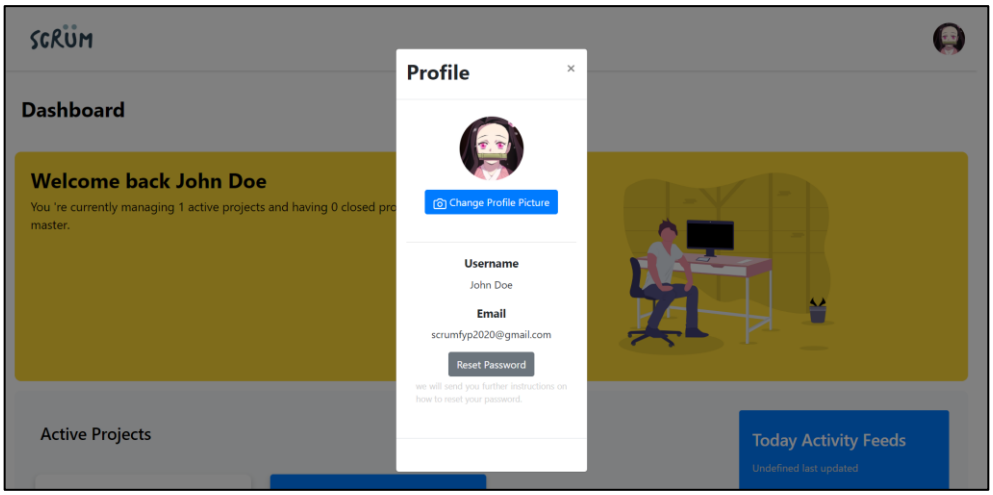


Figure 5.69: Profile screen.

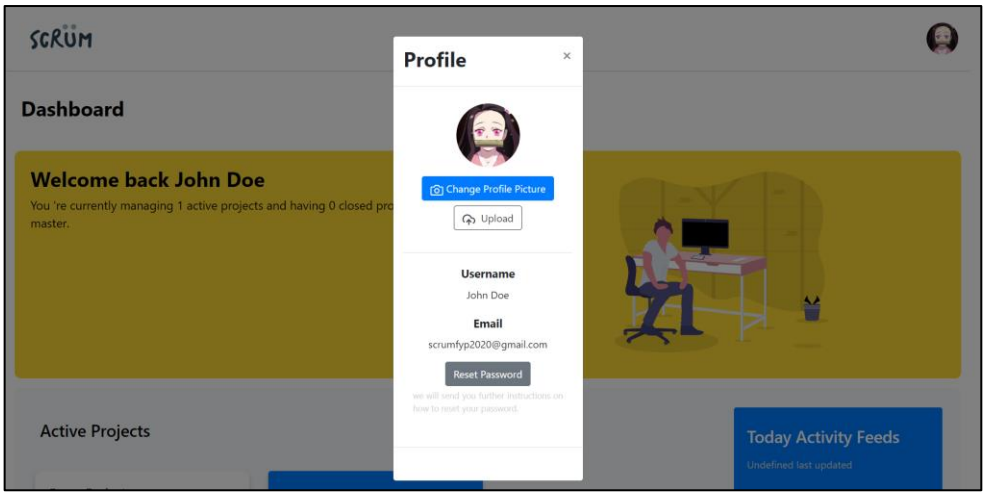


Figure 5.70: Profile with upload button available screen.

```
async uploadProfileImg(img, userId, type, authUser) {
   var metadata = {
     contentType: type
   };
   var storageRef = await firebase.storage().ref();
   var userProfileRef = storageRef.child(`${userId}`);
   await userProfileRef.put(img, metadata);

   let url = await firebase
     .storage()
     .ref(userProfileRef.name)
     .getDownloadURL();

   console.log(url, "url");
   if (url) {
     authUser.photoURL = url;
     localStorage.setItem("authUser", JSON.stringify(authUser));

     await this.auth.currentUser.updateProfile({
       photoURL: url
     });

     let scrumMasterDocRef = await this.db
       .collection("scrumMaster")
       .where("userId", "==", userId)
       .get()
       .then(querySnapshot => {
         let ref = "";
         querySnapshot.forEach(doc => {
           ref = doc.id;
         });
         return ref;
       });

     console.log(scrumMasterDocRef, "scrumMasterDocRef");

     return await this.db
       .collection("scrumMaster")
       .doc(scrumMasterDocRef)
       .update({
         profile_img: url
       });
   }
 }
```

Figure 5.71: Code Segment of upload new profile image.

5.2.12 View Notification and feeds Module

The module shown in Figure 5.72 shows the list of activity feeds that made by the scrum members from the scrum board mobile application. Figure 5.73 shows the toast box notifications on every change made by Scrum Master. Figure 5.74 shows the code segment of fetching all feeds data.

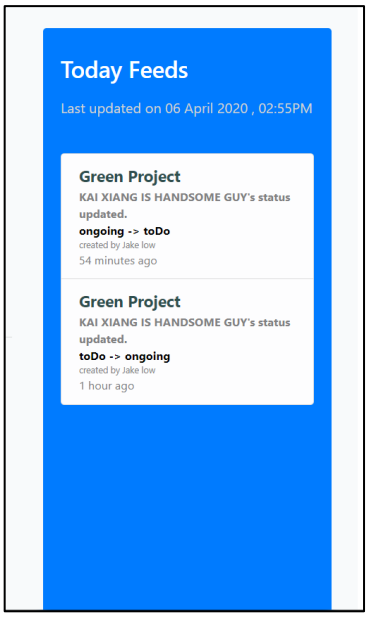


Figure 5.72: Feed list in the project dashboard.

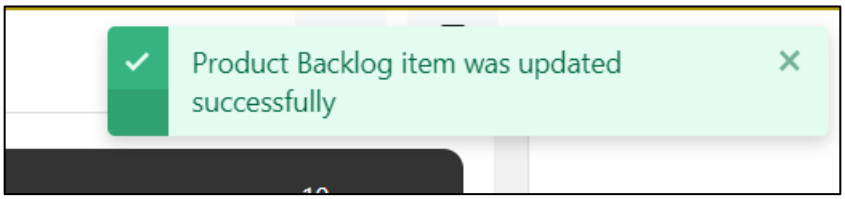


Figure 5.73: toast box notification.

```
async getFeeds(projects) {
  let feeds = [];
  if (projects && projects?.length > 0) {
    let projId = projects.map((project) => {
      return project?.projId;
    });
    let scrumMembers = await this.db
      .collection("scrumMember")
      .get()
      .then((querySnapshot) => {
        let memberData = [];
        querySnapshot.forEach((doc) => {
          let member = doc.data();
          member.ref = doc.id;
          memberData.push(member);
        });
        return memberData;
      });

    feeds = await this.db
      .collection("memberFeeds")
      .where("projId", "in", projId)
      .get()
      .then((querySnapshot) => {
        let feedData = [];
        querySnapshot.forEach((doc) => {
          let feed = doc.data();
          feed.feedId = doc.id;

          if (this.checkIfSameDay(feed?.utc
)) {
            feed.user = scrumMembers.find((
member) => {
              return member?.userId === feed
?.userId;
            });
            feed.projTitle = projects.find((
project) => {
              return feed?.projId === project
?.projId;
            })?.projTitle;
            feedData.push(feed);
          }
        });
        return feedData;
      });

    feeds = feeds.sort((a, b) => new Date(b?.
utc) - new Date(a?.utc));
  }
  const lastUpdate = feeds[0]?.utc || null;

  let activityFeeds = { lastUpdate:
lastUpdate, feeds: feeds };
  return activityFeeds;
}
```

Figure 5.74: Code Segment of fetch feeds.

5.3 Modules for scrum board mobile application

Figure 5.75 shows the welcome screen where Scrum Member can choose to login to the application using existing account and register a new account. Scrum member presses on the Login button to navigate to the login screen. Scrum member presses on the Sign-Up button to navigate to the register screen.

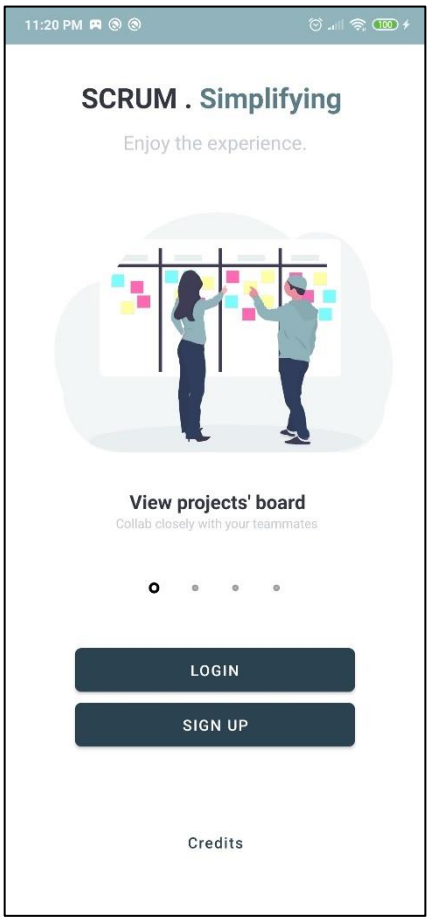


Figure 5.75: Welcome screen.

5.3.1 Login Module

Scrum member enters the valid credentials and the firebase authentication will help to check if the credentials entered matched with the user database. If the credential entered are compliance with the data in the database, it will automatically redirect Scrum Member to his / her project dashboard or else, an error message will be shown.

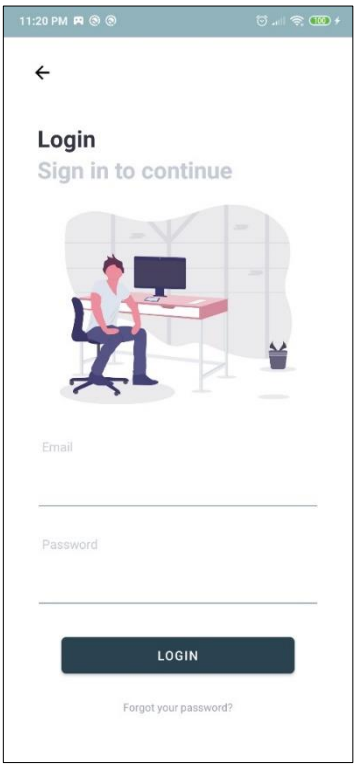
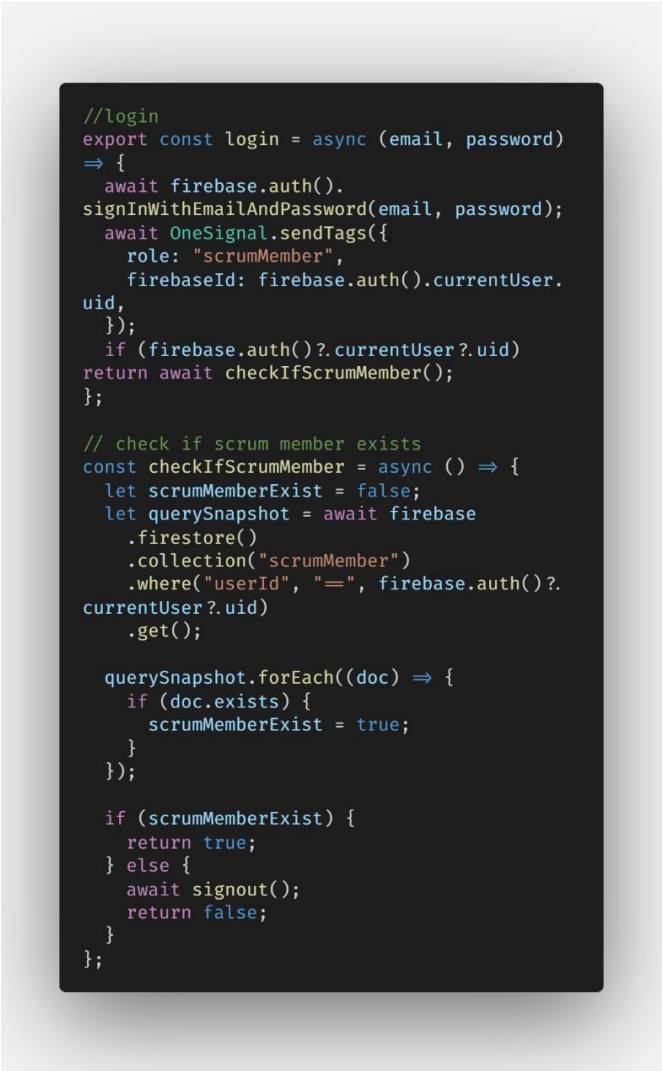


Figure 5.76: Login screen.



```
//login
export const login = async (email, password)
⇒ {
  await firebase.auth().
signInWithEmailAndPassword(email, password);
  await OneSignal.sendTags({
    role: "scrumMember",
    firebaseId: firebase.auth().currentUser.
uid,
  });
  if (firebase.auth()?.currentUser?.uid)
return await checkIfScrumMember();
};

// check if scrum member exists
const checkIfScrumMember = async () ⇒ {
  let scrumMemberExist = false;
  let querySnapshot = await firebase
    .firestore()
    .collection("scrumMember")
    .where("userId", "==", firebase.auth()?.
currentUser?.uid)
    .get();

  querySnapshot.forEach((doc) ⇒ {
    if (doc.exists) {
      scrumMemberExist = true;
    }
  });

  if (scrumMemberExist) {
    return true;
  } else {
    await signout();
    return false;
  }
};
```

Figure 5.77: Code segment of the login verification by firebase authentication

5.3.2 Register Module

The module is to allow scrum member to create a new account with a unique email and password, along with the intended username to be used and viewed by the scrum master.

How does the scrum member create a new account? Scrum member must enter valid information to the input fields given and presses on the Register button. Figure 5.79 shows the code segment of register a new scrum member. The information will be sent to the firebase authentication and new user will be registered. Once it was done, a new scrum member profile will be automatically created as document inside the scrumMember collection. Then, Scrum member will be redirected back to the welcome screen.

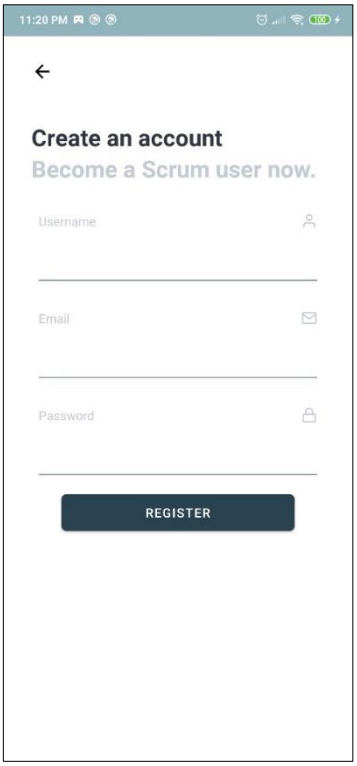


Figure 5.78: Register screen.

```
//register
export const register = async (username,
email, password) => {
  await firebase.auth().
createUserWithEmailAndPassword(email,
password);
  await firebase.auth().currentUser.
updateProfile({
    displayName: username,
  });
  const user = firebase.auth().currentUser;
  await firebase.firestore().collection(
"scrumMember").add({
    username: user.displayName,
    userId: user.uid,
    email: user.email,
    profile_img: "",
    recent_activity: {},
  });
};
```

Figure 5.79: Code segment of the register by firebase authentication

5.3.3 Forgot Password / Reset Password Module

The module shown in Figure 5.80 is to allow Scrum Member to reset the password for his / her account when he / she forgot it. Scrum Member must enter the recovery email of his / her account to the given input field and press the Reset Password button. It will send an email to the recovery account and provide further information on how to reset the password. Figure 5.81 shows the code segment of reset password by firebase authentication.

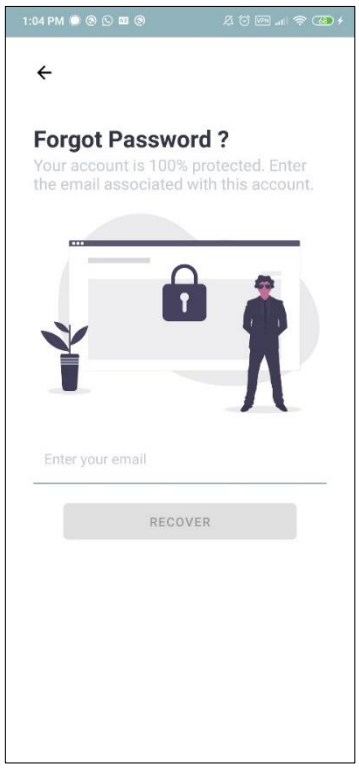


Figure 5.80: Reset password screen.

```
//forgotPassword
export const resetPassword = (resetEmail) ⇒
 {
  return firebase.auth().
sendPasswordResetEmail(resetEmail);
};
```

Figure 5.81: Code segment of the reset password by firebase authentication

5.3.4 View Projects Module

This module allows Scrum Member to view all the projects he / she involves in. Figure 5.81 shows the project dashboard where Scrum Member views all the active projects he / she participates in and the recent activity or also known as sprint backlog item he/ she has updated recently. By pressing the View More or View Older Projects text buttons, Scrum Member will be navigated to a list of the projects that divided into active and close groups based on the project's status, which is shown in Figure 5.83 (left figure). By pressing on one the project card on the project dashboard or the item shown in project list, Scrum Member will be navigated to the screen where it shows a general information of the selected project, which is shown in Figure 5.83 (right figure). Scrum Member also able to view all product backlogs item of the selected project by pressing the View Product Backlogs button. Figure 5.84 shows the list of product backlogs. Figure 5.85 shows the code segment of fetching all active projects data. Figure 5.86 shows the code segment of fetching all projects and fetching single project data.

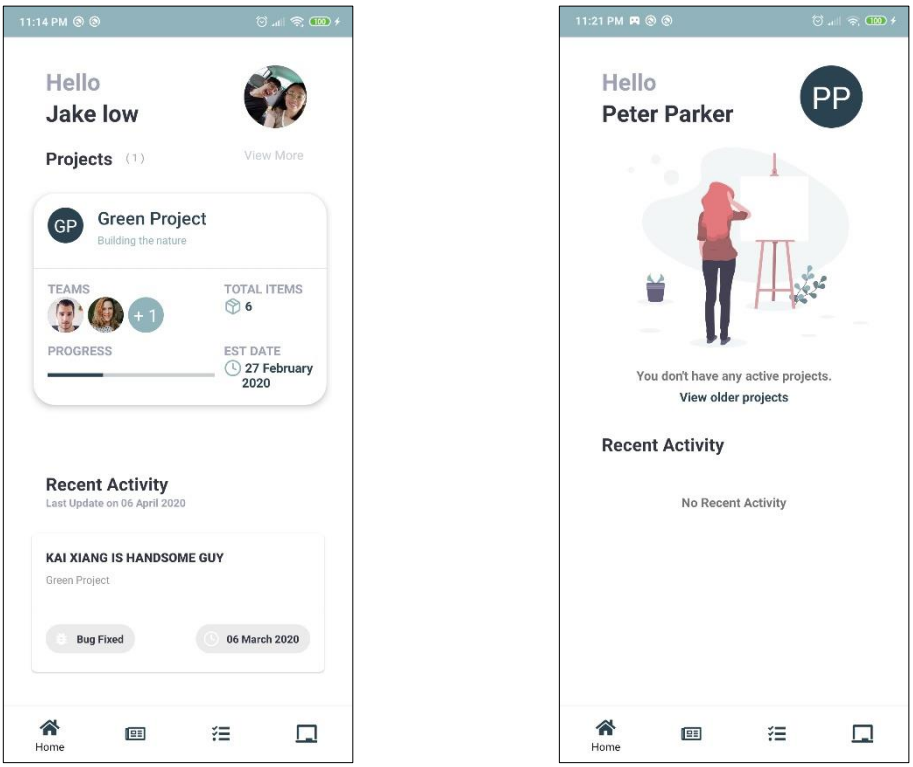


Figure 5.82: Project dashboard screen (left) and empty project dashboard screen (right).

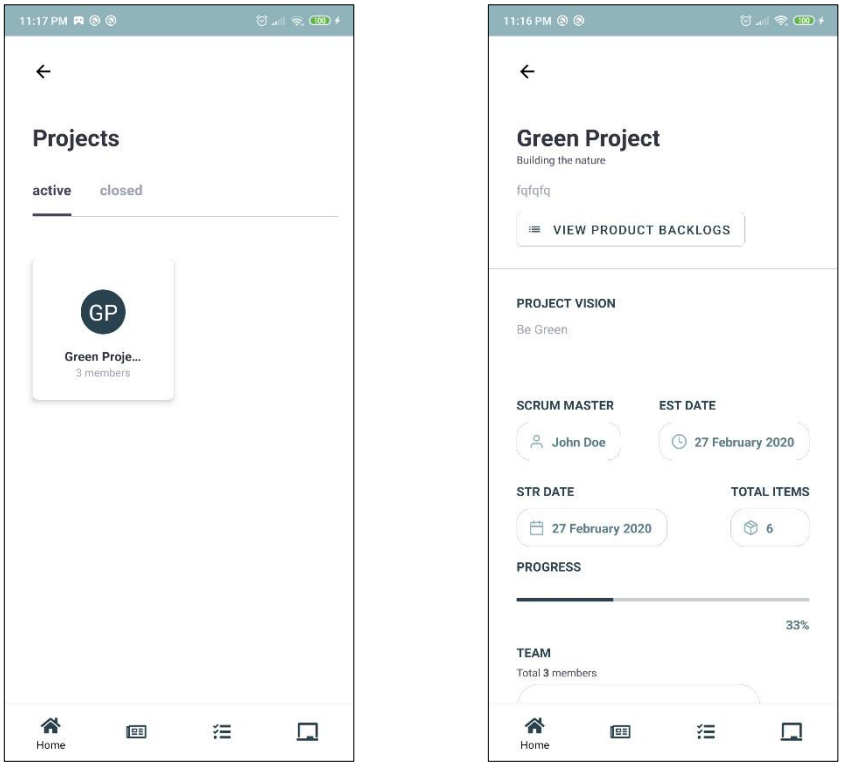


Figure 5.83: Project list screen (left) and Project overview screen (right).

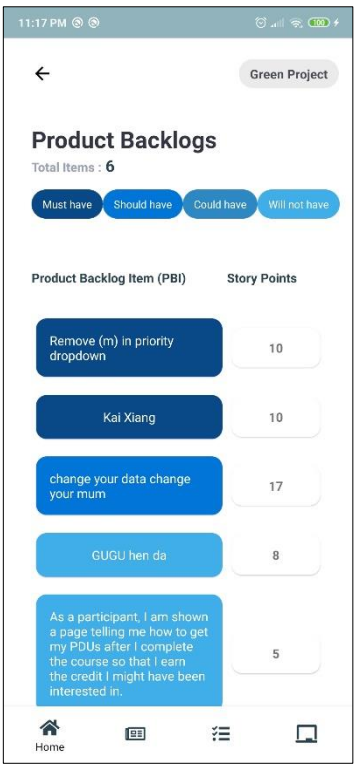


Figure 5.84: View Product Backlog screen.

```
//getActiveProjects
export const getActiveProjects = async () =>
 {
  const userId = firebase.auth().currentUser.
uid;
  let activeProjects = [];

  let querySnapshot = await firebase
    .firestore()
    .collection("scrumProjects")
    .where("participants", "array-contains",
userId)
    .get();

  querySnapshot.forEach((doc) => {
    let projectData = {};
    projectData = doc.data();
    projectData.projId = doc.id;
    if (projectData?.active) activeProjects.
push(projectData);
  });

  let scrumMasterData = [];
  let scrumMemberData = [];
  let sprints = [];
  //fetch all sprints
  sprints = await getSprints();

  //fetch all scrum master data
  scrumMasterData = await getScrumMasters();

  // fetch all scrum member data
  scrumMemberData = await getScrumMembers();

  if (activeProjects && activeProjects?.
length > 0) {
    activeProjects.map((project) => {
      project.scrumMaster = scrumMasterData.
find((master) => {
        return master?.userId === project?.
scrum_master_id;
      });

      if (project?.participants && project?.
participants?.length > 0) {
        project.scrumMembers = [];
        project?.participants.map((
participant) => {
          let participantInfo =
scrumMemberData.find((member) => {
            return member?.userId ===
participant;
          });
          project.scrumMembers.push(
participantInfo);
        });
      }

      if (project?.curr_sprint_id) {
        project.currSprint = sprints?.find((
sprint) => {
          return sprint?.sprintId === project
?.curr_sprint_id;
        });
      }
    });
  }
  return activeProjects;
};
```

Figure 5.85: Code segment of fetch all active projects

```
//getProject
export const getProject = async (projId) => {
  let doc = await firebase
    .firestore()
    .collection("scrumProjects")
    .doc(projId)
    .get();

  let project = doc.data();
  let scrumMasterData = [];
  let scrumMemberData = [];

  //fetch all scrum master data
  scrumMasterData = await getScrumMasters();

  // fetch all scrum member data
  scrumMemberData = await getScrumMembers();

  project.scrumMaster = {};
  project.scrumMaster = scrumMasterData.find
((master) => {
    return master?.userId === project?.
scrum_master_id;
  });

  project.scrumMembers = [];
  if (project?.participants && project?.
participants?.length > 0) {
    project?.participants.map((participant)
=> {
      let participantInfo = scrumMemberData.
find((member) => {
        return member?.userId === participant
;
      });
      project.scrumMembers.push(
participantInfo);
    });
  }
  project.currSprint = {};
  if (project?.curr_sprint_id) {
    project.currSprint = await getSprint(
project?.curr_sprint_id);
  }
  return project;
};

//getAllProjects
export const getAllProjects = async () => {
  const userId = firebase.auth().currentUser.
uid;
  let activeProjects = [];
  let closeProjects = [];
  let allProjects = [];
  let querySnapshot = await firebase
    .firestore()
    .collection("scrumProjects")
    .where("participants", "array-contains",
userId)
    .get();

  querySnapshot.forEach((doc) => {
    let projectData = {};
    projectData = doc.data();
    projectData.projId = doc.id;
    allProjects.push(projectData);
  });

  let scrumMasterData = [];
  let scrumMemberData = [];

  //fetch all scrum master data
  scrumMasterData = await getScrumMasters();

  // fetch all scrum member data
  scrumMemberData = await getScrumMembers();

  if (allProjects && allProjects?.length > 0
) {
    allProjects.map((project) => {
      project.scrumMaster = scrumMasterData.
find((master) => {
        return master?.userId === project?.
scrum_master_id;
      });

      if (project?.participants && project?.
participants?.length > 0) {
        project.scrumMembers = [];
        project?.participants.map((
participant) => {
          let participantInfo =
scrumMemberData.find((member) => {
            return member?.userId ===
participant;
          });
          project.scrumMembers.push(
participantInfo);
        });
      }
    });
  }

  activeProjects = allProjects.filter((
project) => {
    return project?.active === true;
  });

  closeProjects = allProjects.filter((project
) => {
    return project?.active === false;
  });

  return { active: activeProjects, closed:
closeProjects };
};
```

Figure 5.86: Code segment of fetch all projects and fetch single project.

5.3.5 Update assigned sprint tasks Module

This module allows Scrum Member to update the status of his/ her sprints tasks (sprint backlog items / activities). First, Scrum Member selects one of the projects from the project list shown in Figure 5.86 (left). Figure 5.86 (right) shows list of sprint tasks of the current sprint of the selected project. After scrum member select one of the sprint tasks, he/ she will be navigated to an overview of the selected sprint task as shown in Figure 5.88 (left). By pressing the button of the below, a modal will pop out (shown in Figure 5.88 (right)) and shows a group of radio button of status the user intended to update. After selecting one of the status, Scrum Member confirms the action by pressing the ok button. The sprint task would be updated and reflected not only in the Scrum Board Mobile application, also the Scrum Board Web application. Moreover, Scrum Member can view the basic details of the current sprint of the selected project as shown in Figure 5.89. Figure 5.90 shows the code segment of fetching all personal sprint tasks data. Figure 5.91 shows the code segment of updating sprint task's status.

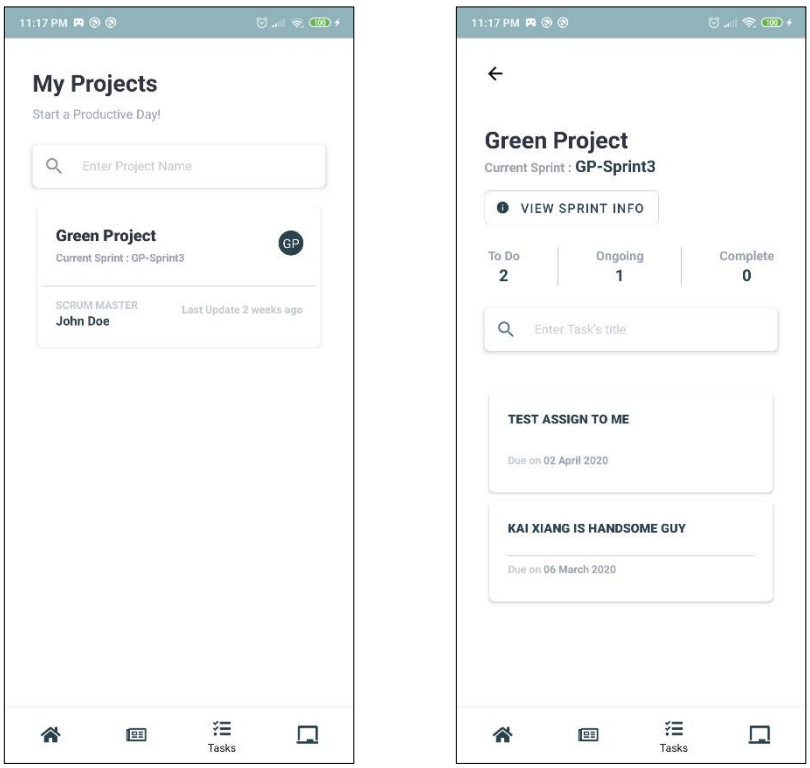


Figure 5.87: Project list screen (left) and sprint tasks screen (right).

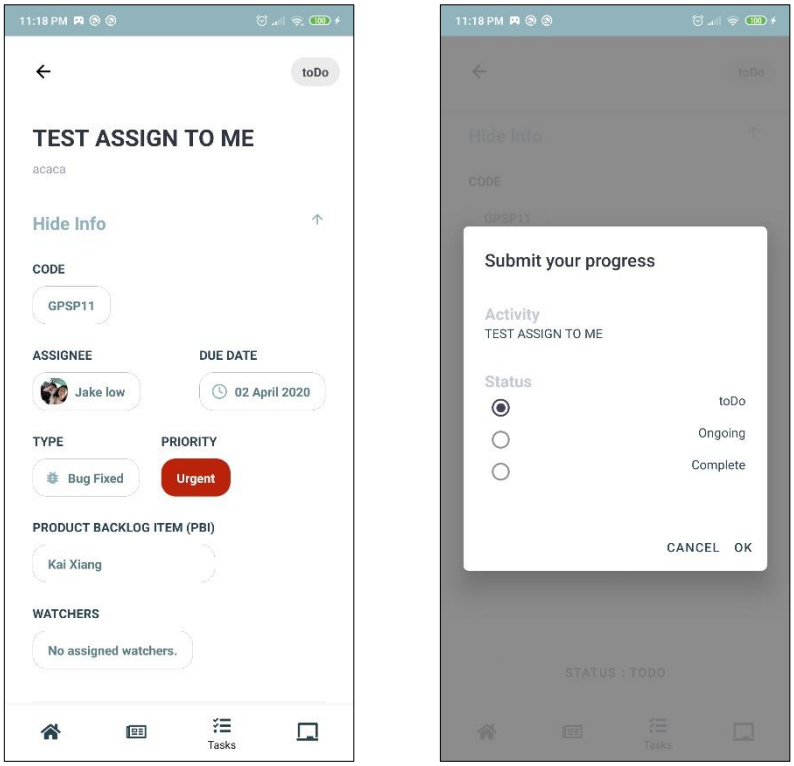


Figure 5.88: Selected sprint task screen and update status modal.

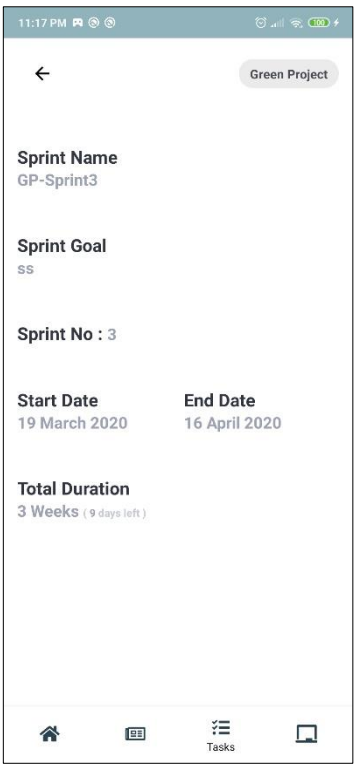
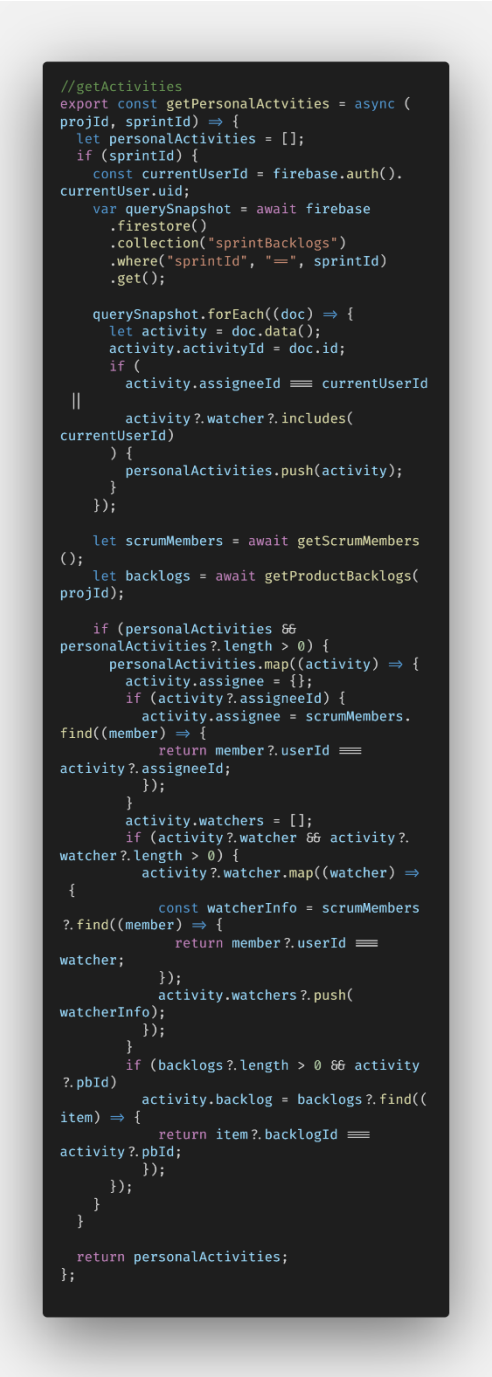


Figure 5.89: Sprint Info screen.



```
//getActivities
export const getPersonalActvities = async (
projId, sprintId) => {
  let personalActivities = [];
  if (sprintId) {
    const currentUserId = firebase.auth().
currentUser.uid;
    var querySnapshot = await firebase
      .firestore()
      .collection("sprintBacklogs")
      .where("sprintId", "==", sprintId)
      .get();

    querySnapshot.forEach((doc) => {
      let activity = doc.data();
      activity.activityId = doc.id;
      if (
        activity.assigneeId === currentUserId
 ||
        activity?.watcher?.includes(
currentUserId)
      ) {
        personalActivities.push(activity);
      }
    });

    let scrumMembers = await getScrumMembers
();
    let backlogs = await getProductBacklogs(
projId);

    if (personalActivities &&
personalActivities?.length > 0) {
      personalActivities.map((activity) => {
        activity.assignee = {};
        if (activity?.assigneeId) {
          activity.assignee = scrumMembers.
find((member) => {
            return member?.userId ===
activity?.assigneeId;
          });
        }
        activity.watchers = [];
        if (activity?.watcher && activity?.
watcher?.length > 0) {
          activity?.watcher.map((watcher) =>
 {
            const watcherInfo = scrumMembers
?.find((member) => {
              return member?.userId ===
watcher;
            });
            activity.watchers?.push(
watcherInfo);
          });
        }
        if (backlogs?.length > 0 && activity
?.pbId)
          activity.backlog = backlogs?.find((
item) => {
            return item?.backlogId ===
activity?.pbId;
          });
      });
    }
  }

  return personalActivities;
};
```

Figure 5.90: Code segment of fetch all personal sprint tasks.

Figure 5.91: Code segment of update sprint task's status.

5.3.6 View Scrum Board of sprint Module

This module allows Scrum Member to view the scrum board of the ongoing sprint of the selected project. After the scrum member select one of the projects from project lists from Figure 5.87 (left), he /she will be able to view the scrum board as shown in Figure 5.92 (left). By pressing on the product backlog item , it will pop out a modal as shown in Figure 5.92 (right) showing the list of available product backlog item which can be selected and filter out a scrum board which its sprint backlogs item matches the corresponding product backlog parent item. Figure 5.93 shows the code segment of fetching the scrum board data.

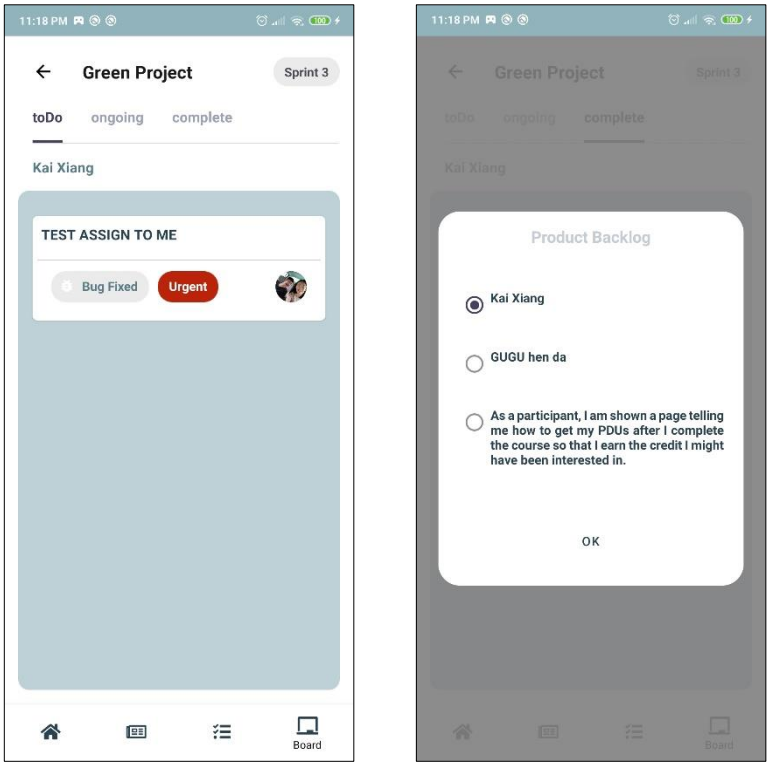


Figure 5.92: Scrum board screen (left) and filter by product backlog item modal (right).

```
export const getScrumBoard = async (projId)
⇒ {
  let doc = await firebase
    .firestore()
    .collection("scrumProjects")
    .doc(projId)
    .get();
  let project = {};
  project = doc.data();

  let backlogs = await getProductBacklogs(
projId);

  let currentSprintId = project?.
curr_sprint_id;

  let scrumMembers = await getScrumMembers();

  if (currentSprintId) {
    let doc = await firebase
      .firestore()
      .collection("sprints")
      .doc(currentSprintId)
      .get();

    let currSprint = {};
    currSprint = doc.data();
    currSprint.sprintId = currentSprintId;

    let querySnapShot = await firebase
      .firestore()
      .collection("sprintBacklogs")
      .where("sprintId", "==",
currentSprintId)
      .get();
    let currSprintActivities = [];

    querySnapShot.forEach((doc) ⇒ {
      let activity = doc.data();
      activity.activityId = doc.id;
      activity.assignee = {};
      if (activity?.assigneeId) {
        activity.assignee = scrumMembers.find
((member) ⇒ {
          return member.userId === activity?.
assigneeId;
        });
      }
      activity.watchers = [];
      if (activity?.watcher && activity?.
watcher?.length > 0) {
        activity.watchers = activity.watcher.
map((watcherId) ⇒ {
          let watching = scrumMembers.find((
member) ⇒ {
            return member?.userId ===
watcherId;
          });
          return watching;
        });
      }
      activity.backlog = backlogs?.find((
backlog) ⇒ {
        return activity?.pbId === backlog?.
backlogId;
      });
      currSprintActivities.push(activity);
    });

    return {
      currSprint: currSprint,
      currActivities: currSprintActivities,
      backlogs: backlogs,
    };
  } else return {};
};
```

Figure 5.93: Code segment of fetching scrum board.

5.3.7 Get unassigned sprint tasks Module

In Figure 5.94, Scrum Member can get unassigned sprint tasks by pressing on the Assign to Yourself button. The sprint task will eventually add into the sprint task list

in Figure 5.87 (right). Figure 5.95 shows the code segment of assigning the unassigned sprint task to yourself.

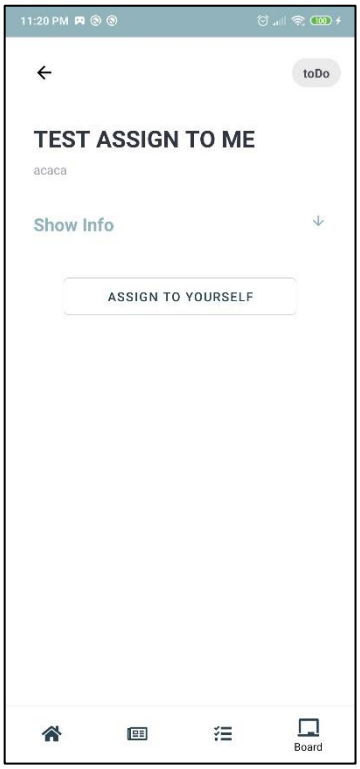


Figure 5.94: Sprint Task screen with assign to yourself button.

```
export const assignTaskToYourself = async (
activity) => {
  try {
    const currentUserId = firebase.auth().
currentUser.uid;
    const currentUsername = firebase.auth().
currentUser.displayName;

    let activityRef = await firebase
      .firestore()
      .collection("sprintBacklogs")
      .doc(activity?.activityId);

    if ((await activityRef.get()).exists) {
      activityRef.update({
        assigneeId: currentUserId,
      });
```

Figure 5.95: Code segment of assign sprint task to yourself.

5.3.8 View Notifications and Feeds Module

This module allows Scrum Member to view all feeds created from all the changes made by the Scrum Master in the Scrum Board Web Application. In Figure 5.96, Scrum Member can either filter the activity feeds by date or filter by projects. Figure 5.97 shows the system notifications on the mobile phone whenever the scrum master makes an update from the scrum board web application. Figure 5.98 shows the code segment to get all feeds data.

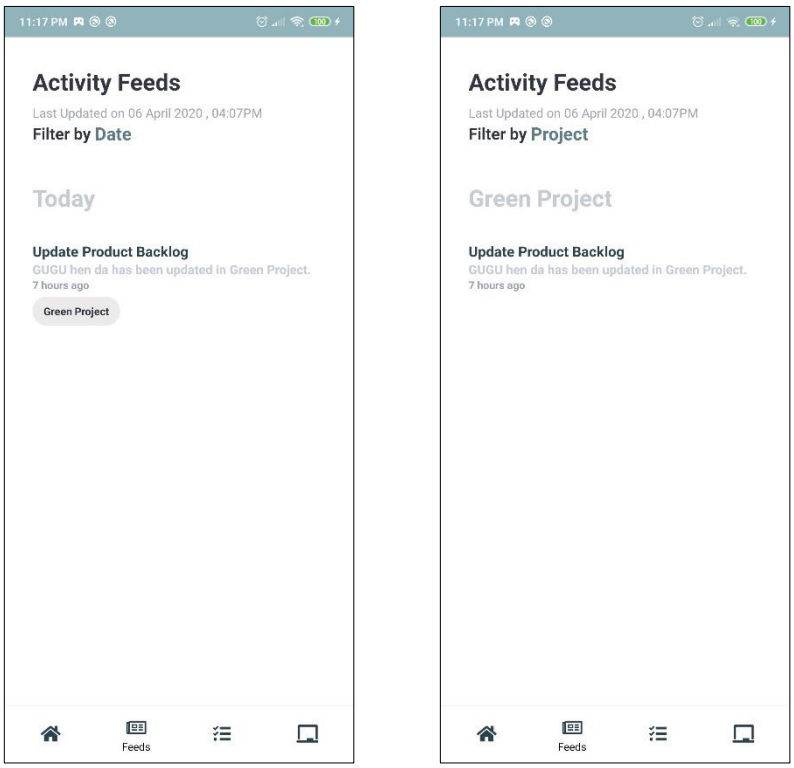


Figure 5.96: Activity Feeds filter by Date or filter by Project screen.

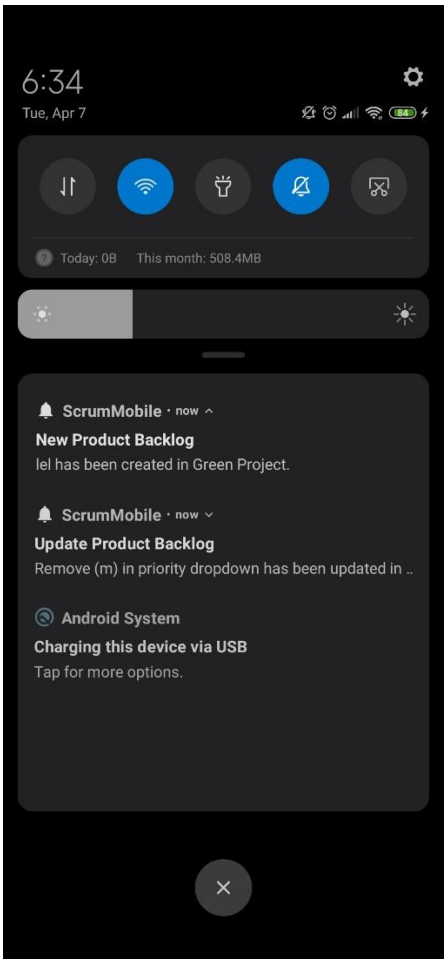
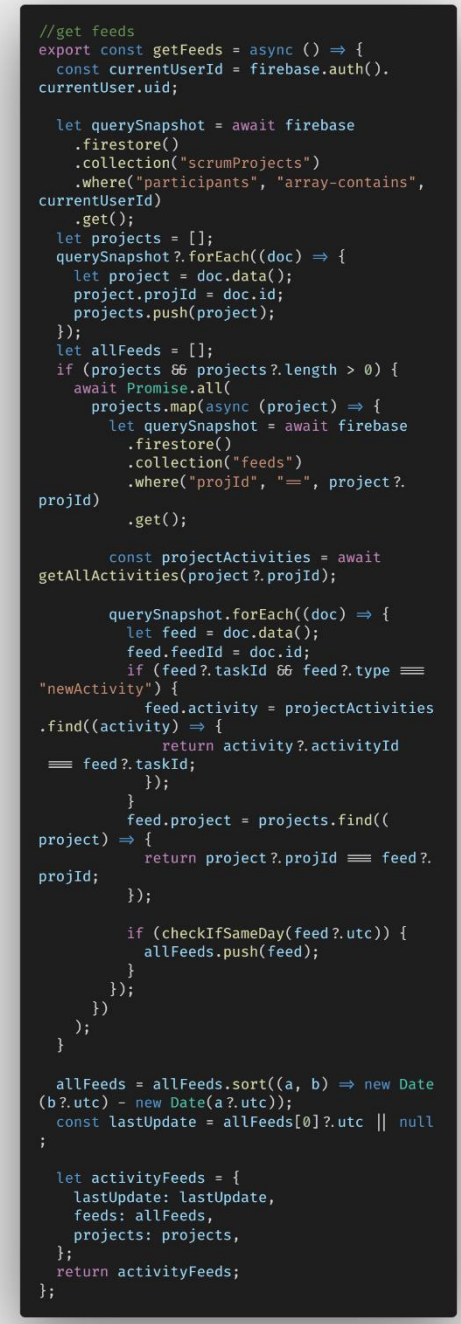


Figure 5.97: Notifications.



```
//get feeds
export const getFeeds = async () => {
  const currentUserId = firebase.auth().
currentUser.uid;

  let querySnapshot = await firebase
    .firestore()
    .collection("scrumProjects")
    .where("participants", "array-contains",
currentUserId)
    .get();
  let projects = [];
  querySnapshot?.forEach((doc) => {
    let project = doc.data();
    project.projId = doc.id;
    projects.push(project);
  });
  let allFeeds = [];
  if (projects && projects?.length > 0) {
    await Promise.all(
      projects.map(async (project) => {
        let querySnapshot = await firebase
          .firestore()
          .collection("feeds")
          .where("projId", "==", project?.
projId)
          .get();

        const projectActivities = await
getAllActivities(project?.projId);

        querySnapshot.forEach((doc) => {
          let feed = doc.data();
          feed.feedId = doc.id;
          if (feed?.taskId && feed?.type ===
"newActivity") {
            feed.activity = projectActivities
.find((activity) => {
              return activity?.activityId
 === feed?.taskId;
            });
          }
          feed.project = projects.find((
project) => {
            return project?.projId === feed?.
projId;
          });

          if (checkIfSameDay(feed?.utc)) {
            allFeeds.push(feed);
          }
        });
      })
    );
  }

  allFeeds = allFeeds.sort((a, b) => new Date
(b?.utc) - new Date(a?.utc));
  const lastUpdate = allFeeds[0]?.utc || null
;

  let activityFeeds = {
    lastUpdate: lastUpdate,
    feeds: allFeeds,
    projects: projects,
  };
  return activityFeeds;
};
```

Figure 5.98: Code segment of get feeds.

5.3.9 Profile Management Module

In the project dashboard, Scrum Member can edit his/ her profile image by pressing on the avatar at the top right of the screen. When the avatar is pressed on, Scrum Member will be navigated to the screen as shown in Figure 5.99 (left). Scrum Member clicks on Camera icon button and select the image he/she wants. Finally, clicks on the Upload Profile Picture button to update his/her profile. Figure 5.100 shows the code segment of uploading new profile image.

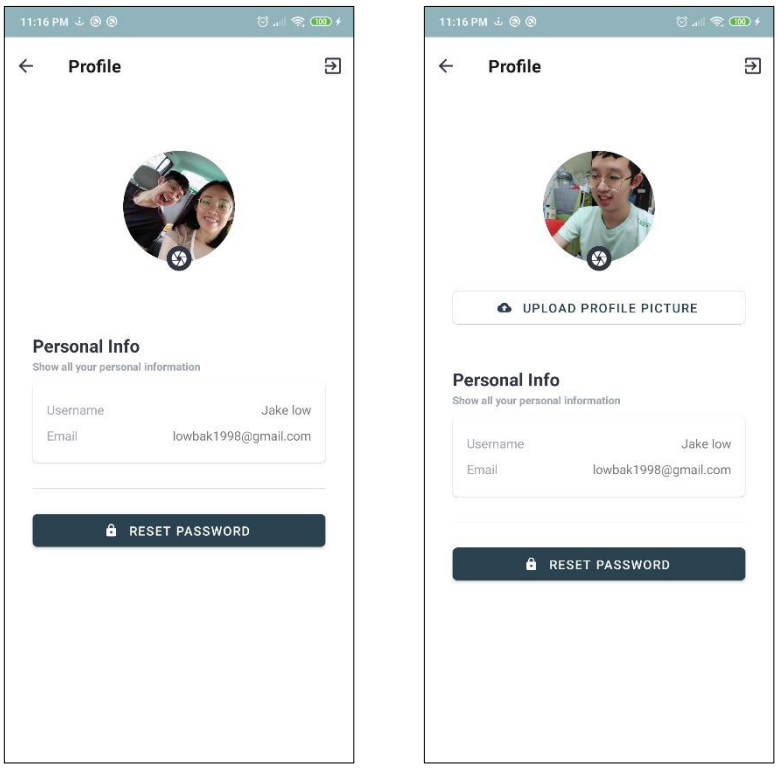


Figure 5.99: Profile screen (left) and updated profile picture profile screen (right).

```
export const uploadProfileImg = async (img,
type) => {
  const currentUserId = firebase.auth().
currentUser?.uid;
  var metadata = {
    contentType: type,
  };
  console.log(img, "img");

  var storageRef = firebase.storage().ref();
  var userProfileRef = storageRef.child(`${
currentUserId}`);

  let uploadTask = await userProfileRef.
putFile(img, metadata);

  await firebase.auth().currentUser.
updateProfile({
    photoURL: uploadTask.downloadURL,
  });

  let querySnapshot = await firebase
    .firestore()
    .collection("scrumMember")
    .where("userId", "==", currentUserId)
    .get();
  let scrumMemberDocRef = null;

  querySnapshot.forEach((doc) => {
    scrumMemberDocRef = doc.id;
  });

  await firebase
    .firestore()
    .collection("scrumMember")
    .doc(scrumMemberDocRef)
    .update({
      profile_img: uploadTask.downloadURL,
    });
};
```

Figure 5.100: Code segment of upload profile image.

5.4 User Authentication and protected route

User Authentication is implemented in the Scrum Board applications using Firebase Authentication. User is only allowed to login to the application with the email and password compliance with what have registered. A token is generated automatically whenever the user login successfully. The token is used to persists user login status as well as the user info.

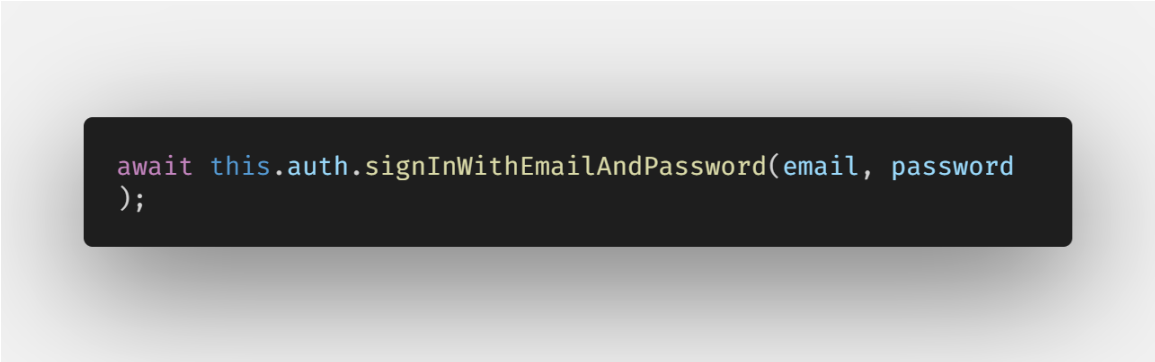


Figure 5.101: Code segment of login with email and password (firebase authentication).

After the login event is triggered as shown in code segment as shown in Figure 5.101, it will actually run the function as shown in code segment as shown in Figure 5.102 which is used to check if the user id registered account is matching with identical user id field of a document inside the scrumMaster collection / scrumMember collection depends on what platform the user is currently using. If it couldn't find the matching document or the document doesn't even exist, the user account will logout automatically. This solution is developed to prevent the user to login in both Scrum Board applications using the same email.

```
// check if scrum member exists
async checkIfScrumMaster() {
  let scrumMasterExist = false;
  await this.db
    .collection("scrumMaster")
    .where("userId", "==", this.auth?.currentUser?.uid)
    .get()
    .then((querySnapshot) => {
      querySnapshot.forEach((doc) => {
        if (doc.exists) {
          scrumMasterExist = true;
        }
      });
    });

  if (scrumMasterExist) {
    return true;
  } else {
    this.signout();
    return false;
  }
}
```

Figure 5.102: Code segment of checkIfScrumMaster

How are protected routes being established in the Scrum Board applications? In the Scrum Board Web application, it takes a long time to fetch the login user data from the Firebase Authentication. The solution is to save the user data in the Window localStorage and ready to be used throughout every web page. At the routing component, the application checks whether is a currentUser data available. If it is not, it will be redirected back to the login component which shows the login page. Otherwise, it will display the according view as expected. Figure 5.103 shows the code segment of the implementation of protected route in Scrum Board web application. Figure 5.104 shows the code segment of the implementation of protected route in Scrum Board mobile application.

Figure 5.103: Code segment of implementation of protected route

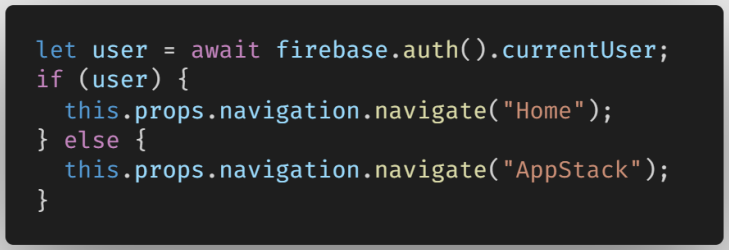


Figure 5.104: Code segment of implementation of protected route

5.5 Form Validation

In Scrum Board web application, there a lot of forms were used. Form validation is important, and it can make certain that the correct data are later saved in the database. All the forms are generated using the Form component of React Bootstrap. The Form component already come with its own form validation method and feedback. In Figure 5.106 shows a sample of the code segment which shows the text area field for the project vision while creating a new project. There is a required term inside the Form control which indicate that this is required input field. When the user submits the form, it will trigger the function in the code segment as shown in Figure 5.107. If one of the input fields was left empty, it will set the variable of validated from false to true. Then, the application will display the specific helper text under the empty input field. The helper text can be configured accordingly in Form Control Feedback inside the targeted Form Group. Figure 5.108 shows the user interface of invalid input field and Figure 5.109 shows the user interface of valid input field.

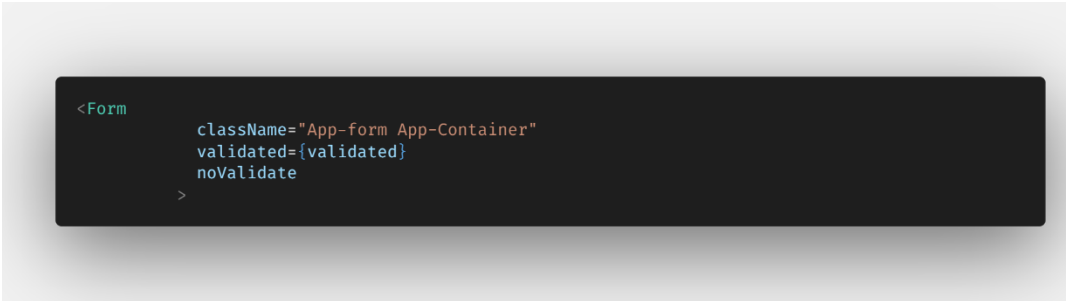


Figure 5.105: Code segment of Form component

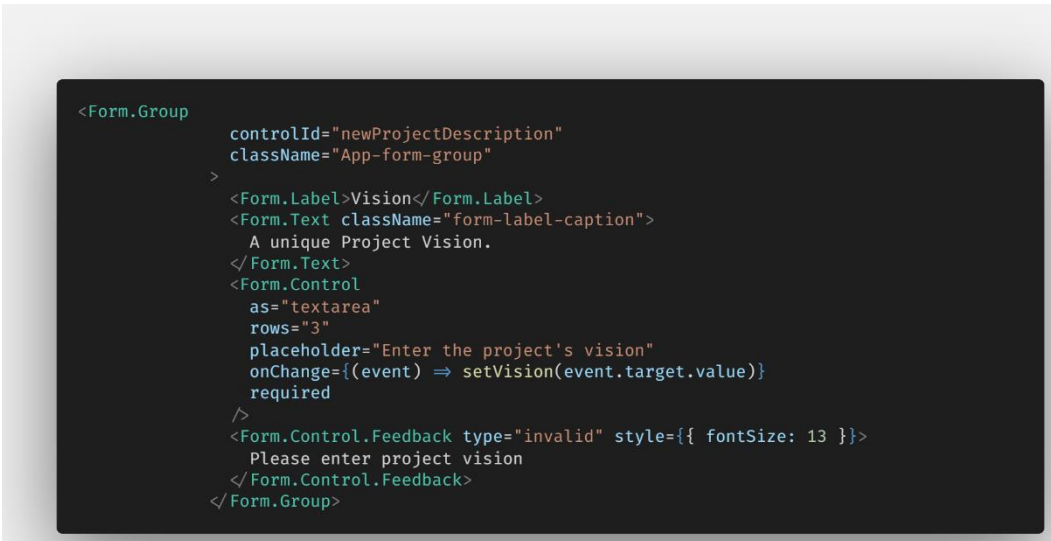


Figure 5.106: Code segment of specific Form Group of the Form component

Figure 5.107: Code segment of Form Validation

Figure 5.108: Invalid form field

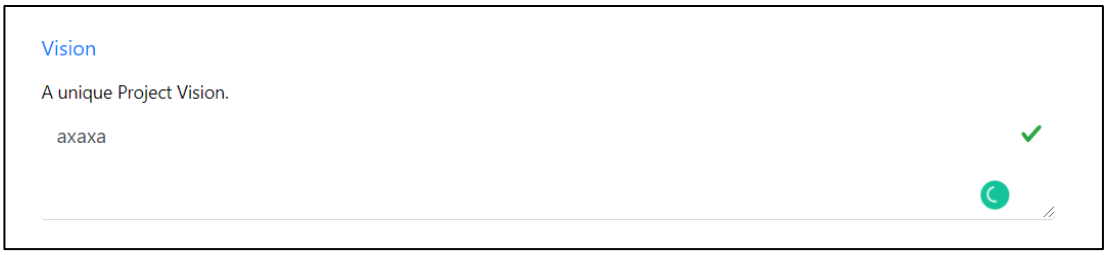


Figure 5.109: Valid form field

5.6 Libraries used in Scrum Board Applications

NPM (Node Package Manager) is an open source package manager and it makes package installation a lot easier. There are a lot of third-party packages which are commonly known as libraries are used in development of the Scrum Board applications. Other than react, react native, Firebase and its toolkits libraries, these are some of the libraries that be used in the development of the Scrum Board Applications.

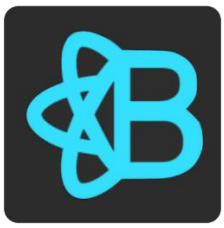


Figure 5.110: React Bootstrap Logo

React Bootstrap is a front-end framework that provides variety of components that are be utilised to develop the user interface of the Scrum Board Web Application. The components it has provided also support Bootstrap layout grid system which make the application more responsive. For further information on the React Bootstrap library, refer to https://react-bootstrap.github.io/getting-started/introduction. However, there

are some limitations when it comes to the available input fields of the form component. React Datepicker provides a Date Picker component that allows users to select the exact date. For further information on the React Datepicker library , refer to https://reactdatepicker.com/.React Multiselect dropdown provides a Multiselect dropdown field component which allows users to select one or many options, it is used for selecting watchers in the sprint backlog items. For further information the React Multiselect dropdown library, refer to https://www.npmjs.com/package/multiselect-react-dropdown. `There are other libraries that are being used in the Scrum Board Web Application to replace the missing components in the React Bootstrap libary:

- React Icons

  A library that includes all popular icons , refer to https://react-icons.github.io/react-icons/ for further information
- React Avatar

  A library that generate an avatar component based on information the user given , it could be a Facebook account , Gmail account etc , refer to https://www.npmjs.com/package/react-avatar for further information
- React Toast Notification

  A library that provides configurable , composable ,toast notification system for React , refer to https://jossmac.github.io/react-toast-notifications/ for further information
- React-sidenav

  A library that provides side navigation menu component that similar to the front end design of Trend Micro antivirus application , refer to https://www.npmjs.com/package/@trendmicro/react-sidenav for further information

Figure 5.111: React Datepicker Logo

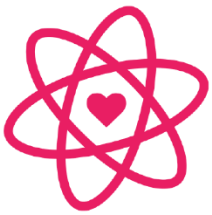


Figure 5.112: React Icons Logo

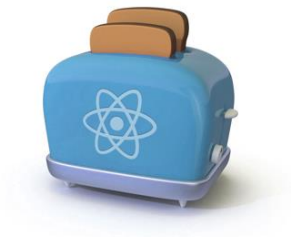


Figure 5.113: React Toast Notification Logo

Figure 5.114: React-pdf Logo

React-pdf is used to create the PDF documents as well as render them on the browser. For further information on React-pdf library, refer to https://react-pdf.org/. React-Data-Export library is used to export the data given into excel documents and saved in it in local storage. For further information on React-Data-Export library, refer to https://www.npmjs.com/package/react-data-export. These libraries are utilised to implement the generate PDF/ Excel Reports feature of the Scrum Board web application.

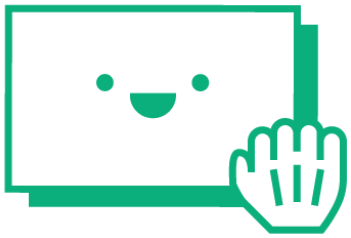


Figure 5.115: React-beautiful-dnd Logo

React-beautiful-dnd is used to develop the Scrum Board in the Scrum Board web application. It supports drag and drop context, beautiful and natural movement of items.

s

Figure 5.116: date-fns Logo

Date-fns , a JavaScript date utility library which is used to manipulate JavaScript dates. For instance, I would like to compare both date to know if the first date is before or ahead of the second date. Rather than writing function by myself, compareAsc and compareDesc functions are utilised. It is used in Scrum Board web application. For further information on the date-fns library , refer to https://date-fns.org/docs/Getting-Started.

Figure 5.117: Lodash Logo

Lodash, a JavaScript Library that is used to makes development using JavaScript much easier by aiding the working with arrays, numbers, objects, strings etc. It is generally used to manipulate and iterate arrays, objects and strings. It is used in Scrum Board applications. For further information on the Lodash library, refer to https://lodash.com/docs/4.17.15 .

Figure 5.118: timeago.js Logo

Timeago.js, a JavaScript Library that is used to format date with "***Time ago". For example, the feed system of Scrum Board applications shows feeds which has attribute showing the how long the feed has been generated in "4 minutes ago" date format. For further information on the timeago.js library, refer to https://timeago.org/.

Figure 5.119: React Native Paper Logo

React Native Paper is a front-end framework that provides variety of components that are be utilised to develop the user interface of the Scrum Board Mobile Application. It is simple and intuitive. For further information on the React Native Paper library , refer to https://callstack.github.io/react-native-paper/index.html. Besides, React-native-vector-icons is being used in Scrum Board Mobile Applications to display icons. For further information on the react-native-vector-icons , refer to https://github.com/oblador/react-native-vector-icons. An animated collapsible view component from react-native-collapsible library is being used in the Scrum Board mobile application, created by the same developer of the react-native-vector-icons library was used Moreover, React-native-image-picker is used to select media for device library or directly from the camera when the user intend to change the profile picture of his/her account in Scrum Board mobile application. For further information on the react-native-image-picker library, refer to https://github.com/react-native-community/react-native-image-picker.

Aside from that, the splash screen of the Scrum Board Mobile application is done using react-native-splash-screen library. The documentation of setup splash screen can be found in https://github.com/crazycodeboy/react-native-splash-screen.

5.7 Notification system using OneSignal

A notification system is implemented inside the Scrum Board applications using OneSignal . First, a new application dubbed “Scrum Board” is created in the OneSignal dashboard. Secondly, setup the web push platform (Scrum Board Web Application) and native app platform (Scrum Board Mobile Application) follow the given documentation. For further information , refer to https://documentation.onesignal.com/docs/web-push-quickstart for web push notification setup and https://documentation.onesignal.com/docs/mobile-sdk-setup for

mobile push notification setup. Figure 5.122 shows the code segment for the initialization for OneSignal for web push notification in index.html file in Scrum Board web application. Figure 5.123 shows the code segment for initialization for OneSignal for mobile push notification in App.js file in Scrum Board mobile application. The installation of react-native-onesignal package is compulsory in the Scrum Board mobile application to complete the setup for OneSignal in React Native application. For further information on the react-native-onesignal library , refer to https://github.com/OneSignal/react-native-onesignal.

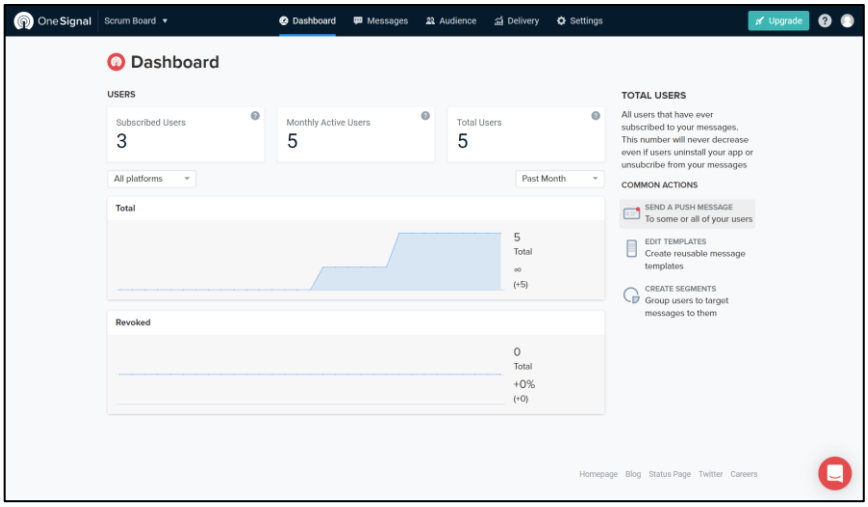


Figure 5.120: OneSignal dashboard screen

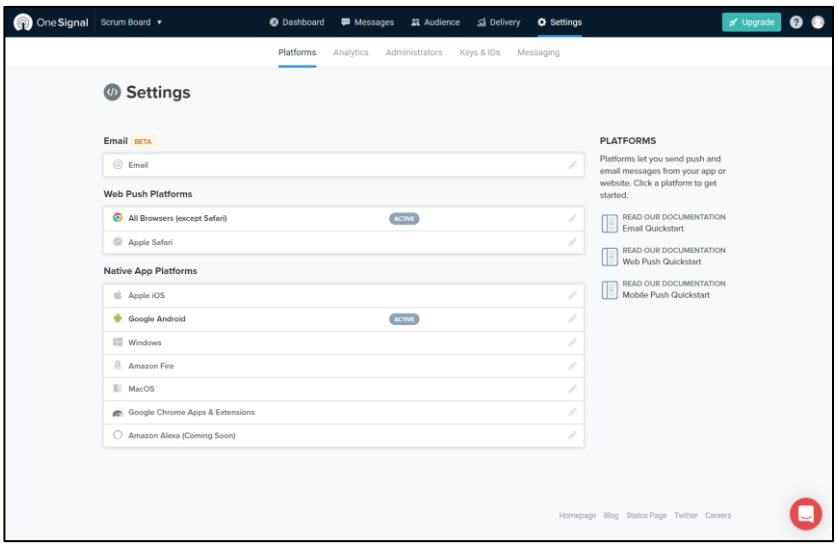


Figure 5.121: OneSignal setting screen

```
<script
  src="https://cdn.onesignal.com/sdks/OneSignalSDK.js"
  async=""
></script>
<script>
  var OneSignal = window.OneSignal || [];
  OneSignal.push(function () {
    OneSignal.init({
      appId: "733ec36b-5a3c-4796-83c4-f6b180462b40",
    });
  });
</script>
```

Figure 5.122: Code segment of initialization of OneSignal for web

```
constructor(properties) {
  super(properties);
  OneSignal.init("733ec36b-5a3c-4796-83c4-f6b180462b40", {
    kOSSettingsKeyAutoPrompt: true,
  }); // set kOSSettingsKeyAutoPrompt to false prompting manually on iOS

  OneSignal.addEventListener("received", this.onReceived);
  OneSignal.addEventListener("opened", this.onOpened);
  OneSignal.addEventListener("ids", this.onIds);
}

componentWillUnmount() {
  OneSignal.removeEventListener("received", this.onReceived);
  OneSignal.removeEventListener("opened", this.onOpened);
  OneSignal.removeEventListener("ids", this.onIds);
}

onReceived(notification) {
  console.log("Notification received: ", notification);
}

onOpened(openResult) {
  console.log("Message: ", openResult.notification.payload.body);
  console.log("Data: ", openResult.notification.payload.additionalData);
  console.log("isActive: ", openResult.notification.isAppInFocus);
  console.log("openResult: ", openResult);
}

onIds(device) {
  console.log("Device info: ", device);
}
```

Figure 5.123: Code segment of initialization of OneSignal for mobile

The setup and initialization of OneSignal are completed. When an user is login successfully to the Scrum Board applications, the user will be subscribed automatically as the audience of the OneSignal application with the data tags that contains the firebaseId , the user id of the authenticated user and the user's role. Figure 5.125 shows the code segment of the send tags and Figure 5.124 shows all the audience that subscribe to the OneSignal application. The data tags will be removed automatically

when the user sign out of the Scrum Board applications , which is shown in the code segment in Figure 5.126.

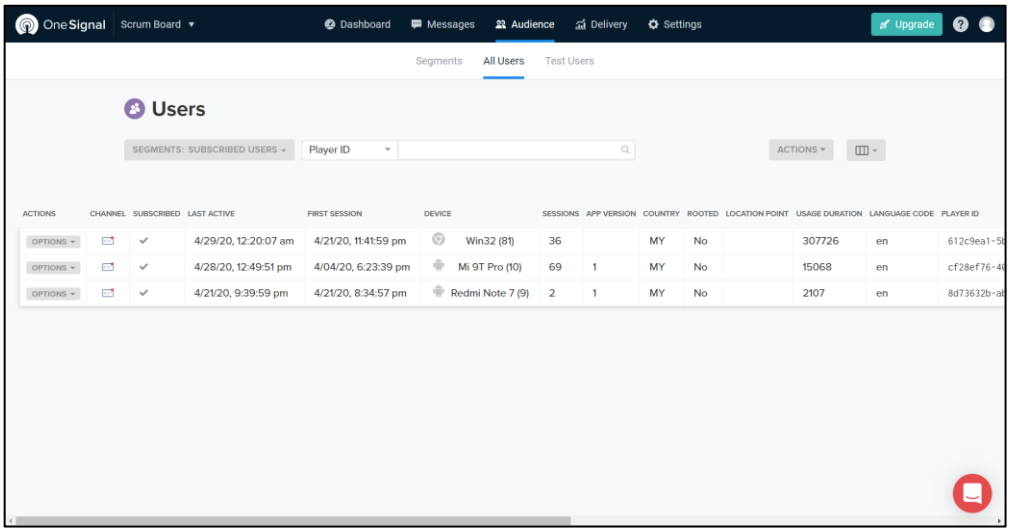


Figure 5.124: OneSignal audience screen

```
await OneSignal.sendTags({
  role: "scrumMember",
  firebaseId: firebase.auth().currentUser.uid,
});
```

Figure 5.125: Code segment of send tags

```
export const signout = () => {
  return firebase
    .auth()
    .signOut()
    .then(async () => {
      await OneSignal.deleteTag("role");
      await OneSignal.deleteTag("firebaseId");
    });
};
```

Figure 5.126: Code segment of delete tags

To send a notification in between both Scrum Board applications, the function is triggered whenever the user makes an action which is shown in the code segment in Figure 5.127. The notification will only be sent to those account that matches the correct data tags information, given in the filters of the body of the API.

```
const sendNotification = async (
  message,
  firebaseId,
  title = "Scrum",
  data = { type: "normal" }
) => {
  try {
    let res = await fetch("https://onesignal.com/api/v1/notifications", {
      method: "POST",
      headers: {
        "Content-Type": "application/json; charset=utf-8",
        Authorization: "Basic MzcxMWE2NGMtZDdiYS00YmFjLWIwNWUtNzgyZTVmNjBkY2Iy",
      },
      body: JSON.stringify({
        app_id: "733ec36b-5a3c-4796-83c4-f6b180462b40",
        contents: { en: message },
        headings: { en: title },
        android_group: title,
        data: data,
        filters: [
          {
            field: "tag",
            key: "firebaseId",
            relation: "=",
            value: firebaseId,
          },
        ],
      }),
    });
    if (!res.ok) {
      throw new Error(res.statusText);
    }
  } catch (err) {
    console.warn(err.message);
  }
};
```

Figure 5.127: Code segment of send notification

5.8 Cloud Firestore

Cloud Firestore is a document-oriented, NoSQL database. It stores data in documents, which are organized into collections. Each document contains a set of key-value pairs. It is quite easy to manage and setup the Cloud Firestore according to the documentation. Compare to SQL database, it has more flexible data model which allows nested objects in addition to subcollections to be created within a document. It is designed to have real time updates as well as one-time fetch queries efficiently. Most importantly, it is schema less and doesn't have any restriction on data structure. In this section, a simple introduction and demonstration will be given to describe the efficiency of writing create, read, update queries using Cloud Firestore in the Scrum Board applications.

Refer to the documentation given in https://firebase.google.com/docs/firestore for further information and better understanding.

5.8.1 Add data

The query is simple. When adding a new document inside a collection, specify the collection name. Attach the add function with the object data as the param and the Cloud Firestore will automatically create the new document inside the collection. Cloud Firestore will automatically generate the document id for newly created document. Then document id can be used for reference for other collection / documents.

```
await this.db
      .collection("scrumProjects")
      .add(newProject)
```

Figure 5.128: Code segment of adding new document

5.8.2 Update / Edit data

When updating one of the documents of the collection, specify the collection name along with the document id. Finally, specify the new key-value pairs you intended to change. Attach the update function at the end and the Cloud Firestore will automatically update the document with value changes on the keys in collection.

```
await this.db.collection("sprints").doc(latestSprintId).update({
  status: "ongoing",
});
```

Figure 5.129: Code segment of editing an existing document

5.8.3 View data

Fetching data is easy but comes with limitation. Cloud Firestore read query has limited usage of where clauses and filter conditions that fetching data with complex relationship harder. Specify the collection name and the condition, the data will be retrieved as QuerySnapShots by attaching get function at the end.. QuerySnapShots is the results of a query and it contains zero or more documents. Iterate it with a ForEach loop method to get item and save it in another array variable to be served for further usage.

Figure 5.130: Code segment of fetch collection of documents

5.8.4 Delete data

When deleting one of the documents of the collection, specify the collection name along with the document id. Attach the delete function at the end and the Cloud Firestore will automatically force delete the document with the specific document id from the collection.

Figure 5.131: Code segment of delete document

CHAPTER 6

SYSTEM TESTING

6.1 Introduction

Software testing is important because it can produce better quality software and improve user experience when bugs and error are significantly reduced until all of them are eradicated. Moreover, earlier detection of bugs helps reduce costs and cause less inconvenience in future. In this project, there were four types of testing were performed. They were user acceptance testing (UAT), unit testing, integration testing and usability testing. 3 Students were participated in the user acceptance testing. Some of them has experience with existing Scrum Board Application such as Jira during their Internship. Students were taken as the main targets during the UAT due to lockdown of MCO. It was a hardship in finding appropriate test users or real-life development team to participate in the UAT where movement between different geographic location is restricted.

6.2 Testing objectives

The objectives of testing are defined as below:

i. Ensure the applications are working as expected
ii. Detect bugs and missing logics
iii. Check if the users can use the applications smoothly according to the user acceptance testing guidelines
iv. Check if the applications is designed in a way that allows users to follow a logical and intuitive path
v. Introduce Scrum Methodology and the usage of Scrum Board to inexperience users

6.3 Testing strategy

6.3.1 Testing scope

In both applications, these are main components that should be tested:

i. Uses interface

- To check if the users can navigate through screens follow their intuitive and previous experience

ii. All the features and functionalities addressed in user requirements

- To check if the applications are working as expected, without any errors or bugs

iii. Usability of the application

- To check if the user satisfies with the features and outlook of the applications

iv. Database testing

- To ensure that all CRUD operations can be performed effortlessly

In contrast, the components that will not be tested are listed below:

i. Performance test of the applications

ii. Firestore rules

6.3.2 Testing type

These were the types of testing would be used in this project. They were unit test, integration test, usability test and user acceptance test.

Unit test was used to test all the applications' modules as well as the basics CRUD operations of firebase firestore and firebase authentication. Integration test was to test of a group correlated unit tests to complete a feature of the applications. User acceptance testing was carried out by actual users to verify the applications are behaving exactly as anticipated such as giving out the correct outputs. Finally, usability test was performed to evaluate how easy the applications can be used. The test took placed with the user acceptance testing.

6.4 Test cases and results

6.4.1 Unit Test

6.4.1.1 Scrum Board Web Application

Table 6.1: Unit test case for login verification

Test Case no	1		**Test Case name**	Login Verification		
Designed by	Low Zi Jian		**Design date**	20 April 2020		
Module	Login Module					
Executed by	Low Zi Jian		**Execution Date**	20 April 2020		
Pre-conditions User requires a valid email and password to login.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Enter valid user email and valid password.	1. Enter email 2. Enter password 3. Click Login Button	- Valid Email - Valid Password	Login Successful	User will be redirected to the project dashboard.	Login Successful	Pass
Enter invalid user email and valid password	1. Enter email 2. Enter password 3. Click Login Button	- Invalid Email - Valid Password	Error message displayed.		Error message displayed.	Pass

Enter valid user email and invalid password.	1. Enter email 2. Enter password 3. Click Login Button	- Valid Email - Invalid Password	Error message displayed.		Error message displayed.	Pass
Enter invalid user email and invalid password.	1. Enter email 2. Enter password 3. Click Login Button	- Invalid Email - Invalid Password	Error message displayed.		Error message displayed.	Pass

This unit test case is reused in scrum board mobile application

Table 6.2: Unit test case for register

Test Case no	2	**Test Case name**	Register Scrum Master			
Designed by	Low Zi Jian	**Design date**	20 April 2020			
Module	Register Module					
Executed by	Low Zi Jian	**Execution Date**	20 April 2020			
Pre-conditions User enters a valid email, username and password to register a new account.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**

Enter valid user email, password, and username.	1. Enter email 2. Enter username 3. Enter password 4. Click Register Button	- Email - password - username	Register Successful	User will be redirected to the project dashboard.	Register Successful	Pass
Missing out some input fields or invalid email address entered	1. Enter email 2. Enter username 3. Enter password 4. Click Register Button		Error message displayed.		Error message displayed.	Pass

This unit test case is reused in scrum board mobile application

Table 6.3: Unit test case for reset password

Test Case no	3	**Test Case name**	Reset Password			
Designed by	Low Zi Jian	**Design date**	20 April 2020			
Module	Reset Password Module					
Executed by	Low Zi Jian	**Execution Date**	20 April 2020			
Pre-conditions User enters a valid email to reset password.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**

Enter valid user email	1. Enter email 2. Click Reset Password Button	- Email	Register Successful		Register Successful	Pass
Missing out some input fields or invalid email address format entered	1. Enter email 2. Click Reset Password Button	- Email	Error message displayed.		Error message displayed.	Pass

This unit test case is reused in scrum board mobile application

Table 6.4: Unit test case for view projects

Test Case no	4		**Test Case name**	View all projects		
Designed by	Low Zi Jian		**Design date**	20 April 2020		
Module	Project Management Module (View Projects)					
Executed by	Low Zi Jian		**Execution Date**	20 April 2020		
Pre-conditions User needs to login to the application.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**

View all active and closed projects.		Existing projects data	Display all projects	User can view all his current project cards in the project dashboard.	Display all projects	Pass

This unit test case is reused in scrum board mobile application

Table 6.5: Unit test case for create project

Test Case no	5	**Test Case name**	Create new project			
Designed by	Low Zi Jian	**Design date**	20 April 2020			
Module	Project Management Module (Create Project)					
Executed by	Low Zi Jian	**Execution Date**	20 April 2020			
Pre-conditions User needs to login to the application. User clicks on the Add Project Card in the project dashboard. User enters all the necessary fields to create a new project.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Fill in all information of a new project	1.Enter all the information in the input fields	- Title - Description - Caption - Vision - Start Date	A new project is successfully created, and a new project card is	User will be redirected to the project dashboard.	A new project is successfully created, and a new project card is	Pass

	given in the form. 3. Click on Create button.	- Completion Date	added to the project dashboard.		added to the project dashboard.	
Missing out some input fields or invalid input format entered	1.Enter all the information in the input fields given in the form. 3. Click on Create button.	- Title - Description - Caption - Vision - Start Date - Completion Date	Error message displayed.		Error message displayed.	Pass
Cancel the create project operation	1. Click on Cancel button.		No new project will be created.	User will be redirected to the project dashboard	No new project will be created.	Pass

Table 6.6: Unit test case for view project

Test Case no	6	**Test Case name**	View project
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Project Management Module (View Project)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020
Pre-conditions User needs to login to the application.			

User selected one the project cards in the project dashboard.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
View all information of the selected project.		Existing selected project data	Display the selected project information.	. User can view an overview of the selected project.	Display the selected project informatio n.	Pass

This unit test case is reused in scrum board mobile application

Table 6.7: Unit test case for edit project

Test Case no	7	**Test Case name**	Edit project
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Project Management Module (Edit Project)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020
Pre-conditions User needs to login to the application. User selected one of the project cards from the project dashboard. User clicks on the edit icon button. User enters all changes to the necessary fields to edit the project.			

Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**

Fill in all information of an existing project	1.Enter all the information in the input fields given in the form. 2. Click on Save Changes button.	- Title - Description - Caption - Vision - Start Date - Completion Date	The project is updated successfully.	User will be redirected to the overview of the selected project.	The project is updated successfully.	Pass
Missing out some input fields or invalid format entered	1.Enter all the information in the input fields given in the form. 2. Click on Save Changes button.	- Title - Description - Caption - Vision - Start Date - Completion Date	Error message displayed.		Error message displayed.	Pass
Cancel the edit project operation	1. Click on Cancel button.		The selected project will not be updated.	User will be redirected to the overview of the selected project.	The selected project will not be updated.	Pass

Table 6.8: Unit test case for delete project

Test Case no	8	**Test Case name**	Delete project
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Project Management Module (Delete Project)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020

Pre-conditions

User needs to login to the application.

User selected one the project cards in the project dashboard.

User clicks on the delete icon button.

Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Delete project	1. Click on the delete icon button.	Existing selected project data	The project is deleted successfully.	User will be redirected to the project dashboard.	The project is deleted successfully.	Pass

Table 6.9: Unit test case for open / close project

Test Case no	9	**Test Case name**	Open / Close project
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Project Management Module (Open Project)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020

Pre-conditions

User needs to login to the application.

User selected one the project cards in the project dashboard.

Test case summary	Test steps	Test data	Expected Result	Post condition / Action	Actual Result	Status (PASS/ FAIL)
The selected project is active.	1. Click on the Close button.	Existing selected project data	The project status is updated successfully. (active to false)	User will be redirected to the project dashboard.	The project status is updated successfully. (active to false)	Pass
The selected project is closed.	1. Click on the Reopen button.	Existing selected project data	The project status is updated successfully. (active to true)	User will be redirected to the project dashboard.	The project status is updated successfully. (active to true)	Pass

Table 6.10: Unit test case for create product backlog

Test Case no	10	**Test Case name**	Create new product backlog
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Product Backlog Management Module (Create Product Backlog)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020
Pre-conditions User needs to login to the application. User selected one the project cards in the project dashboard. User selected the backlog icon of side navigation bar.			

User clicks on the Create new Product Backlog button.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Fill in all information of a new product backlog item	1.Enter all the information in the input fields given in the form. 2. Click on Save Changes button.	- Code - Backlog - Priority - Type - Story Point - Completion Date - Status	A new product backlog item is successfully created, and a new product backlog item is added to the list.	The modal will be closed.	A new product backlog item is successfully created, and a new product backlog item is added to the list..	Pass
Missing out some input fields or invalid input format entered	1.Enter all the information in the input fields given in the form. 3. Click on Create button.	- Code - Backlog - Priority - Type - Story Point - Completion Date - Status	Error message displayed.		Error message displayed.	Pass
Cancel the create product backlog item operation	1. Click on Close button.		No new product backlog item will be created.	The modal will be closed.	No new product backlog item will be created.	Pass

Table 6.11: Unit test case for view product backlog

Test Case no	11	**Test Case name**	View product backlog
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Product Backlog Management Module (View Product Backlog)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020

Pre-conditions

User needs to login to the application.

User selected one the project cards in the project dashboard.

User selected the backlog icon of side navigation bar.

User selected one of the product backlog items from the product backlog list.

Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
View all information of the product backlog item.		Existing selected product backlog data	Display the selected product backlog information.	User can view an overview of the product backlog item.	Display the selected product backlog informatio n.	Pass

Table 6.12: Unit test case for edit product backlog

Test Case no	12	**Test Case name**	Edit product backlog
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Product Backlog Management Module (Edit Product Backlog)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020
Pre-conditions			

User needs to login to the application. User selected one the project cards in the project dashboard. User selected the backlog icon of side navigation bar. User selected one of the product backlog items from the product backlog list. User clicks on Edit button.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Fill in all information of an existing product backlog item	1.Enter all the information in the input fields given in the form. 2. Click on Save Changes button.	- Backlog - Priority - Type - Story Point - Completion Date - Status	The product backlog item is updated successfully.	The modal will be closed.	The product backlog item is updated successfully.	Pass
Missing out some input fields or invalid format entered	1.Enter all the information in the input fields given in the form. 2. Click on Save Changes button.	- Backlog - Priority - Type - Story Point - Completion Date - Status	Error message displayed.		Error message displayed.	Pass

Cancel the edit product backlog item operation	1. Click on Close button.		The selected product backlog item will not be updated.	The modal will be closed.	The selected product backlog item will not be updated	Pass

Table 6.13: Unit test case for delete product backlog

Test Case no	13	**Test Case name**	Delete product backlog			
Designed by	Low Zi Jian	**Design date**	20 April 2020			
Module	Project Management Module (Delete Product Backlog)					
Executed by	Low Zi Jian	**Execution Date**	20 April 2020			
Pre-conditions User needs to login to the application. User selected one the project cards in the project dashboard. User selected the backlog icon of side navigation bar. User selected one of the product backlog items from the product backlog list. User clicks on Delete button.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Delete product backlog item	1. Click on the Delete button.	Existing selected product backlog data	The product backlog item is deleted successfully.	The modal will be closed.	The product backlog item is deleted successfully.	Pass

Test cases are applicable for modules for sprint management and sprint backlog management. The only difference are the test data and preconditions. In conclusion, all the test cases have passed.

For Sprint & Sprint Backlogs,
The preconditions are:
(extend after the user selected one of the project cards in the project dashboard)
User selected the sprint icon of side navigation bar.

If it is view, edit or delete
User selected one of the sprint backlog items from the sprint backlog list. / User selected one of the sprints from the sprint list

If it is creating,
User clicks on the Create new Sprint Backlog button / Create new Sprint button
The test data are:

Table 6.14: Test data for Sprint and Sprint Backlogs

Sprint	Sprint Backlogs
- Sprint length - Start Date - Sprint Goal	- Code (Cannot be edited) - Title - Backlog - Description - Priority - Type - Status - Due Date - Assignee - Watchers - Targeted Sprint

Table 6.15: Unit test case for start / terminate sprint

Test Case no	14	**Test Case name**	Start / Terminate sprint
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Sprint Management Module (Start / Terminate Sprint)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020

Pre-conditions

User needs to login to the application.

User selected one the project cards in the project dashboard.

User selected the sprint icon of side navigation bar.

User selected accordion title of the sprint

Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
The sprint is ongoing.	1. Click on the Close This Sprint button.	Existing selected sprint data	The project current sprint is updated to latest sprint id.		The project current sprint is updated to latest sprint id.	Pass
The sprint is open.	1. Click on the Start this Sprint button.	Existing selected sprint data	The project current sprint is set back to empty.		The project current sprint is set back to empty.	Pass

Table 6.16: Unit test case for delete everything of a sprint

Test Case no	15	**Test Case name**	Delete everything of a sprint
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Sprint Management Module (Delete Everything of a sprint)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020

Pre-conditions

User needs to login to the application.

User selected one the project cards in the project dashboard.

User selected the sprint icon of side navigation bar.

User selected one of the sprints from the sprint list.

User clicks on Delete Everything button.

Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Delete the sprint and its sprint backlogs items	1. Click on the Delete Everything button.	Existing selected sprint data and the sprint backlog item which has the matching targeting sprint id	The Sprint and its related Sprint backlog items are deleted successfully.	The modal will be closed.	The Sprint and its related Sprint backlog items are deleted successfully	Pass

Table 6.17: Unit test case for view archives

Test Case no	16	**Test Case name**	View archives

Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	View Archives Module		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020

Pre-conditions

User needs to login to the application.

User selected one the project cards in the project dashboard.

User selected the archive icon of side navigation bar.

User clicks on one of the archives accordion title from the archive list.

Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
View archive item		Existing selected archive data	Display the selected archive item	User can view all the archives as well as the details of the selected archive (such as sprint info)	Display the selected archive item	Pass
View archive ‘s Scrum Board	1. User clicks on the View Scrum Board button	Existing selected archive data	Display the selected archive item’s scrum board overview.	User can view the archived sprint’s scrum board.	Display the selected archive item’s scrum board overview.	Pass

View archive's sprint backlog items	1. User clicks on one of the sprint backlog items from the sprint backlog list.	Existing selected archive data	Display the selected sprint backlog informatio n	User can view an overview of the sprint backlog item.	Display the selected sprint backlog informatio n	Pass

Table 6.18: Unit test case for view scrum board

Test Case no	17	**Test Case name**	View Scrum Board			
Designed by	Low Zi Jian	**Design date**	20 April 2020			
Module	View Scrum Board Module					
Executed by	Low Zi Jian	**Execution Date**	20 April 2020			
Pre-conditions User needs to login to the application. User selected one the project cards in the project dashboard. User selected the board icon of side navigation bar.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**

Ongoing sprint in the project		Sprint data and related sprint backlog items data.	Display the scrum board of ongoing sprint.	User can view the scrum board of the ongoing sprint.	Display the scrum board of ongoing sprint.	Pass
No ongoing sprint in the project		.	Display empty scrum board.		Display empty scrum board.	Pass

This unit test case is reused in scrum board mobile application

Table 6.19: Unit test case for remove activity from sprint

Test Case no	18	**Test Case name**	Remove activity from sprint			
Designed by	Low Zi Jian	**Design date**	20 April 2020			
Module	View Scrum Board Module (Remove activity from sprint)					
Executed by	Low Zi Jian	**Execution Date**	20 April 2020			
Pre-conditions User needs to login to the application. User selected one the project cards in the project dashboard. User selected the board icon of side navigation bar.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**

Remove the sprint backlog item from the ongoing sprint	1. User select the sprint activity from scrum board swim lane. 2. User clicks on the Remove from this sprint button.	Existing sprint backlog item data	The Sprint backlog item is removed from the sprint and scrum board.	The modal will be closed.	The Sprint backlog item is removed from the sprint and scrum board.	Pass

Table 6.20: Unit test case for Inviting / Removing scrum members

Test Case no	19	**Test Case name**	Invite or remove scrum members			
Designed by	Low Zi Jian	**Design date**	20 April 2020			
Module	View Scrum Board Module (Inviting / Removing scrum members)					
Executed by	Low Zi Jian	**Execution Date**	20 April 2020			
Pre-conditions User needs to login to the application. User selected one the project cards in the project dashboard. User selected the board icon of side navigation bar.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**

Invite new scrum members	1. User clicks on add icon button beside the list of avatar icons 2. User enter email of new scrum member. 3. User clicks on Add button. 4. User clicks on Save Changes Button	List of scrum member	The new scrum member has added into the project.	The modal will be closed.	List of scrum member	Pass
Remove existing scrum members	1. User clicks on add icon button beside the list of avatar icons 2. User clicks on	List of scrum member	The existing scrum member has removed from the project.	The modal will be closed.	The existing scrum member has removed from the project.	Pass

	one of the members from the list. 3. User clicks on Save Changes Button					
Cancel either operations	1. User clicks on add icon button beside the list of avatar icons 2. User clicks on Close button		Scrum member of the project remains unchanged.	The modal will be closed.	Scrum member of the project remains unchanged.	Pass

Generate report module had been tested and the application generate the expected reports in PDF and excel with 100% data accuracy.

Table 6.21: Unit test case for Profile Management

Test Case no	20	**Test Case name**	Upload Profile Image
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Profile Management Module (Upload Profile Image)		

Executed by	Low Zi Jian	**Execution Date**	20 April 2020			
Pre-conditions User needs to login to the application. User clicks on the avatar on the right end of the top navigation bar User clicks on the cog icon button in the popover						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Upload profile image	1. User clicks on Change Profile Picture button 2. User select the picture he / she wants 3. User click on Upload button	New profile picture	User profile image is updated successfully.	The modal will be closed.	User profile image is updated successfully.	Pass
Cancel upload profile image operation	1. User close the modal.		User profile image is remained unchanged.	The modal will be closed.	User profile image is remained unchanged.	Pass

This unit test case is reused in scrum board mobile application

Table 6.22: Unit test case for View notifications and feeds

Test Case no	21		**Test Case name**	View Feeds		
Designed by	Low Zi Jian		**Design date**	20 April 2020		
Module	View notifications and feeds Module (View Feeds)					
Executed by	Low Zi Jian		**Execution Date**	20 April 2020		
Pre-conditions User needs to login to the application.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Get Feed		Existing Feeds data	Display daily activity feeds created from the Scrum Board Mobile Application.		Display daily activity feeds created from the Scrum Board Mobile Applicatio n.	Pass

This unit test case is reused in scrum board mobile application

Table 6.23: Unit test case for View notifications and feeds

Test Case no	22	**Test Case name**	Create Feeds
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	View notifications and feeds Module (Create Feeds)		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020

Pre-conditions User needs to login to the application. User execute either CRUD operations that affect the project itself and its contents.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**
Create Feed	1. User create / edit / delete product backlog item and save changes.		The application will create an activity feed according what action the User has done.		The applicatio n will create an activity feed according what action the User has done.	Pass

This unit test case is reused in scrum board mobile application

6.4.1.2 Scrum Board Mobile Application

Most of the test cases from Scrum Board Web Application are applicable in the Scrum Board Mobile Application. The only difference the screen the user requires to navigate to it in order to access the feature. The preconditions of redundant test cases for Scrum Board Mobile Application is illustrated as below table.

Table 6.24: Modules and its preconditions

Modules	Preconditions
Login, Register, Reset Passwords	Remain unchanged
View Projects	Remain unchanged
View Notification and Feeds	After User login, User presses on Feeds icon in the bottom navigator
Profile Management	After User login,

	User presses on Avatar on top right side of the screen
View Scrum Board	After User login, User presses on Board icon in the bottom navigator User select one of the project cards
View Sprint	After User login, User selected the tasks icon of bottom navigation bar. User selected one the project cards. User pressed on View Sprint Info button
View Product Backlogs	After User login, User selected one of project cards. User pressed on View Product Backlogs button

Table 6.25: Unit test case for update assigned sprint task

Test Case no	23		**Test Case name**	Update assigned sprint task		
Designed by	Low Zi Jian		**Design date**	20 April 2020		
Module	Update assigned sprint task Module					
Executed by	Low Zi Jian		**Execution Date**	20 April 2020		
Pre-conditions User needs to login to the application. User selected the tasks icon of bottom navigation bar. User selected one the project cards. User selected one of the sprint tasks.						
Test case summary	**Test steps**	**Test data**	**Expected Result**	**Post condition / Action**	**Actual Result**	**Status (PASS/ FAIL)**

Update status of the sprint task	1.User presses on the status button. 2. A modal pops out and user selects one of status from the radio button group. 3. User presses on ok button.	-New sprint task status	The sprint backlog item ‘s status is updated successfully.	The modal will be closed.	The sprint backlog item ‘s status is updated successfully.	Pass

Table 6.26: Unit test case for get unassigned sprint task

Test Case no	24	**Test Case name**	Get unassigned sprint task
Designed by	Low Zi Jian	**Design date**	20 April 2020
Module	Get unassigned sprint task Module		
Executed by	Low Zi Jian	**Execution Date**	20 April 2020
Pre-conditions User needs to login to the application. User selected the board icon of bottom navigation bar. User selected one the project cards. User selected one of the unassigned sprint tasks.			

Test case summary	Test steps	Test data	Expected Result	Post condition / Action	Actual Result	Status (PASS/ FAIL)
Get unassigned sprint task	1.User presses on Assigned to Yourself button.	-Current User Id	The sprint backlog item has been assigned to user successfully.		The sprint backlog item has been assigned to user successfully.	Pass

6.4.2 Unit Test Results and Discussion

Since there is no backend setup for project, firestore-jest-mock library is being used to mock the Cloud Firestore and to check if the functions are interacting with the mock firestore as expected or not. Each test case is treated individually and tested in the most fundamental way which to get the correct and accurate data instead of combining all the required data with different function to achieve the correct result to be shown in the user interface.For further information on firestore-jest-mock library ,refer to https://www.npmjs.com/package/firestore-jest-mock.

In conclusion, this is the overview of result of the unit test. Figure 6.1 shows the result of all test cases in unit test.

Table 6.27: Unit Case Test result

Module	Test case no	Status (Pass / Fail)
Login Module	1	**Tested separately,** Pass
Register Module	2	**Tested separately**, Pass
Reset Password	3	**Tested separately**, Pass
Project Management module	4	Pass
	5	Pass
	6	Pass

	7	Pass
	8	Pass
	9	Pass
Product Backlog Management & Sprint Backlog Management & Sprint Management	10	**All have been tested**, Pass
	11	**All have been tested**, Pass
	12	**All have been tested**, Pass
	13	**All have been tested**, Pass
Sprint Management	14	Pass
	15	Pass
View Archives	16	Pass
View Scrum board	17	Pass
Sprint Backlog Management	18	Pass
Project Management	19	Pass
Profile Management	20	**Tested separately**, Pass
View notification and feeds	21	Pass
	22	Pass
Update Sprint Task	23	Pass
Get unassigned Sprint Task	24	Pass

All redundant test cases have been tested and all of them have pass

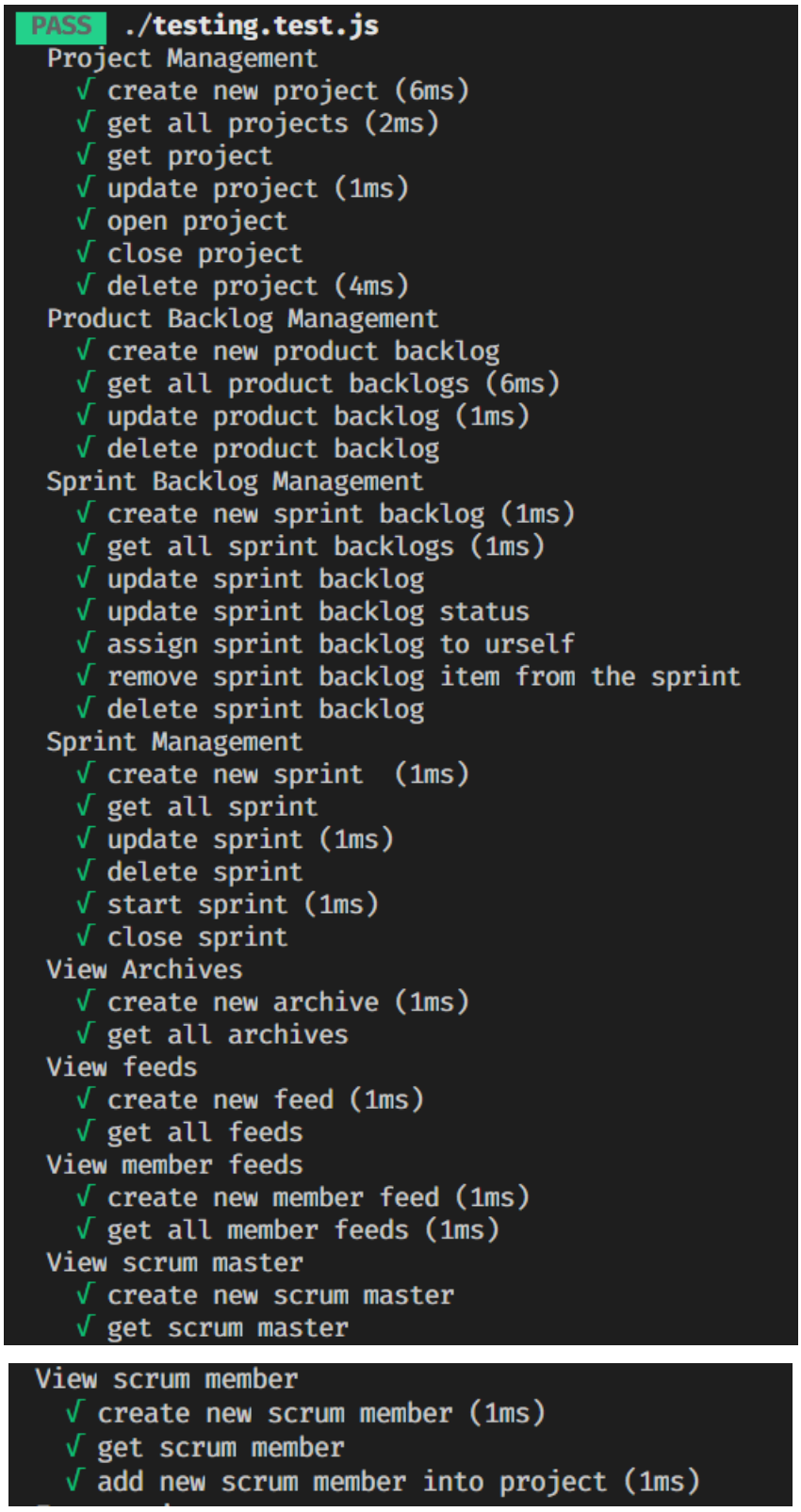


Figure 6.1: Terminal showing the result of the unit testing

How do the unit test be carried out? First, prepare a mock Cloud Firestore database with numerous collections and documents. The same mock database is used for integration testing. Write test script using Jest, a JavaScript Testing Framework and user Expect functions to check if the return data is correct or not. Figure 6.2 shows the code segment of the mock Cloud Firestore database. Figure 6.3 shows the code segment of the test script.

```
mockFirebase({
  database: {
    scrumProjects: [
      {
        id: "1",
        active: true,
        alias: "GP",
        caption: "Be Green",
        curr_sprint_id: "",
        description: "Create a green environment",
        estimate_date: "2020-04-17T08:56:04.649Z",
        is_master_control: false,
        items: 0,
        progress: 0,
        scrum_master_id: "abc123",
        start_date: "2020-04-17T08:56:04.649Z",
        created_at: "2020-04-17T08:56:04.649Z",
        title: "Green Project",
        vision: "Protect enviroment is the responsibility of everyone",
        participants: [],
      },
    ],
    sprints: [
      {
        id: "1",
        sprint_goal: "Complete the feature",
        sprint_length: "14",
        sprint_name: "GP-Sprint1",
        sprnt_num: 1,
        estimate_date: "2020-04-17T08:56:04.649Z",
        create_date: "2020-04-17T08:56:04.649Z",
        start_date: "2020-04-17T08:56:04.649Z",
        projId: "abc123",
        status: "ongoing",
      },
    ],
```

Figure 6.2: Code Segment of mock database

```
test("create new project", async () => {
   const firebase = require("firebase");
   const db = firebase.firestore();

   let newProject = {
     id: "1",
     active: true,
     alias: "GP",
     caption: "Be Green",
     curr_sprint_id: "",
     description: "Create a green environment",
     estimate_date: "2020-04-17T08:56:04.649Z",
     is_master_control: false,
     items: 0,
     progress: 0,
     scrum_master_id: "abc123",
     start_date: "2020-04-17T08:56:04.649Z",
     created_at: "2020-04-17T08:56:04.649Z",
     title: "Green Project",
     vision: "Protect enviroment is the responsibility of everyone",
     participants: [],
   };

   await db
     .collection("scrumProjects")
     .add(newProject)
     .then((doc) => {
       expect(doc.data()).toBeTruthy();
       let savedData = doc.data();
       savedData.id = doc.id;
       expect(savedData).toHaveProperty("id", "1");
       expect(savedData).toStrictEqual(newProject);
     });
 });
```

Figure 6.3: Code Segment of the test script

6.4.3 Integration Test

Table 6.28: Integration test case and results

Modules Involved	**Test Case**	**Test Execution Step**	**Expected Output**	**Status (PASS/ FAIL)**
1. Project Management / Sprint Management / Product Backlog Management / Sprint Backlog Management 2. Create Feed **(Scrum Board Web Application)**	Scrum Master make changes on the project and its related items, the application will automatically create the feed.	1. User clicks on Add Project card. 2. User filled in all information of new project and clicks on Create button.	1. A feed record is created in the database, in the feeds collection. 2. The feed will be shown in Scrum Board Mobile Application, only to those Scrum Members who participate in the project	Pass *This was tested in all management modules*
1. Update the project title 2. Rename every sprint name for consistency	Scrum master update the title of the project, the application will automatically rename the all sprint name.	1. User clicks on one of the projects from the dashboard. 2. User clicks on the edit icon button. 3. User set a new title for the project in the title input field. 4. User clicks on Save Changes button.	1. All the previously created and new sprints will change according to latest project title.	Pass

(Scrum Board Web Application)				
1. Delete sprint 2. Reorder sprint order **(Scrum Board Web Application)**	Scrum master delete a sprint that is not the latest sprint, the application will automatically reorder the sprint number.	1. User clicks on one of the projects from the dashboard. 2. User clicks on the edit icon button. 3. User set a new title for the project in the title input field. 4. User clicks on Save Changes button.	1. All sprint will have the new sequence of sprint number.	Pass
1. Update assigned sprint tasks status 2. Create Feed	Scrum Member update the status of the assigned sprint task, the application will automatically create the feed.	1. User presses on the tasks icon in bottom navigation bar. 2. User presses on one of the projects. 3. User presses on one of the sprint tasks. 4. User presses on the status button.	1. A feed record is created in the database, in the memberFeeds collection. 2. The feed will be shown in Scrum Board Web Application, only to the Scrum Master who created project.	Pass

(Scrum Board Mobile Application)		5. User select one of the status. 6. User presses on the ok button.		
1. get unassigned sprint tasks status 2. Create Feed **(Scrum Board Mobile Application)**	Scrum Member get unassigned sprint task, the application will automatically create the feed	1. User presses on the board icon in bottom navigation bar. 2. User presses on one of the projects. 3. User presses on one of the unassigned sprint tasks. 4. User presses on the Assign to Yourself button.	1. A feed record is created in the database, in the memberFeeds collection. 2. The feed will be shown in Scrum Board Web Application, only to the Scrum Master who created project.	Pass

6.4.4 Integration Test Results and Discussion

Figure 6.4 show the result of the integration test and Figure 6.75 shows the code segment of the integration test script. The integration test is performed by combining multiple unit test script and run it at once. All the test case in integration test have passed.

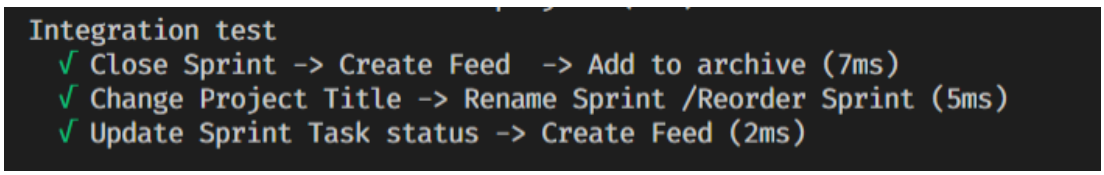


Figure 6.4: Terminal showing the result of the integration test

```
test("create new project", async () => {
   const firebase = require("firebase");
   const db = firebase.firestore();

   let newProject = {
     id: "1",
     active: true,
     alias: "GP",
     caption: "Be Green",
     curr_sprint_id: "",
     description: "Create a green environment",
     estimate_date: "2020-04-17T08:56:04.649Z",
     is_master_control: false,
     items: 0,
     progress: 0,
     scrum_master_id: "abc123",
     start_date: "2020-04-17T08:56:04.649Z",
     created_at: "2020-04-17T08:56:04.649Z",
     title: "Green Project",
     vision: "Protect enviroment is the responsibility of everyone",
     participants: [],
   };

   await db
     .collection("scrumProjects")
     .add(newProject)
     .then((doc) => {
       expect(doc.data()).toBeTruthy();
       let savedData = doc.data();
       savedData.id = doc.id;
       expect(savedData).toHaveProperty("id", "1");
       expect(savedData).toStrictEqual(newProject);
     });
 });
```

Figure 6.5: Code Segment of the test script

6.4.5 User Acceptance Test

Register Module / Reset Password has been skipped.

Table 6.29: Introductory questions

1) **Are you experienced with applications that allow users to schedule and plan their work tasks? Select one of the options below.**
○ Yes ○ No
2) **Have you ever used / heard of Trello? Select one of the options below.**
○ Yes ○ No
3) **What do you expect from a simple board list, cards that helps you to keep track the tasks of the project? Comments on below:**

4) **Have you ever heard of Scrum Methodology / Kanban Methodology? Select one of the options below.**
◦ Yes ◦ No
5) **How long does you take to familiarize and fully understands the features and functionality of an application? Select one of the options below.**
◦ Within a day ◦ One week ◦ A month

6.4.5.1 Scrum Board Web Application

Table 6.30: User Acceptance test for Scrum Board Mobile Application

Testing Date			
Testing Start Time			
Testing End Time			
Name of the tester **(s)**			
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	1. Scrum master login to their account 2. Enter the credentials as below: - Email: scrumfyp2020@gmail.com - Password: 12345678		
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-

Project Management Module	<u>Create Project</u> 1. Scrum Master wants to create a new project dubbed “Green Project” 2. Navigate to the project dashboard 3. Clicks on Add Project button 4. Enter the title as below: “Green Project” 5. Enter the caption as below: “To protect the nature” 6. Enter the description as below: “The project is created for testing” 7. Enter the vision as below: “To encourage people to protect the earth” 8. Enter the Start Date and Completion Date as below: Start date: today Completion date: three months after start date <u>Invite new members</u> 9. Invite a new member with the email: lowbak1998@gmail.com		

	10. Clicks on the create button		
	<u>View Project</u> 1. Scrum Master wants to view the project dubbed “Green Project” 2. Navigate to the project dashboard 3. Selects on the project card dubbed “Green Project”		
	<u>Edit Project</u> 1. Scrum Master edit the project vision of the project dubbed “Green Project” to “Testing changing vision” 2. Navigate to the project dashboard 3. Selects on the project card dubbed “Green Project” 4. Clicks on edit icon button. 5. Enter the project vision as below: “Testing changing vision” 6. Clicks on Save Changes.		
	<u>Close Project</u> 1. Scrum Master close the project “Green Project”		

	2. Navigate to the project dashboard 3. Selects on the project card dubbed “Green Project” 4. Clicks on the Close button		
	<u>Open Project</u> 1. Scrum Master reopen the project “Green Project” 2. Navigate to the project dashboard 3. Selects on the project card dubbed “Green Project” 4. Clicks on the Reopen Project button		
	<u>Delete Project</u> 1. Scrum Master delete the project “Green project” 2. Navigate to the project dashboard 3. Selects on the project card dubbed “Green Project” 4. Click on the delete icon button		
Product Backlog Management Module	<u>Create Product Backlog</u> 1. Scrum Master wants to create a new product backlog item in “Green Project”		

	2. Clicks on backlog from the side navigator. 3. Clicks on Create new Product Backlog button 4. Enter the code as below: "GP1" 5. Enter the backlog as below: "As an earth protector, application should provide us the basic information of recycling" 6. Select priority "Urgent" 7. Select type "User Story / Feature" 8. Enter the story point as below: 21 9. Set completion date to a week later 10. Set status to "Open" 11. Click on Save Changes button		
	<u>View Product Backlog</u> 1. Scrum Master wants to view product backlog item "As an earth protector, application should provide us the basic information of recycling" in "Green Project" 2. Clicks on backlog from the side navigator.		

	3. Select “As an earth protector, application should provide us the basic information of recycling” from the product backlog list		
	<u>Edit Product Backlog</u> 1. Scrum Master wants to edit product backlog item “As an earth protector, application should provide us the basic information of recycling” ‘s status in “Green Project” 2. Clicks on backlog from the side navigator. 3. Select “As an earth protector, application should provide us the basic information of recycling” from the product backlog list 4. Clicks on Edit button 5. Set status from “Open” to “Ongoing” 6. Clicks on Save Changes button.		

	Delete Product Backlog 1. Scrum Master wants to delete product backlog item “As an earth protector, application should provide us the basic information of recycling” ‘s status in “Green Project” 2. Clicks on backlog from the side navigator. 3. Select “As an earth protector, application should provide us the basic information of recycling” from the product backlog list 4. Clicks on Delete button		
Sprint Backlog Management Module	Create Sprint Backlog 1. Scrum Master wants to create a new sprint backlog item in “Green Project” 2. Clicks on Sprint from the side navigator. 3. Clicks on Create new Sprint Backlog button 4. Enter the code as below: “GPA1” 5. Enter the title as below: “Create Recycling Information Feature” 6. Select the backlog as below:		

	"As an earth protector, application should provide us the basic information of recycling" 7. Enter the description as below: "Create a feature where user can access to wide range of facts and info about recycling" 8. Select priority "Urgent" 9. Select type "Feature" 10. Set status to "toDo" 11. Set completion date to a week later 12. Choose an assignee dubbed "Jake Low" 13. Click on Save Changes button		
	<u>View Sprint Backlog</u> 1. Scrum Master wants to view sprint backlog item "Create Recycling Information Feature" in "Green Project" 2. Clicks on Sprint from the side navigator. 3. Select "Create Recycling Information Feature" from the sprint backlog list		
	<u>Edit Sprint Backlog</u>		

	1. Scrum Master wants to edit product backlog item "Create Recycling Information Feature" 's status in "Green Project" 2. Clicks on Sprint from the side navigator. 3. Select "Create Recycling Information Feature" from the sprint backlog list 4. Clicks on Edit button 5. Set status from "toDo" to "Ongoing" 6. Clicks on Save Changes button.		
	<u>Delete Sprint Backlog</u> 1. Scrum Master wants to delete sprint backlog item "Create Recycling Information Feature" in "Green Project" 2. Clicks on Sprint from the side navigator. 3. Select "Create Recycling Information Feature" from the sprint backlog list 4. Clicks on Delete button		
Sprint Management Module	<u>Create Sprint</u> 1. Scrum Master wants to create a new sprint in "Green Project"		

	2. Clicks on Sprint from the side navigator. 3. Clicks on Create new Sprint button 4. Select Sprint length 4 weeks 5. Set Start day to today 6. Enter Sprint goal as below: “Complete recycling info feature” 7. Click on Save Changes button		
	<u>View Sprint</u> 1. Scrum Master wants to view sprint “GP-Sprint1” in “Green Project” 2. Clicks on Sprint from the side navigator. 3. Select “GP-Sprint1” from the sprint list		
	<u>Edit Sprint</u> 1. Scrum Master wants to edit sprint “GP-Sprint1” ‘s sprint length in “Green Project” 2. Clicks on Sprint from the side navigator. 3. Select “GP-Sprint1” from the sprint list 4. Clicks on Edit button 5. Set sprint length to 2 Weeks		

	6. Clicks on Save Changes button.		
	<u>Delete Sprint</u> 1. Scrum Master wants to delete sprint “GP-Sprint1” ‘s sprint length in “Green Project” 2. Clicks on Sprint from the side navigator. 3. Select “GP-Sprint1” from the sprint list 4. Clicks on Delete button		
	<u>Delete Everything</u> 1. Scrum Master wants to delete sprint “GP-Sprint1” ‘s sprint length in “Green Project” 2. Clicks on Sprint from the side navigator. 3. Select “GP-Sprint1” from the sprint list 4. Clicks on Delete button		
	<u>Add activity in Sprint</u> 1. Scrum Master wants to add in sprint backlog item in sprint “GP-Sprint1” in “Green Project” 2. Clicks on Sprint from the side navigator. 3. Expands “GP-Sprint1” from the sprint list 4. Clicks on Add Sprint Backlog now button		

	5. Repeat Create Sprint Backlog steps here		
	<u>Start a Sprint</u> 1. Scrum Master wants to start sprint “GP-Sprint1” in “Green Project” 2. Clicks on Sprint from the side navigator. 3. Expands “GP-Sprint1” from the sprint list 4. Clicks on Start this sprint button.		
View Scrum Board Module	<u>View Scrum Board</u> 1. Scrum Master wants to view the scrum board of “GP-Sprint1” in “Green Project” 2. Clicks on Board from the side navigator.		
	<u>View Scrum Board (Filter by product backlog)</u> 1. Scrum Master wants to view the scrum board of “GP-Sprint1” in “Green Project” 2. Clicks on Board from the side navigator. 3. Filter by user story “As an earth protector, application should provide us the basic information of recycling”		

	View Scrum Board (Group by status / story) 1. Scrum Master wants to view the scrum board of “GP-Sprint1” in “Green Project” group by status / group by story 2. Clicks on Board from the side navigator. 3. Clicks on drop down list for Group by from Group by status to Group by Story		
	View Scrum Board (board view / list view) 1. Scrum Master wants to view the scrum board of “GP-Sprint1” in “Green Project” in list view 2. Clicks on Board from the side navigator. 3. Clicks on list view		
	Invite new member 1. Scrum Master wants to add new member in “Green Project” 2. Clicks on Board from the side navigator. 3. Clicks on the Add button 4. Add a new member with the email as below: john@email.com		

	Terminate Sprint 1. Scrum Master wants to terminate “GP-Sprint1” in “Green Project” 2. Clicks on Board from the side navigator. 3.Clicks on Terminate this sprint button		
View Archive Module	View Archive 1. Scrum Master wants to view all the terminated sprint “GP-Sprint 1 and its activities. 2. Clicks on Archive from the side navigator. 3. Select the “GP-Sprint 1” title		
	View Scrum Board 1. Scrum Master wants to view the terminated sprint “GP-Sprint 1” ‘s scrum board 2. Clicks on Archive from the side navigator. 3. Select the “GP-Sprint 1” title 4.Clicks on the View Scrum Board button		
Generate Reports Module	Generate PDF report 1. Scrum Master wants to generate a pdf report of		

	product backlog items in "Green Project" 2. Clicks on Backlog from the side navigator. 3. Click on the print icon 4. Select PDF 5. Clicks on Download button		
	<u>Generate Excel report</u> 1. Scrum Master wants to generate a excel report of product backlog items in "Green Project" 2. Clicks on Backlog from the side navigator. 3. Click on the print icon 4. Select Excel 5. Clicks on Download button		
Profile Management Module	<u>Update Profile Picture</u> 1. Scrum master wants to change his account profile picture. 2. Navigate to the project dashboard. 3. Clicks on the Avatar on top navigator 4. Selects the cog icon in the popover 5. Clicks on the Change Profile Picture button 6. Select a random image		

	7. Clicks on the Upload button		
View Notifications and feeds Module	View activity feeds 1. Scrum master wants to view the activity feeds of "Green Project". 2. Navigate to the project dashboard.		

6.4.5.2 Scrum Board Mobile Application

Table 6.31: User Acceptance test for Scrum Board Mobile Application

Testing Date			
Testing Start Time			
Testing End Time			
Name of the tester **(s)**			
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	1. Scrum member login to their account 2. Enter the credentials as below: - Email: lowbak1998@gmail.com - Password: 12345678		
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-

View Projects	<u>View Project Detail</u> 1. Scrum member wants to view the project “Green Project” and its information 2. Presses on the home icon in bottom navigator. 3. Presses on the project card which titled “Green Project”		
	<u>View Project Backlogs of the project</u> 1. Scrum member wants to view the project “Green Project” ‘s product backlogs items 2. Presses on the home icon in bottom navigator. 3. Presses on the project card which titled “Green Project” 4. Presses on the View Product Backlogs button		

	View Project current Sprint Info 1. Scrum member wants to view the project “Green Project” ‘s product backlogs items 2. Presses on the tasks icon in bottom navigator. 3. Select “Green Project” 4. Presses on the View Sprint Info button		
	View all projects 1. Scrum member wants to view all the projects that he is currently participating, both active and close projects 2. Presses on the home icon in bottom navigator. 3. Presses on View More text button		
Update assigned sprint tasks Module	Update Sprint task’s status 1. Scrum member wants to update the status of the sprint task named “Testing Update Task” in “Green Project”. 2. Presses on the task icon in bottom navigator. 3. Select “Green Project”		

	4. Select “Testing Update Task” from the “toDo” list 5. Presses on the status button. 6. Select “ongoing” 7. Presses Ok		
View Scrum Board of current sprint tasks Module	<u>View Scrum Board of current sprint</u> 1. Scrum member wants to view the scrum board of “Green Project”. 2. Presses on the board icon in bottom navigator. 3. Select “Green Project”		
Get unassigned sprints tasks Module	<u>Get unassigned sprint task</u> 1. Scrum member wants to assign himself to a sprint task named “Testing Assign Task to yourself” in “Green Project”. 2. Presses on the board icon in bottom navigator. 3. Select “Green Project” 4. Select “Testing Assign Task to yourself” from the “ongoing” list 5. Presses on the Assign to Yourself button		
View Notifications and Feeds Module	<u>View activity feeds</u> 1. Scrum member wants to view the activity feeds of “Green Project”.		

	2. Presses on the feeds icon in bottom navigator. 3. Change filter to Projects		
Profile Management Module	<u>Update Profile Picture</u> 1. Scrum member wants to change his account profile picture. 2. Presses on the home icon in bottom navigator. 3. Presses on the Avatar 4. Presses on the camera icon 5. Select a random image 6. Presses on the upload button		

6.4.6 User Acceptance Test Results and Discussion

Due to MCO (Movement Control Order), the UAT is conducted using online chat platform dubbed “Discord”. The complete UAT form is prepared and the tester is required to fill in during the testing. The result of the UAT are shown in the appendix. There are total 3 testers participate in the UAT. They are Lee Hoe Mun, Chin Kai Xiang and Chong Zi Ming. Lee Hoe Mun is the only tester who had previous experience in Scrum Board during his internship. He often shares his ideas during the development of the Scrum Board applications.

Figure 6.6: Discord Logo

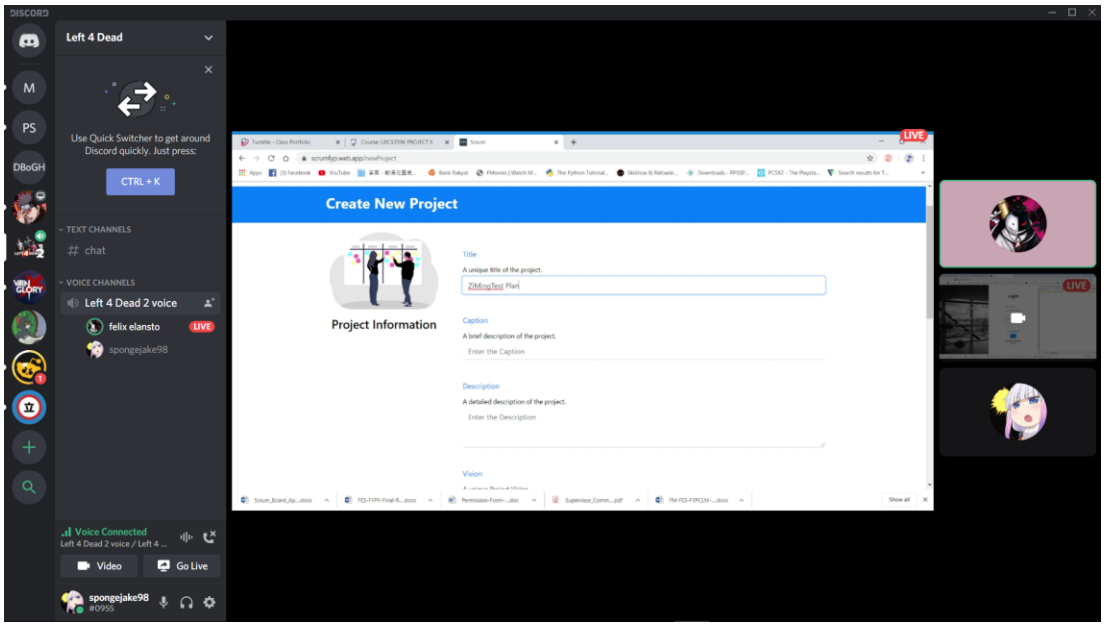


Figure 6.7: UAT using Discord

The UAT is carried out step by step, module by module. Whenever the user starts the UI testing of the new module, an introduction and explanation about the purpose of the feature and its contribution towards Scrum are provided. Overall, the testers can complete all the modules effortlessly or with a little help. They are satisfied with design of user interface and wish to see more features in the future.

6.4.7 Usability Test

6.4.7.1 Usability test survey form template

Table 6.32: User satisfaction survey form

	Strongly Disagree 1	**Disagree 2**	**Neutral 3**	**Agree 4**	**Strongly Agree 5**
1. I think that I would use the applications for planning and scheduling my projects.					
2. I found the applications are unnecessarily complex.					
3. I found the applications are relatively easy to use.					
4. I found the applications have all the mandatory feature.					
5. I found that I need the help of technical person to be able to use the applications.					
6. I found the applications can be easily navigate without a lot of backtracking and data re-entry.					
7. I believe that other users would learn to user the applications effortlessly.					

8. I found the applications have very awkward or inappropriate content.					
9. I felt very confident using the applications.					
10. I need to learn of things before I could get going with the applications.					

1. What did you like best about the scrum board applications (web & mobile)?

_____

_____

_____

2. What did you like least about the scrum board applications (web & mobile)?

_____

_____

_____

3. If you were to describe the scrum board applications (web & mobile) to a colleague in a sentence of two, what would you say?

_____

_____

_____

4. Do you have any final comments or questions?

_____

_____

6.4.8 Usability Test Results and Discussion

Table 6.33: User satisfaction survey result

Question	**Evaluator**			**Average**
	1	**2**	**3**	
1. Usability	4	5	3	4/5
2. Complexity	2	3	4	3/5
3. Ease of User	4	4	2	3.3/5
4. Usability	4	5	2	3.6/5
5. Learnability	3	3	4	3.3/5
6. Navigability	4	4	4	4/5
7.Learnabilty	3	4	2	3/5
8. Satisfaction (Dissatisfaction)	1	1	2	1.3/5
9. Satisfaction	5	4	3	4/5
10. Learnability	2	4	3	3/5
Total	32	37	29	32.7 / 40

In conclusion, the average for the usability testing is 32.7 over 40. It can be considered as a good result.

6.5 Conclusion

All the testing objectives have been met. Although some of redundant integration test cases or unit test case might not be tested with Jest, they are proved to work correctly with the logic given. User acceptance testing and usability testing are carried out concurrently to find out whether new Scrum users can adapt and understand the given system easily. The average time spent to complete a module in the UAT is 4 to 5 minutes. The overall satisfaction rate is slightly above average. Total users participate in the UAT and UT were 3. They were Chong Zi Ming, Chin Kai Xiang and Lee Hoe Mun. Zi Ming and Kai Xiang are new to Scrum and take longer time to understand the features of each modules given to be tested compare to Hoe Mun, an amateur who have previously used Jira during his internship. In conclusion, the number of testers is insufficient. There should be at least 5 users to 20 users participate in the acceptance testing for better and accurate quantitative studies. Moreover, the UAT and UT should be carried out amongst development team members, not individual software developer or student. The behaviour and response given by the testers should be keep tracked as well as the time used to complete each module. The lesson learnt should be documented more rigidly and used as a helpful source for further feedbacks and insights. Hence, more efforts are needed to put in UAT and UT coming in finding the sufficient and right audience for better commitment in achieving better UAT and UT results. UAT should not be taken place at the end the project. The workflow of UAT should be revised.

CHAPTER 7

CONCLUSION

7.1 Conclusion

The Scrum Board web and mobile application increase progress visibility. Every team member can constantly visualize project progress with a scrum board. It is easy to see what tasks are done and what still needs to be finished. A digital scrum board also promotes interaction between team members. Team members will be able to view the board and discuss how different tasks are progressing. The discussion can improve interaction and collaboration. With a scrum board, the team commit to the overall project when team members keep track the progress together as a team instead of individually. Hence, it encourages team collaboration and ownership. By adopting the scrum board digitally, it has successfully overcome the limitation of a physical scrum board. Scrum master no longer need to worry about having difficulty with monitoring and keep track of the project. Scrum Board web application allows scrum master to manage multiple projects concurrently along with all product backlogs, sprint backlogs, sprints involve in the project easily. The scrum board in Scrum Board web application doesn't have physical space constraints and it can contain a lot of tasks. It also provides filtering by user stories / product backlogs and group by status / story options for Scrum Master to easily view through the contents effortlessly. Moreover, Scrum master could also use the Scrum Board web application to generate reports and these reports could be saved digitally. If they were deleted accidentally in local storage, they could redownload the reports in the archive. Most importantly, the Scrum Board mobile application could be accessed anywhere and in whenever time which eventually improve collaboration between members who are working in different location or time zone. Scrum Board mobile application also allows Scrum members to keep track the sprint task and progress of the project by providing a visual representation of the scrum board. To contribute to the project, Scrum member can assign himself / herself to an unassigned sprint task of the project.

Moreover, the applications are developed and designed at least 70% accurately to the scrum methodology when it comes to sprint, product backlogs and sprint backlogs management. Sprint has a limited duration and the scrum master is urged to

close the sprint when it is expired. In every sprint, proper management of sprint backlog items are implemented. The application group sprint backlog items by user stories / product backlogs which brought transparency and achievable to the sprint goal. Each sprint backlog items are driven by their product backlog parent item.

All the project objectives have been met as the applications are developed and delivered in time. These are achieved project objectives:

1. To analyse project requirements and its specifications by reviewing existing similar applications during Project 1 period.
3. To design and develop a prototype model of these Scrum Board applications during Project 1 period.
4. To develop and design web application to create and monitor the overall project activities and requirements in product backlog and sprint backlog within Project 2 period.
5. To develop and design a mobile application that serve as Scrum Board that has push notification function within Project 2 period.

These are unachieved project objectives:

2. To develop Scrum board applications which are easily learned and adapted by inexperienced Scrum user.
6. To evaluate the application functionalities with sufficient acceptance testing within Project 2 period.

Since there aren't sufficient acceptance testing and the right user to perform it. It couldn't prove that the Scrum Board applications has high usability and can be accepted by inexperience Scrum users.

The applications are tested, and end user are satisfied with the user interface as shown in results in the appendix. These are test results:

- Unit Test (100% Pass)
- Integration Test (100% Pass)
- User Acceptance Test (100% Pass, user satisfied with the Scrum Board applications)
- Usability Test (Good)

7.2 Recommendation for future works

Despite the applications were developed and fulfilled all the core specifications stated in the scope statement, the applications still have a lot of limitations and can be improved in the future. The table below shows some future enhancement with suggestion that could increase usability of the applications and increase the collaboration between team members.

Table 7.1: Table of recommendations for future work

No	Limitation	Suggestion and recommendations
1	Being a watcher of the sprint task, couldn't comment and interact with assignee of the sprint task **[Scrum Board mobile/web application]**	As an enhancement, the optional functional requirements of allowing Scrum Master / Member to post comments and upload files should be implemented in both Scrum Board mobile application and Scrum Board web application
2	Missing search and filter module in project management, sprint management, sprint backlog management, product backlog management. **[Scrum Board web application]**	As an enhancement, there should be a search and filter module which allows Scrum Master to easily find and access the item he / she wants. Imagine there are 20 ++ product backlog items in the list. It will be a struggle if the Scrum Master have to scroll through the list and look one by one. Search module is a must. The Scrum Board web application should provide more filtering options for the Scrum Board such as "Group by member" which allows Scrum Master to easily know one assignee is being allocated to how many sprint tasks.
3	Automated sprint management	To increase the productivity and efficiency of the Scrum Master, the Scrum Board web application should integrate with artificial intelligence / bot to

	[Scrum Board web application]	help Scrum Master to manage the sprint. These are some of the examples of automated feature: - Terminate sprint automatically when it is expired and send an email to notify the Scrum Master - Start date is planned and set according to the previous sprint estimated end date - Automatically move all unfinish sprint task to the next sprint And etc
4	Email notifications **[Scrum Board mobile/web application]**	Both applications should send email notification whenever the Scrum Master / Scrum Member make changes.
5	Registration verifying email **[Scrum Board mobile/web application]**	Email should be verified when a user creates a new account to prevent personal email is being used by unauthorised users and prevent user to create an account with non-existing email.

Other from the table above, these are the other prior limitations and recommendations that should be looked into:

Database Storage and Structure

NoSQL database is not suitable for the Scrum Board applications. Firebase is being used in current moment as it is easiest database to setup, and it is freemium. However, a relational database might be more suitable to maintain data consistency of the data structure. Relational database also provides more query options then firebase which doesn't require data processing to be done purposely after the data being fetched. Next, performance is an important criterion. Using relational database might have a little improvement in the performance compare to Firebase Firestore. These are the available options for relational database:

Figure 7.1: AWS Relational Databases Service

Figure 7.2: Microsoft Azure

CRON

Cron is time-base job scheduler and it is very useful to automate repetitive tasks. It is very important in automating sprint management in Scrum Board web application. For example, the Cron job is implemented and a script is being executed every 12:00am to check whether the sprint of the project is expired. If does, another script will be automatically executed to terminate the sprint. Moreover, it can be also used in the Scrum Board Mobile application. For instance, a script will be executed every few hours to notify the user about the almost due sprint tasks as a reminder. Example of Cron for websites:

Figure 7.3: Cron-job.org

REFERENCES

Akpata, E. and Riha, K., 2004. Can Extreme Programming be used by a Lone Programmer? *Systems Integration*, [online] pp.167–175. Available at: <http://faculty.ksu.edu.sa/ghazy/541_papers_sem1_2829/XP by Lone Programmer.pdf>.

ATIMI, 2019. *Firebase Spark SDK*. [online] Available at: <https://www.atimi.com/reviews-firebase-sdk/> [Accessed 20 Jun. 2019].

Carvalho, W.C.D.S., Rosa, P.F., Soares, M.D.S., Cunha, M.A.T. Da, Buiatte, L.C. and Da Cunha, M.A.T., 2012. A comparative analysis of the agile and traditional software development processes productivity. *Proceedings - International Conference of the Chilean Computer Science Society, SCCC*, pp.74–82.

Delcheva, Y.D., 2017. *Challenges during the transition to Agile Methodologies: A Holistic Overview*.

Dzhurov, Y., Krasteva, I. and Ilieva, S., 2009. Personal Extreme Programming–An Agile Process for Autonomous Developers. *International Conference on software, services & semantic technologies*, [online] (August 2016), pp.252–259. Available at: <https://www.researchgate.net/publication/229046039_Personal_Extreme_Programming-An_Agile_Process_for_Autonomous_Developers>.

Fadillah, T., 2018. *Scrum Board : Physical or Virtual?* [online] Available at: <https://medium.com/@tommydinuri/scrum-board-physical-or-virtual-9a18c014c575> [Accessed 25 Jun. 2019].

Grechanik, M., McKinley, K.S. and Perry, D.E., 2007. Recovering and using use-case-diagram-to-source-code traceability links.

Gunga, V., Kishnah, S. and Pudaruth, S., 2013. Design of A Multi-Agent System Architecture for the Scrum Methodology. *International Journal of Software Engineering & Applications*, 4(4), pp.1–18.

John, S.J., 2013. *Agile vs Traditional Project Management*. [online] Available at: <https://www.slideshare.net/saqibjaved/agile-vs-traditional-project-management> [Accessed 20 Jun. 2019].

Levison, M., 2013. *Choosing a Sprint Length – Shorter Trumps Longer*. [online] Available at: <https://agilepainrelief.com/notesfromatooluser/2013/07/choosing-sprint-length-shorter-trumps-longer.html> [Accessed 19 Jun. 2019].

Lynch, W., 2019. *The Brief of History of Scrum*. [online] Available at: <htt ps://medium.com/@warren2lynch/the-brief-of-history-of-scrum-15efb73b4701> [Accessed 15 Jul. 2019].

Madadipouy, K., 2015. An Examination and Evaluation of Agile Methodologies for Systems Development. *Australasian Journal of Computer Science*.

Mitchell, I., 2017. *No Walking Through a Definition of Done*. [online] Available at: <https://www.scrum.org/resources/blog/walking-through-definition-done> [Accessed 28 Jul. 2019].

Muslihat, D., 2018a. *7 Popular Project Management Methodologies And What They're Best Suited For*. [online] Available at: <https://zenkit.com/en/blog/7-popular-project-management-methodologies-and-what-theyre-best-suited-for/> [Accessed 20 Jun. 2019].

Muslihat, D., 2018b. *Scrum 101: An Introduction To Scrum Project Management*. [online] Available at: <https://zenkit.com/en/blog/scrum-101-an-introduction-to-scrum-project-management/> [Accessed 20 Jun. 2019].

Neagu, C., 2013. *Is There a Project Management Methodology for One-Person Projects?* [online] Available at: <https://www.rationalplan.com/projectmanagementblog/is-there-a-project-management-methodology-for-one-person-projects/> [Accessed 20 Jun. 2019].

Nishijima, R.T. and Dos Santos, P.D.J.G., 2013. THE CHALLENGE OF IMPLEMENTING SCRUM AGILE METHODOLOGY IN A TRADITIONAL DEVELOPMENT ENVIRONMENT. *INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY*, [online] 5(2), pp.98–108. Available at: <www.cirworld.com>.

Panchal, D., 2011. *What is the Definition of Done (DoD) in Agile?* [online] Available at: <https://www.solutionsiq.com/resource/blog-post/what-is-the-definition-of-done-dod-in-agile/> [Accessed 28 Jul. 2019].

Patel, U., 2016. *Why Firebase is the Best Mobile Backend as a Service?* [online] Available at: <https://www.tristatetechnology.com/blog/firebase-backend-mobile-app/>.

Permana, P.A.G., 2015. Scrum Method Implementation in a Software Development Project Management. *International Journal of Advanced Computer Science and Applications*, [online] 6(9), pp.198–204. Available at: <www.ijacsa.thesai.org>.

Pino, M. Del, 2018. *SPI and Agile Software Development*. [online] Available at:

<https://www.spiinc.com/single-post/2018/11/07/SPI-and-Agile-Software-Development> [Accessed 20 Jun. 2019].

Schwaber, K. and Sutherland, J., 2017. *The Scrum Guide™ The Definitive Guide to Scrum: The Rules of the Game*.

ScrumAlliance, 2017. *STATE OF SCRUM 2017-2018 scaling and agile transformation*.

ScrumAlliance, 2019. *Scrum Roles Demystified*. [online] Available at: <https://www.scrumalliance.org/agile-resources/scrum-roles-demystified> [Accessed 20 Jun. 2019].

Singh, R., 2016. *Easy way to understand SDLC*. [online] Available at: <https://www.dignitasdigital.com/blog/easy-way-to-understand-sdlc/> [Accessed 20 Jun. 2019].

Singh, V., 2018. *Introduction to Firebase*. [online] Available at: <https://medium.com/codingurukul/introduction-to-firebase-f9f6ccc8a785> [Accessed 20 Jun. 2019].

Smartsheet, 2019. *Physical Scrum Board vs. Smartsheet Online Scrum Board*. [online] Available at: <https://www.smartsheet.com/physical-scrum-board-vs-smartsheet-online-scrum-board> [Accessed 20 Jun. 2019].

Trnka, D., 2018. *Mobile App Development: React Native vs Native (iOS, Android)*. [online] Available at: <https://medium.com/mop-developers/mobile-app-development-react-native-vs-native-ios-android-49c5c168045b> [Accessed 19 Jul. 2019].

Umphress, D. and Agarwal, R., 2008. Extreme programming for a single person team. In: *Proceedings of the 46th Annual Southeast Regional Conference, 2008, Auburn, Alabama, March 28-29, 2008*. [online] pp.82–87. Available at: <https://www.researchgate.net/publication/220996614>.

APPENDICES

APPENDIX A: Scrum Board Applications UAT Form

Scrum Board Applications UAT Form

Please read and sign this form.

In this user acceptance test:

- You will be asked to perform certain tasks on your computer.
- We will also conduct an interview with you regarding the tasks you performed.

Participation in this user acceptance study is voluntary. All information will remain strictly confidential. The descriptions and findings may be used to help improve the Scrum Board applications (Web & Mobile) However, at no time will your name or any other identification be used. You can withdraw your consent to the experiment and stop participation at any time.

If you have any questions after today, please contact Low Zi Jian lowzijian1998@1utar.my.

I have read and understood the information on this form and had all my questions answered.

_____

Name: **Date:** 21 April 2020

1) **Are you experienced with applications that allow users to schedule and plan their work tasks? Select one of the options below.**
○ Yes ○ No
2) **Have you ever used / heard of Trello? Select one of the options below.**
○ Yes ○ No
3) **What do you expect from a simple board list, cards that helps you to keep track the tasks of the project? Comments on below:**
4) **Have you ever heard of Scrum Methodology / Kanban Methodology? Select one of the options below.**
○ Yes ○ No
5) **How long does you take to familiarize and fully understands the features and functionality of an application? Select one of the options below.**
○ Within a day ○ One week ○ A month

Scrum Board Web Application

Testing Date			
Testing Start Time			
Testing End Time			
Name of the tester **(s)**			
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	Scrum master is able to login to his/her account	Pass	
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-

Project Management Module	Create Project Scrum Master is able to create a new project.		
	Invite new members Scrum master is able to invite a new member while creating the project.		
	View Project Scrum Master is able to view the project.		
	Edit Project Scrum Master is able to edit the project.		
	Close Project Scrum Master is able to close a project.		
	Open Project Scrum Master is able to reopen a project.		
	Delete Project Scrum Master is able to delete a project.		
Product Backlog Management Module	Create Product Backlog Scrum Master is able to create a new product backlog item.		
	View Product Backlog Scrum Master is able to view product backlog item.		
	Edit Product Backlog		

	Scrum Master is able to edit product backlog item.		
	Delete Product Backlog Scrum Master is able to delete product backlog item.		
Sprint Backlog Management Module	Create Sprint Backlog Scrum Master is able to create a new sprint backlog item.		
	View Sprint Backlog Scrum Master is able to view sprint backlog item.		
	Edit Sprint Backlog Scrum Master is able to edit sprint backlog item.		
	Delete Sprint Backlog Scrum Master is able to delete sprint backlog item.		
Sprint Management Module	Create Sprint Scrum Master is able to create a new sprint.		
	View Sprint Scrum Master is able to view sprint.		
	Edit Sprint Scrum Master is able to edit sprint.		
	Delete Sprint		

	Scrum Master is able to delete sprint.		
	Delete Everything Scrum Master is able to delete sprint and its sprint backlog items.		
	Add activity in Sprint Scrum Master is able to add in sprint backlog item in sprint.		
	Start a Sprint Scrum Master is able to start sprint.		
View Scrum Board Module	View Scrum Board Scrum Master is able to view the scrum board of sprint.		
	View Scrum Board (Filter by product backlog) Scrum Master is able to view the scrum board of sprint filtered by specific product backlog item.		
	View Scrum Board (Group by status / story) Scrum Master is able to view the scrum board of sprint group by status / group by story		

	View Scrum Board (board view / list view) Scrum Master is able to view the scrum board of sprint in list view / board view.		
	Invite new member Scrum Master is able to add new member in the project.		
	Terminate Sprint Scrum Master is able to terminate current ongoing sprint of the project.		
View Archive Module	View Archive Scrum Master is able to view all the terminated sprints and its activities.		
	View Scrum Board Scrum Master is able to view the terminated sprint ‘s scrum board.		
Generate Reports Module	Generate PDF report Scrum Master is able to generate a pdf report / documents of project items.		
	Generate Excel report		

	Scrum Master is able to generate a excel report / documents of project items.		
Profile Management Module	Update Profile Picture Scrum master is able to change his account profile picture.		
View Notifications and feeds Module	View activity feeds Scrum master is able to view the activity feeds of all projects.		

Scrum Board Mobile Application

Testing Date			
Testing Start Time			
Testing End Time			
Name of the tester **(s)**			
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	Scrum member is able to login to his/her account		
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-

View Projects	View Project Detail Scrum member is able to view the project.		
	View Product Backlogs of the project Scrum member is able to view the project ‘s product backlogs items		
	View Project current Sprint Info Scrum member is able to view the project ‘s current sprint info.		
	View all projects Scrum member is able to view all the projects that he / she is currently participating, both active and close projects		
Update assigned sprint tasks Module	Update Sprint task’s status Scrum member is able to update the status of the assigned sprint task.		
View Scrum Board of current sprint tasks Module	View Scrum Board of current sprint Scrum member is able to view the scrum board of the project.		
Get unassigned sprints tasks Module	Get unassigned sprint task		

	Scrum member is able to assign himself / herself to an unassigned sprint task.		
View Notifications and Feeds Module	View activity feeds Scrum member is able to view the activity feeds of projects		
Profile Management Module	Update Profile Picture Scrum member is able to change his / her account profile picture.		

	Strongly Disagree 1	**Disagree 2**	**Neutral 3**	**Agree 4**	**Strongly Agree 5**
1. I think that I would use the applications for planning and scheduling my projects.					
2. I found the applications are unnecessarily complex.					
3. I found the applications are relatively easy to use.					
4. I found the applications have all the mandatory feature.					

5. I found that I need the help of technical person to be able to use the applications.					
6. I found the applications can be easily navigate without a lot of backtracking and data re-entry.					
7. I believe that other users would learn to user the applications effortlessly.					
8. I found the applications have very awkward or inappropriate content.					
9. I felt very confident using the applications.					
10. I need to learn of things before I could get going with the applications.					

1. What did you like best about the scrum board applications (web & mobile)?

_____

_____

_____

2. What did you like least about the scrum board applications (web & mobile)?

_____

_____

_____

3. If you were to describe the scrum board applications (web & mobile) to a colleague in a sentence of two, what would you say?

_____

_____

_____

4. Do you have any final comments or questions?

_____

_____

APPENDIX B: UAT form filled up by UAT tester (1)

Scrum Board Applications UAT Form

Please read and sign this form.

In this user acceptance test:

- You will be asked to perform certain tasks on your computer.
- We will also conduct an interview with you regarding the tasks you performed.

Participation in this user acceptance study is voluntary. All information will remain strictly confidential. The descriptions and findings may be used to help improve the Scrum Board applications (Web & Mobile) However, at no time will your name or any other identification be used. You can withdraw your consent to the experiment and stop participation at any time.

If you have any questions after today, please contact Low Zi Jian lowzijian1998@1utar.my.

I have read and understood the information on this form and had all my questions answered.

Signature,

Name: Chong Zi Ming **Date:** 21 April 2020

1) **Are you experienced with applications that allow users to schedule and plan their work tasks? Select one of the options below.**
◦ Yes ◦ No
2) **Have you ever used / heard of Trello? Select one of the options below.**
◦ Yes ◦ No
3) **What do you expect from a simple board list, cards that helps you to keep track the tasks of the project? Comments on below:**
Notify me.
4) **Have you ever heard of Scrum Methodology / Kanban Methodology? Select one of the options below.**
◦ Yes ◦ No
5) **How long does you take to familiarize and fully understands the features and functionality of an application? Select one of the options below.**
◦ Within a day ◦ One week ◦ A month

Scrum Board Web Application

Testing Date	21 April 2020		
Testing Start Time	4.25 pm		
Testing End Time	5.20 pm		
Name of the tester **(s)**	Chong Zi Ming		
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	Scrum master is able to login to his/her account	Pass	
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-

Project Management Module	Create Project Scrum Master is able to create a new project.	Pass	
	Invite new members Scrum master is able to invite a new member while creating the project.	Pass	- should be have a cross button to delete.
	View Project Scrum Master is able to view the project.	Pass	
	Edit Project Scrum Master is able to edit the project.	Pass	
	Close Project Scrum Master is able to close a project.	Pass	
	Open Project Scrum Master is able to reopen a project.	Pass	
	Delete Project Scrum Master is able to delete a project.	Pass	
Product Backlog Management Module	Create Product Backlog Scrum Master is able to create a new product backlog item.	Pass	
	View Product Backlog Scrum Master is able to view product backlog item.	Pass	

	Edit Product Backlog Scrum Master is able to edit product backlog item.	Pass	
	Delete Product Backlog Scrum Master is able to delete product backlog item.	Pass	
Sprint Backlog Management Module	Create Sprint Backlog Scrum Master is able to create a new sprint backlog item.	Pass	
	View Sprint Backlog Scrum Master is able to view sprint backlog item.	Pass	
	Edit Sprint Backlog Scrum Master is able to edit sprint backlog item.	Pass	
	Delete Sprint Backlog Scrum Master is able to delete sprint backlog item.	Pass	
Sprint Management Module	Create Sprint Scrum Master is able to create a new sprint.	Pass	
	View Sprint Scrum Master is able to view sprint.	Pass	
	Edit Sprint	Pass	

	Scrum Master is able to edit sprint.		
	Delete Sprint Scrum Master is able to delete sprint.	Pass	
	Delete Everything Scrum Master is able to delete sprint and its sprint backlog items.	Pass	
	Add activity in Sprint Scrum Master is able to add in sprint backlog item in sprint.	Pass	
	Start a Sprint Scrum Master is able to start sprint.	Pass	
View Scrum Board Module	View Scrum Board Scrum Master is able to view the scrum board of sprint.	Pass	
	View Scrum Board (Filter by product backlog) Scrum Master is able to view the scrum board of sprint filtered by specific product backlog item.	Pass	

	View Scrum Board (Group by status / story) Scrum Master is able to view the scrum board of sprint group by status / group by story	Pass	
	View Scrum Board (board view / list view) Scrum Master is able to view the scrum board of sprint in list view / board view.	Pass	
	Invite new member Scrum Master is able to add new member in the project.	Pass	- should not have delete member function inside
	Terminate Sprint Scrum Master is able to terminate current ongoing sprint of the project.	Pass	
View Archive Module	View Archive Scrum Master is able to view all the terminated sprints and its activities.	Pass	
	View Scrum Board	Pass	

	Scrum Master is able to view the terminated sprint ‘s scrum board.		
Generate Reports Module	Generate PDF report Scrum Master is able to generate a pdf report / documents of project items.	Pass	
	Generate Excel report Scrum Master is able to generate a excel report / documents of project items.	Pass	
Profile Management Module	Update Profile Picture Scrum master is able to change his account profile picture.	Pass	- should display filename
View Notifications and feeds Module	View activity feeds Scrum master is able to view the activity feeds of all projects.	Pass	

Scrum Board Mobile Application

Testing Date	21 April 2020		
Testing Start Time	5.20 pm		
Testing End Time	6.00 pm		
Name of the tester **(s)**	Chong Zi Ming		
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	Scrum member is able to login to his/her account	Pass	
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-
View Projects	View Project Detail Scrum member is able to view the project.	Pass	
	View Product Backlogs of the project Scrum member is able to view the project ‘s product backlogs items	Pass	
	View Project current Sprint Info Scrum member is able to view the project ‘s current sprint info.	Pass	
	View all projects Scrum member is able to view all the projects that	Pass	

	he / she is currently participating, both active and close projects		
Update assigned sprint tasks Module	Update Sprint task's status Scrum member is able to update the status of the assigned sprint task.	Pass	
View Scrum Board of current sprint tasks Module	View Scrum Board of current sprint Scrum member is able to view the scrum board of the project.	Pass	
Get unassigned sprints tasks Module	Get unassigned sprint task Scrum member is able to assign himself / herself to an unassigned sprint task.	Pass	
View Notifications and Feeds Module	View activity feeds Scrum member is able to view the activity feeds of projects	Pass	- should be able to hide / collapse according to the projects
Profile Management Module	Update Profile Picture Scrum member is able to change his / her account profile picture.	Pass	

	Strongly Disagree 1	**Disagree 2**	**Neutral 3**	**Agree 4**	**Strongly Agree 5**
1. I think that I would use the applications for planning and scheduling my projects.				√	
2. I found the applications are unnecessarily complex.		√			
3. I found the applications are relatively easy to use.				√	
4. I found the applications have all the mandatory feature.				√	
5. I found that I need the help of technical person to be able to use the applications.			√		
6. I found the applications can be easily navigate without a lot of backtracking and data re-entry.				√	
7. I believe that other users would learn to user the applications effortlessly.			√		
8. I found the applications have very awkward or inappropriate content.	√				
9. I felt very confident using the applications.					√
10. I need to learn of things before I could get going with the applications.		√			

1. What did you like best about the scrum board applications (web & mobile)?

UI design is beautiful and the icons is easily for me to understand and find specific pages in both web and mobile.

2. What did you like least about the scrum board applications (web & mobile)?

Do not have real-time update on the data updated on mobile and web.

3. If you were to describe the scrum board applications (web & mobile) to a colleague in a sentence of two, what would you say?

Beautiful UI and easy to use.

4. Do you have any final comments or questions?

No.

APPENDIX C: UAT form filled up by UAT tester (2)

Scrum Board Applications UAT Form

Please read and sign this form.

In this user acceptance test:

- You will be asked to perform certain tasks on your computer.
- We will also conduct an interview with you regarding the tasks you performed.

Participation in this user acceptance study is voluntary. All information will remain strictly confidential. The descriptions and findings may be used to help improve the Scrum Board applications (Web & Mobile) However, at no time will your name or any other identification be used. You can withdraw your consent to the experiment and stop participation at any time.

If you have any questions after today, please contact Low Zi Jian lowzijian1998@1utar.my.

I have read and understood the information on this form and had all my questions answered.

Kaixiang

Name: Chin Kai Xiang **Date:** 21 April 2020

1) **Are you experienced with applications that allow users to schedule and plan their work tasks? Select one of the options below.**
◦ Yes ◦ No
2) **Have you ever used / heard of Trello? Select one of the options below.**
◦ Yes ◦ No
3) **What do you expect from a simple board list, cards that helps you to keep track the tasks of the project? Comments on below:**
Give an overview of tasks as well as the details of tasks
4) **Have you ever heard of Scrum Methodology / Kanban Methodology? Select one of the options below.**
◦ Yes ◦ No
5) **How long does you take to familiarize and fully understands the features and functionality of an application? Select one of the options below.**
◦ Within a day ◦ One week ◦ A month

Scrum Board Web Application

Testing Date			
Testing Start Time			
Testing End Time			
Name of the tester **(s)**			
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	Scrum master is able to login to his/her account	Pass	
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-

Project Management Module	Create Project Scrum Master is able to create a new project.	Pass	Should use delete button for delete member
	Invite new members Scrum master is able to invite a new member while creating the project.	Pass	
	View Project Scrum Master is able to view the project.	Pass	
	Edit Project Scrum Master is able to edit the project.	Pass	
	Close Project Scrum Master is able to close a project.	Pass	
	Open Project Scrum Master is able to reopen a project.	Pass	
	Delete Project Scrum Master is able to delete a project.	Pass	
Product Backlog Management Module	Create Product Backlog Scrum Master is able to create a new product backlog item.	Pass	
	View Product Backlog Scrum Master is able to view product backlog item.	Pass	

	Edit Product Backlog Scrum Master is able to edit product backlog item.	Pass	
	Delete Product Backlog Scrum Master is able to delete product backlog item.	Pass	
Sprint Backlog Management Module	Create Sprint Backlog Scrum Master is able to create a new sprint backlog item.	Pass	
	View Sprint Backlog Scrum Master is able to view sprint backlog item.	Pass	
	Edit Sprint Backlog Scrum Master is able to edit sprint backlog item.	Pass	
	Delete Sprint Backlog Scrum Master is able to delete sprint backlog item.	Pass	
Sprint Management Module	Create Sprint Scrum Master is able to create a new sprint.	Pass	
	View Sprint Scrum Master is able to view sprint.	Pass	

	Edit Sprint Scrum Master is able to edit sprint.	Pass	
	Delete Sprint Scrum Master is able to delete sprint.	Pass	
	Delete Everything Scrum Master is able to delete sprint and its sprint backlog items.	Pass	
	Add activity in Sprint Scrum Master is able to add in sprint backlog item in sprint.	Pass	
	Start a Sprint Scrum Master is able to start sprint.	Pass	
View Scrum Board Module	View Scrum Board Scrum Master is able to view the scrum board of sprint.	Pass	
	View Scrum Board (Filter by product backlog) Scrum Master is able to view the scrum board of sprint filtered by specific product backlog item.	Pass	Should fix the filter buttons

	View Scrum Board (Group by status / story) Scrum Master is able to view the scrum board of sprint group by status / group by story	Pass	
	View Scrum Board (board view / list view) Scrum Master is able to view the scrum board of sprint in list view / board view.	Pass	
	Invite new member Scrum Master is able to add new member in the project.	Pass	
	Terminate Sprint Scrum Master is able to terminate current ongoing sprint of the project.	Pass	
View Archive Module	View Archive Scrum Master is able to view all the terminated sprints and its activities.	Pass	
	View Scrum Board	Pass	

	Scrum Master is able to view the terminated sprint ‘s scrum board.		
Generate Reports Module	Generate PDF report Scrum Master is able to generate a pdf report / documents of project items.	Pass	
	Generate Excel report Scrum Master is able to generate a excel report / documents of project items.	Pass	
Profile Management Module	Update Profile Picture Scrum master is able to change his account profile picture.	Pass	
View Notifications and feeds Module	View activity feeds Scrum master is able to view the activity feeds of all projects.	Pass	

Scrum Board Mobile Application

Testing Date			
Testing Start Time			
Testing End Time			
Name of the tester **(s)**			
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	Scrum member is able to login to his/her account		
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-
View Projects	View Project Detail Scrum member is able to view the project.	Pass	
	View Product Backlogs of the project Scrum member is able to view the project 's product backlogs items	Pass	
	View Project current Sprint Info Scrum member is able to view the project 's current sprint info.	Pass	
	View all projects Scrum member is able to view all the projects that	Pass	

	he / she is currently participating, both active and close projects		
Update assigned sprint tasks Module	Update Sprint task's status Scrum member is able to update the status of the assigned sprint task.	Pass	
View Scrum Board of current sprint tasks Module	View Scrum Board of current sprint Scrum member is able to view the scrum board of the project.	Pass	The product backlog filter is not intuitive
Get unassigned sprints tasks Module	Get unassigned sprint task Scrum member is able to assign himself / herself to an unassigned sprint task.	Pass	
View Notifications and Feeds Module	View activity feeds Scrum member is able to view the activity feeds of projects	Pass	
Profile Management Module	Update Profile Picture Scrum member is able to change his / her account profile picture.	Pass	

	Strongly Disagree 1	**Disagree 2**	**Neutral 3**	**Agree 4**	**Strongly Agree 5**
1. I think that I would use the applications for planning and scheduling my projects.					√
2. I found the applications are unnecessarily complex.			√		
3. I found the applications are relatively easy to use.				√	
4. I found the applications have all the mandatory feature.					√
5. I found that I need the help of technical person to be able to use the applications.			√		
6. I found the applications can be easily navigate without a lot of backtracking and data re-entry.				√	
7. I believe that other users would learn to user the applications effortlessly.				√	
8. I found the applications have very awkward or inappropriate content.	√				
9. I felt very confident using the applications.				√	
10. I need to learn of things before I could get going with the applications.				√	

1. What did you like best about the scrum board applications (web & mobile)?

- Beautiful and intuitive UI allows easy understanding of information
- Easy tracking of tasks.

2. What did you like least about the scrum board applications (web & mobile)?

No real-time data update

3. If you were to describe the scrum board applications (web & mobile) to a colleague in a sentence of two, what would you say?

Beautiful and intuitive UI together with all the core features

4. Do you have any final comments or questions?

No

APPENDIX D: UAT form filled up by UAT tester (3)

Scrum Board Applications UAT Form

Please read and sign this form.

In this user acceptance test:

- You will be asked to perform certain tasks on your computer.
- We will also conduct an interview with you regarding the tasks you performed.

Participation in this user acceptance study is voluntary. All information will remain strictly confidential. The descriptions and findings may be used to help improve the Scrum Board applications (Web & Mobile) However, at no time will your name or any other identification be used. You can withdraw your consent to the experiment and stop participation at any time.

If you have any questions after today, please contact Low Zi Jian lowzijian1998@1utar.my.

I have read and understood the information on this form and had all my questions answered.

Name: Lee Hoe Mun **Date:** 21 April 2020

1) **Are you experienced with applications that allow users to schedule and plan their work tasks? Select one of the options below.**
◦ Yes ◦ No
2) **Have you ever used / heard of Trello? Select one of the options below.**
◦ Yes ◦ No
3) **What do you expect from a simple board list, cards that helps you to keep track the tasks of the project? Comments on below:**
To specify the expected start date and end date. Also the actual start date and end date so I can measure if I am on track for a particular project.
4) **Have you ever heard of Scrum Methodology / Kanban Methodology? Select one of the options below.**
◦ Yes ◦ No
5) **How long does you take to familiarize and fully understands the features and functionality of an application? Select one of the options below.**
◦ Within a day ◦ One week ◦ A month

Scrum Board Web Application

Testing Date	21-4-2020		
Testing Start Time	8:40PM		
Testing End Time	9:20PM		
Name of the tester **(s)**	Lee Hoe Mun		
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	Scrum master is able to login to his/her account	Fail	Something went wrong.
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-

Project Management Module	Create Project Scrum Master is able to create a new project.	Pass	Good.
	Invite new members Scrum master is able to invite a new member while creating the project.	Pass	Good.
	View Project Scrum Master is able to view the project.	Pass	Good.
	Edit Project Scrum Master is able to edit the project.	Pass	Good.
	Close Project Scrum Master is able to close a project.	Pass	Good.
	Open Project Scrum Master is able to reopen a project.	Pass	Good.
	Delete Project Scrum Master is able to delete a project.	Pass	Good.
Product Backlog Management Module	Create Product Backlog Scrum Master is able to create a new product backlog item.	Pass	Good.
	View Product Backlog Scrum Master is able to view product backlog item.	Pass	Good.

	Edit Product Backlog Scrum Master is able to edit product backlog item.	Pass	Good.
	Delete Product Backlog Scrum Master is able to delete product backlog item.	Pass	Good.
Sprint Backlog Management Module	Create Sprint Backlog Scrum Master is able to create a new sprint backlog item.	Pass	Good.
	View Sprint Backlog Scrum Master is able to view sprint backlog item.	Pass	Good.
	Edit Sprint Backlog Scrum Master is able to edit sprint backlog item.	Pass	Good.
	Delete Sprint Backlog Scrum Master is able to delete sprint backlog item.	Pass	Good.
Sprint Management Module	Create Sprint Scrum Master is able to create a new sprint.	Pass	Good.
	View Sprint Scrum Master is able to view sprint.	Pass	Good.
	Edit Sprint	Pass	Good.

	Scrum Master is able to edit sprint.		
	Delete Sprint Scrum Master is able to delete sprint.	Pass	Good.
	Delete Everything Scrum Master is able to delete sprint and its sprint backlog items.	Pass	Good.
	Add activity in Sprint Scrum Master is able to add in sprint backlog item in sprint.	Pass	Good.
	Start a Sprint Scrum Master is able to start sprint.	Pass	Good.
View Scrum Board Module	View Scrum Board Scrum Master is able to view the scrum board of sprint.	Pass	Good.
	View Scrum Board (Filter by product backlog) Scrum Master is able to view the scrum board of sprint filtered by specific product backlog item.	Pass	Good.

	View Scrum Board (Group by status / story) Scrum Master is able to view the scrum board of sprint group by status / group by story	Pass	Good.
	View Scrum Board (board view / list view) Scrum Master is able to view the scrum board of sprint in list view / board view.	Pass	Good.
	Invite new member Scrum Master is able to add new member in the project.	Pass	Good.
	Terminate Sprint Scrum Master is able to terminate current ongoing sprint of the project.	Pass	Good.
View Archive Module	View Archive Scrum Master is able to view all the terminated sprints and its activities.	Pass	Good.
	View Scrum Board	Pass	Good.

	Scrum Master is able to view the terminated sprint ‘s scrum board.		
Generate Reports Module	Generate PDF report Scrum Master is able to generate a pdf report / documents of project items.	Pass	Good.
	Generate Excel report Scrum Master is able to generate a excel report / documents of project items.	Pass	Good
Profile Management Module	Update Profile Picture Scrum master is able to change his account profile picture.	Pass	Good.
View Notifications and feeds Module	View activity feeds Scrum master is able to view the activity feeds of all projects.	Pass	Good.

Scrum Board Mobile Application

Testing Date	21-4-2020		
Testing Start Time	9:25PM		
Testing End Time	9:40PM		
Name of the tester **(s)**	Lee Hoe Mun		
Test Module	Test Scenario	Pass/ Fail	Comments
Login Module	Scrum member is able to login to his/her account	Pass	Good.
Register Module	-	-	-
Forgot Password / Reset Password Module	-	-	-
View Projects	View Project Detail Scrum member is able to view the project.	Pass	Good.
	View Product Backlogs of the project Scrum member is able to view the project 's product backlogs items	Pass	Good.
	View Project current Sprint Info Scrum member is able to view the project 's current sprint info.	Pass	Good.
	View all projects Scrum member is able to view all the projects that	Pass	Good.

	he / she is currently participating, both active and close projects		
Update assigned sprint tasks Module	Update Sprint task's status Scrum member is able to update the status of the assigned sprint task.	Pass	Good.
View Scrum Board of current sprint tasks Module	View Scrum Board of current sprint Scrum member is able to view the scrum board of the project.	Pass	Good.
Get unassigned sprints tasks Module	Get unassigned sprint task Scrum member is able to assign himself / herself to an unassigned sprint task.	Pass	Good.
View Notifications and Feeds Module	View activity feeds Scrum member is able to view the activity feeds of projects	Pass	Good.
Profile Management Module	Update Profile Picture Scrum member is able to change his / her account profile picture.	Pass	Good.

	Strongly Disagree 1	Disagree 2	Neutral 3	Agree 4	Strongly Agree 5
1. I think that I would use the applications for planning and scheduling my projects.			√		
2. I found the applications are unnecessarily complex.				√	
3. I found the applications are relatively easy to use.		√			
4. I found the applications have all the mandatory feature.		√			
5. I found that I need the help of technical person to be able to use the applications.				√	
6. I found the applications can be easily navigate without a lot of backtracking and data re-entry.				√	
7. I believe that other users would learn to user the applications effortlessly.		√			
8. I found the applications have very awkward or inappropriate content.		√			
9. I felt very confident using the applications.			√		
10. I need to learn of things before I could get going with the applications.			√		

1. What did you like best about the scrum board applications (web & mobile)?

The overall design of the application is very nice.

2. What did you like least about the scrum board applications (web & mobile)?

Navigation is sometime hard to find.

3. If you were to describe the scrum board applications (web & mobile) to a colleague in a sentence of two, what would you say?

Design and the feel of the application is up to par, worth a try.

4. Do you have any final comments or questions?

No.