COS60011 – Technology Design Project

Team Design Concept

Team 3

Team members:

Student Name: Mitchell Anton	**Student ID:** 103750586
Student Name: Lucas Qin	**Student ID:** 103527269
Student Name: Minji Kim	**Student ID:** 103510175
Student Name: Natalie Huang	**Student ID:** 103698990
Student Name: Jin Chen	**Student ID:** 103685934
Student Name: Mona Ma	**Student ID:** 103699029

Workshop time: Thursday 18:30-20:30

Tutor: Dr. Chris McCarthy

Table of content

1 Introduction **2**

1.1 Background and Motivation 2

1.2 Project Goals/Objectives 2

2 Method and Design Frames **6**

3 Features, UI design and Technologies **10**

3.1 Feature Priority 10

3.2 User interface design and prototyping 10

3.3 Technologies Used 12

4 Conclusion **13**

5 Resources **13**

Appendix **14**

Functional requirements **14**

1 Introduction

1.1 Background and Motivation

To ensure the maintenance of public assets, local city councils must rely on the public to notify them on the state of assets. Traditionally, this has been through the means of either contacting the local city council or filling out a web-based form. This is an inefficient and ineffective method. Firstly, the tracking of assets relies heavily on the public phoning, emailing or filling out a web-based form. Once an issue is submitted, it is impossible to know the current state of the asset, or when the issue will be completely resolved. Additionally, there is no way to know if the issue has already been reported. Finally, it is difficult in some instances to precisely determine the location of the issue. This uncovers our motivation for this project, which is to provide a more efficient and streamlined solution in reporting issues, to tackle the above problems.

1.2 Project Goals/Objectives

Considering our solution will take the form of a mobile app, the team aims to learn more about mobile app development, especially considering our lack of knowledge in this domain. Ideally, by the end of the project, we will have developed a fully functioning mobile app which solves the current problems in public asset issue reporting. Additionally, we also aim to learn how to make an app appealing for users to use. This is particularly important for this app because of the importance of public engagement.

The desired outcomes and benefits for the team are:

1. A better understanding of mobile app development.
2. More experience in developing software as a team.
3. More experience in developing software that has a practical use in society.
4. Developing group work and communication skills.

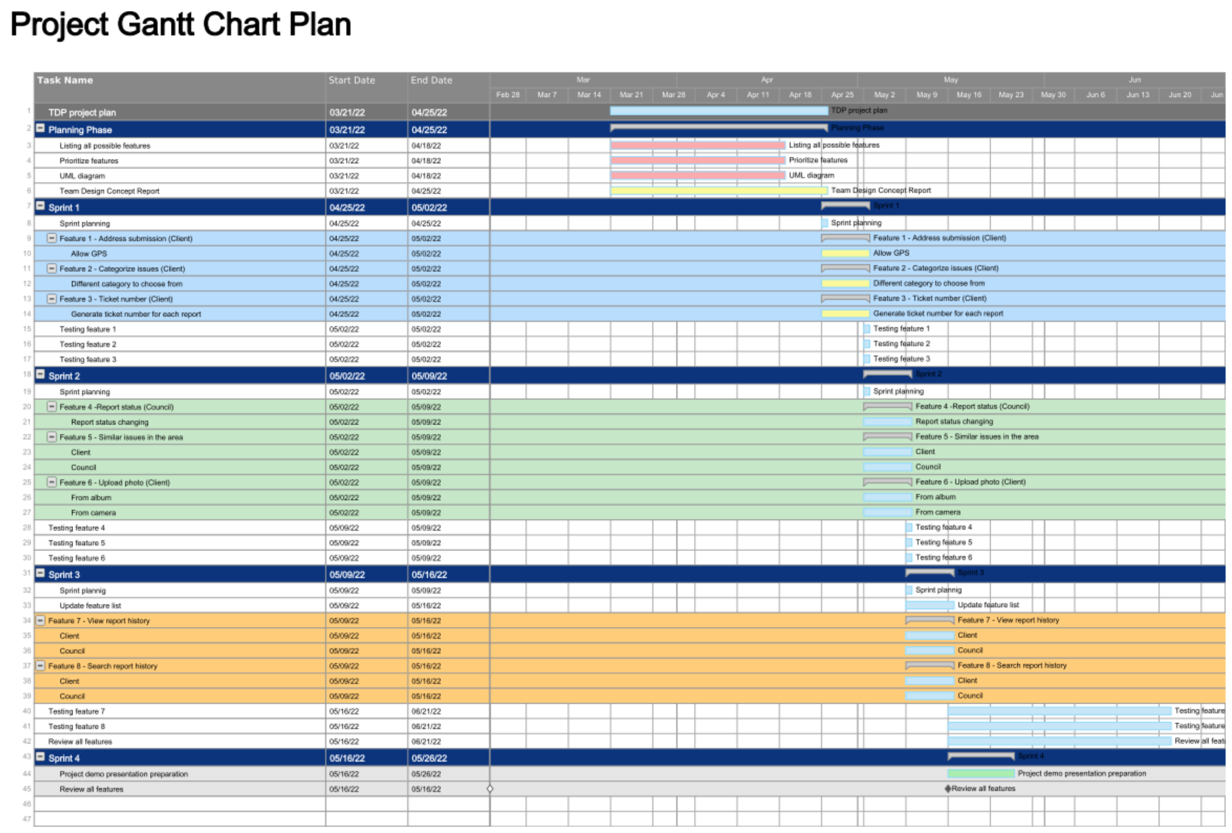
The desired outcomes and benefits for the client are:

1. A better solution in reporting issues regarding the state of public assets, one which is more efficient, resulting in quicker response times from local city councils.
2. Keeps clients updated on the state of public assets when they submit a report.
3. For the respective local city council, they can more easily locate where an issue is.
4. For the local city council, there is less time wasted on phone calls or emails. The user does not need to communicate with the council directly to submit an issue.

1.3 Project Schedule/Plan

After prioritizing the features we wish to implement, a detailed plan has been developed based on the project deadline. The project is divided into a planning phase and another 4 sprints. In each sprint, we plan to work on three or two features based on the difficulty and priority of the features. Each sprint includes a planning stage, production stage, and a testing stage. In sprint four, an overall review of all the features that have been developed will be done. As we might face possible difficulty while developing some of the features, a feature review is needed before the last sprint. In the last sprint, we will work on any less important features we might be able to develop, or work on fixing any possible problems that happened while developing the features during the last few sprints. As we are an agile work team, all the work will be done in an iterative process.

Project schedule is listed in the Gantt Chart as below.



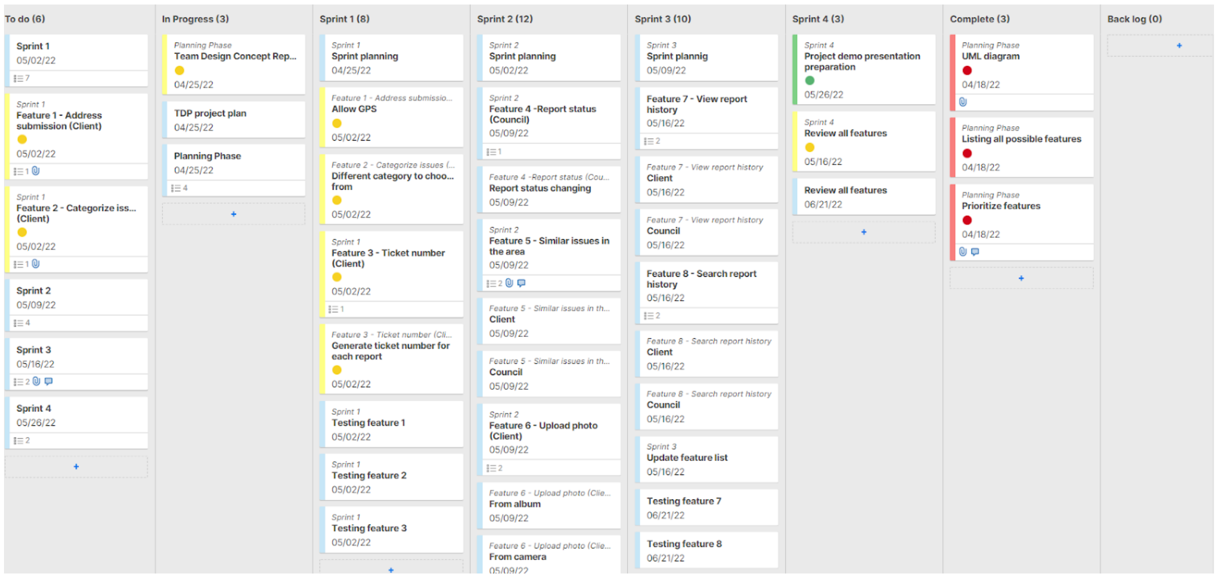
Project Gantt Chart Plan

	Task Name	Start Date	End Date
1	TDP project plan	03/21/22	04/25/22
2	Planning Phase	03/21/22	04/25/22
3	Listing all possible features	03/21/22	04/18/22
4	Prioritize features	03/21/22	04/18/22
5	UML diagram	03/21/22	04/18/22
6	Team Design Concept Report	03/21/22	04/25/22
7	Sprint 1	04/25/22	05/02/22
8	Sprint planning	04/25/22	04/25/22
9	Feature 1 - Address submission (Client)	04/25/22	05/02/22
10	Allow GPS	04/25/22	05/02/22
11	Feature 2 - Categorize issues (Client)	04/25/22	05/02/22
12	Different category to choose from	04/25/22	05/02/22
13	Feature 3 - Ticket number (Client)	04/25/22	05/02/22
14	Generate ticket number for each report	04/25/22	05/02/22
15	Testing feature 1	05/02/22	05/02/22
16	Testing feature 2	05/02/22	05/02/22
17	Testing feature 3	05/02/22	05/02/22
18	Sprint 2	05/02/22	05/09/22
19	Sprint planning	05/02/22	05/02/22
20	Feature 4 -Report status (Council)	05/02/22	05/09/22
21	Report status changing	05/02/22	05/09/22
22	Feature 5 - Similar issues in the area	05/02/22	05/09/22
23	Client	05/02/22	05/09/22
24	Council	05/02/22	05/09/22
25	Feature 6 - Upload photo (Client)	05/02/22	05/09/22
26	From album	05/02/22	05/09/22
27	From camera	05/02/22	05/09/22
28	Testing feature 4	05/09/22	05/09/22
29	Testing feature 5	05/09/22	05/09/22
30	Testing feature 6	05/09/22	05/09/22
31	Sprint 3	05/09/22	05/16/22
32	Sprint plannig	05/09/22	05/09/22
33	Update feature list	05/09/22	05/16/22
34	Feature 7 - View report history	05/09/22	05/16/22
35	Client	05/09/22	05/16/22
36	Council	05/09/22	05/16/22
37	Feature 8 - Search report history	05/09/22	05/16/22
38	Client	05/09/22	05/16/22
39	Council	05/09/22	05/16/22
40	Testing feature 7	05/16/22	06/21/22
41	Testing feature 8	05/16/22	06/21/22
42	Review all features	05/16/22	06/21/22
43	Sprint 4	05/16/22	05/26/22
44	Project demo presentation preparation	05/16/22	05/26/22
45	Review all features	05/16/22	05/16/22
46			
47			

Figure 1: Gantt Chart

Link:
https://app.smartsheet.com/sheets/pwPwxP9cr4QgHvM2mXcvvCRgxFGHHXH5PRF8GH71?view=gantt

A Kanban style to-do list is also created for our project. Tasks are divided into several sections. All the tasks in each sprint are listed under the specific sprint section. The tasks that are currently undertaken are listed under the 'In Progress' section. When a task is completed, it will be moved to the 'Complete' section. A backlog section is also created for any possible tasks to be put on hold in the future.



To do (6)

Sprint 1 — 05/02/22

Sprint 1 — Feature 1 - Address submission (Client) — 05/02/22

Sprint 1 — Feature 2 - Categorize iss... (Client) — 05/02/22

Sprint 2 — 05/09/22

Sprint 3 — 05/16/22

Sprint 4 — 05/26/22

In Progress (3)

Planning Phase — Team Design Concept Rep... — 04/25/22

TDP project plan — 04/25/22

Planning Phase — 04/25/22

Sprint 1 (8)

Sprint 1 — Sprint planning — 04/25/22

Feature 1 - Address submissio... — Allow GPS — 05/02/22

Feature 2 - Categorize issues (... — Different category to choo... from — 05/02/22

Sprint 1 — Feature 3 - Ticket number (Client) — 05/02/22

Feature 3 - Ticket number (Cli... — Generate ticket number for each report — 05/02/22

Sprint 1 — Testing feature 1 — 05/02/22

Sprint 1 — Testing feature 2 — 05/02/22

Sprint 1 — Testing feature 3 — 05/02/22

Sprint 2 (12)

Sprint 2 — Sprint planning — 05/02/22

Sprint 2 — Feature 4 -Report status (Council) — 05/09/22

Feature 4 -Report status (Cou... — Report status changing — 05/09/22

Sprint 2 — Feature 5 - Similar issues in the area — 05/09/22

Feature 5 - Similar issues in th... — Client — 05/09/22

Feature 5 - Similar issues in th... — Council — 05/09/22

Sprint 2 — Feature 6 - Upload photo (Client) — 05/09/22

Feature 6 - Upload photo (Clie... — From album — 05/09/22

Feature 6 - Upload photo (Clie... — From camera — 05/09/22

Sprint 3 (10)

Sprint 3 — Sprint plannig — 05/09/22

Feature 7 - View report history — 05/16/22

Feature 7 - View report history — Client — 05/16/22

Feature 7 - View report history — Council — 05/16/22

Feature 8 - Search report history — 05/16/22

Feature 8 - Search report history — Client — 05/16/22

Feature 8 - Search report history — Council — 05/16/22

Sprint 3 — Update feature list — 05/16/22

Testing feature 7 — 06/21/22

Testing feature 8 — 06/21/22

Sprint 4 (3)

Sprint 4 — Project demo presentation preparation — 05/26/22

Sprint 4 — Review all features — 05/16/22

Review all features — 06/21/22

Complete (3)

Planning Phase — UML diagram — 04/18/22

Planning Phase — Listing all possible features — 04/18/22

Planning Phase — Prioritize features — 04/18/22

Back log (0)

Figure 2: To-do list

Currently, we have completed the UML diagram design, and the features prioritizing discussion. These tasks are moved to 'Complete'. As we can see from the 'Progress' section, our team design concept report and project planning is still in progress.

The following table shows the relationship between project tasks and team members.

Table 1: Human Resource Management

	Jin Chen	**Lucas Qin**	**Natalie Huang**	**Mona Ma**	**Minji Kim**	**Mitchell Anton**
Define Problem	A	C	C	C	R	R
Design Solution	C	R	A	R	C	C
User Interface Design	I	I	I	C	R	I
Develop database	R	C	I	A	I	I
Coding Features	R	R	R	A	R	R
Testing	R	A	R	R	R	R

Key:
R- Responsible for completing the work
A-Accountable for ensuring task completion
C-Consulted before any decisions are made
I-Informed of when an action has been made

The following table shows the resource management of the project. Each feature is assigned to a different number of team members based on the difficulty to implement. Team members may have different working hours acquired in each sprint. In case of any possible delay when executing the plan, the team members who have less working hours during that sprint can help with the delayed work.

Table 2: The resource management

	Features	**People needed**	**Working hours per person**	**Assign to**
Sprint 1	Feature 1	2	30	Mona Ma Lucas Qin
	Feature 2	1	15	Minji Kim
	Feature 3	1	30	Natalie Huang
	Testing	1	10	Jin Chen
Sprint 2	Feature 4	2	20	Natalie Huang Jin Chen
	Feature 5	2	25	Lucas Qin Mona Ma
	Feature 6	2	20	Mitchell Anton Minji Kim
	Testing	1	10	Minji Kim
Sprint 3	Feature 7	1	10	Mitchell
	Feature 8	1	20	Jin Chen
	Testing	1	10	Natalie Huang
Sprint 4	Project demo preparation	6	20	All

2 Method and Design Frames

2.1 Project objectives and methods:

1. To simplify the issue reporting method.

   Solution: The traditional methods like emails required users to report their issues in their own words, they might missed important descriptions or cause misunderstanding due to different level of language expression, apart from that, the council will need to manually take record of the email, and when the record is done, the council need to send an email back to confirm more details or provide a track number. This project aims to add additional convenience to both councils and the public in managing and dealing with public assets by providing an mobile app, which features in easy login(or no login), auto-add location, auto-identify similar reports from same area, pre-built issues classifications, easy tracking progress by unique number provided and easy to provide photos in app. Compared to the traditional solution, the new solution saves both council and the public a lot of effort in managing community assets.

2. To help council manage the monitoring and maintenance of a diverse range of assets and resources.

   Solution: A category selection will be added in the report submission process. The user will have to choose from several specific categories while logging an issue. The category chosen will also affect some other features in the project solution. For example, both client and council work may be able to search for issues based on different categories . Also, it is also possible to locate the similar issue around the area in the same category to avoid duplicate issue submission.

3. To help both client and council worker get an accurate location from issue reports.

   Solution: The council workers and staff should know the accurate location of the issues have been reported by clients. The application will provide the GPS for clients when they report an issue. The client can allow the application to get the location from the system GPS and they can also select a location from Google map of the region. After clients choose a location on the map, they can edit the specific number or street name to make the location more accurate. The council staff and workers can check the location of reporting issues by both address and position located by Google map, which will help them schedule and resolve issues quickly and efficiently.

4. Allow both client and council worker to know if there is some conflicting infromation, and reduce reporting of duplicate issues.

   Solution: To avoid the public reporting the same issue repeatedly, the application will provide a set of reports in the nearby location from the same category when the client enters the location of the issue, which could be the same street, same intersection, or within specific distance. The system will ask the client to compare and check these reports, and the client can add some information to the same report or exit from the function if the issue has been reported. This function avoids conflicting information and reduces duplicate issues and will save councils a lot of effort in dealing with repeated reports. On the council side, if they found the same issues in different reports they can combine these reports together and all ticket numbers will be under this issue. The council worker will follow the combined report and resolve the issue.

5. Track submitted issue reports.

   Solution: To help clients track a report, we provide a ticket number generated automatically for each report submitted. The ticket number should be unique.

6. Viewing the up to date information of a reported issue from the resident portal. Show the time when the last state was updated.

   Solution: The client is able to track the state of a logged issue by entering a ticket number. An issue may have several status. The council work is able to change the issue status during different stages of the progress. The council worker may leave comments about the details of the latest update of the issue. These comments, and the time of the comments left, are viewable by the resident.

7. Viewing the up to date information of a reported issue from the council portal. Show the time when the last state was updated. Change the state of the logged issue and leave comments.

   Solution: The council worker can also access a logged issue immediately by using a specific ticket number. The council worker can view the status of the issue, and even change the status of the issue. When an issue is logged, it is tagged as ‘Open’. When a council worker confirmed the issue and assigned it to the specific team, the issue could be changed to ‘In progress’. When a job is done and the council worker has confirmed it is done properly, the council worker could change the status of the issue to ‘Complete’. While changing the status of the issue, the change time will be recorded. All the status changes as well as the changing time can be viewed by both client and council worker. The council worker can also leave comments to update the current information of currently logged issues. For example, when the job is assigned to a specific team, the council worker can leave a comment mentioning that the job is to be done by a specific date.

2.2 Design frame

To achieve all these objectives and complete the whole project, we decided to create a mobile application for the client and a web application for the council. Before starting to design the functions, we design the frame of the project and a UML diagram has been shown below.

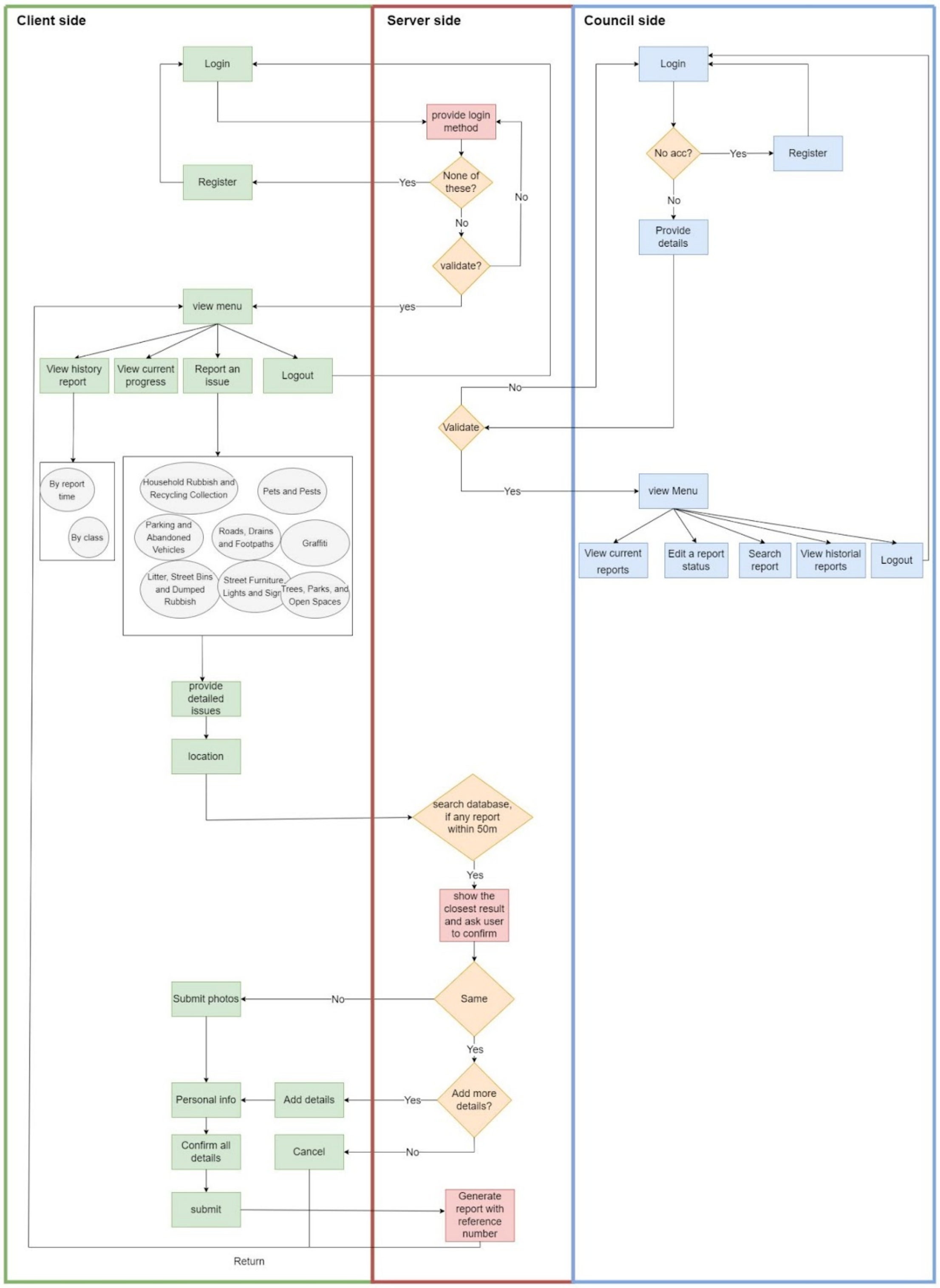


Figure 3: UML diagram

3 Features, UI design and Technologies

3.1 Feature Priority

To decide on the feature priority of our project, we conducted functional requirements brainstorming first (Appendix, Functional Requirements); the following table summarizes the features we consider, from the most important to the lowest priority we need to develop.

Priority	Features
High	Address Submission/GPS(Client side)
High	Categorize Issues(Client side)
High	Get Ticket Number(Client side)
High	Track Ticket Number(Client side)
High	Change Report Status(Council side)
Medium	View Report History(Client side)
Medium	View Report History(Council side)
Medium	Possible similar issue in the area(Client side)
Medium	Possible similar issue in the area(Council side)
Low	Search Report History(Client side)
Low	Search Report History(Council side)
Low	Camera Use(Client side)

Table 3: Feature priorities

3.2 User interface design and prototyping

The UI design of the project application utilizes Figma. It is designed to be divided into Android applications for clients and web applications that will be used by council workers. Based on our requirements and UML diagram, UI designs were developed. Moreover, prototyping was created to validate that our app can communicate with clients precisely.

Prototype Links:
https://www.figma.com/file/p3zHK0POfNghPvc34DR5Oh/City-council-mobile-app-Design?node-id=0%3A1

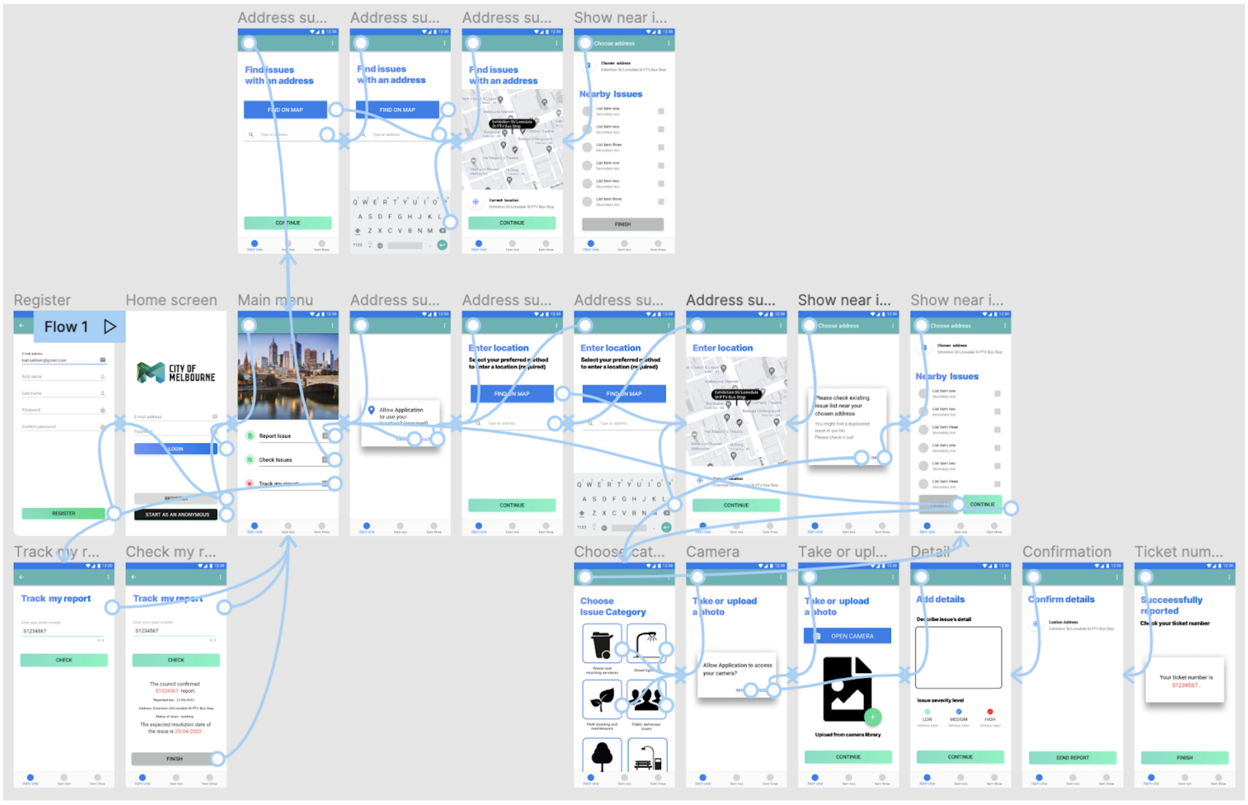
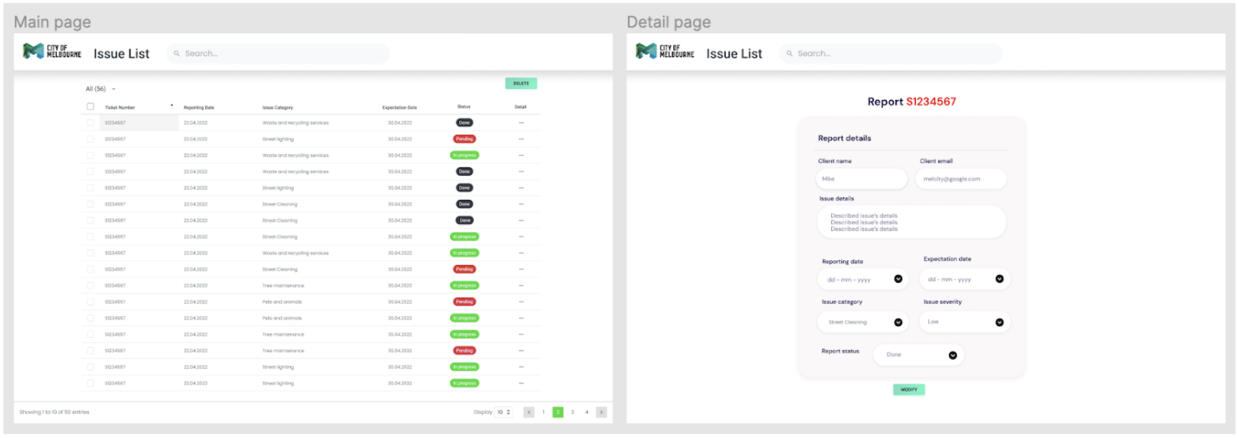


Figure 4: User interface design and prototyping of our mobile application

We also plan to develop a simple web application for the council worker side. Council workers only can access this page and manage the reports. The web will display the data of reports and give council workers the authorization to modify the status.


Main page

Issue List

Search...

All (56)

DELETE

Ticket Number	Reporting Date	Issue Category	Expectation Date	Status	Detail
S1234567	22.04.2022	Waste and recycling services	30.04.2022	Done	···
S1234567	22.04.2022	Street lighting	30.04.2022	Pending	···
S1234567	22.04.2022	Waste and recycling services	30.04.2022	In progress	···
S1234567	22.04.2022	Waste and recycling services	30.04.2022	Done	···
S1234567	22.04.2022	Street lighting	30.04.2022	Done	···
S1234567	22.04.2022	Street Cleaning	30.04.2022	Done	···
S1234567	22.04.2022	Street Cleaning	30.04.2022	Done	···
S1234567	22.04.2022	Street Cleaning	30.04.2022	In progress	···
S1234567	22.04.2022	Waste and recycling services	30.04.2022	In progress	···
S1234567	22.04.2022	Street Cleaning	30.04.2022	Pending	···
S1234567	22.04.2022	Tree maintenance	30.04.2022	In progress	···
S1234567	22.04.2022	Pets and animals	30.04.2022	Pending	···
S1234567	22.04.2022	Pets and animals	30.04.2022	In progress	···
S1234567	22.04.2022	Tree maintenance	30.04.2022	In progress	···
S1234567	22.04.2022	Tree maintenance	30.04.2022	Pending	···
S1234567	22.04.2022	Street lighting	30.04.2022	In progress	···
S1234567	22.04.2022	Street lighting	30.04.2022	In progress	···

Showing 1 to 10 of 50 entries

Display 10 ‹ 1 2 3 4 ›

Detail page

Issue List

Search...

Report S1234567

Report details

Client name: Mike

Client email: melcity@google.com

Issue details:
Described issue's details
Described issue's details
Described issue's details

Reporting date: dd - mm - yyyy

Expectation date: dd - mm - yyyy

Issue category: Street Cleaning

Issue severity: Low

Report status: Done

MODIFY

Figure 5: User interface design of our web application

3.3 Technologies Used

The above design with Figma can easily convert to Android Studio with full Java support. Before converting to Android Studio, we will review the font and device size. Then, to organize layers and folders such as the image and texts, the icon of one screen needs to be in a frame to prevent errors. Once we have the correct frame size, we can export it to Android Studio and grab the Java and Res files of the design.

This project will use Java and Kotlin as programming languages to build an Android application in Android Studio. On the one hand, Java is a mature object-oriented language with huge community support and rich API & libraries, that means developers can always find help from the community and save a lot of effort when using APIs and libraries. Its English-like syntax is also easy for beginners to understand compared to low-level languages like C++. On the other hand, Kotlin is a rather new language that was announced to be the official language of Android by Google in 2017, the main characteristic for Kotlin is concise syntax, which means less code and less error, also less code improves readability and helps maintenance. Most importantly, Kotlin is compatible with Java, that means Java and Kotlin can co-exist in the same project, Kotlin file can use Java class and framework, developers using Java can always change to Kotlin at any time. With great documents and increasing support from Google, Kotlin will become the most suitable language for Android development. Apart from languages choosing, we will use a NoSQL database in the implementation, and firebase as our database platform. For version control, we will use git and github.

For our IDE, we will be using Android Studio. Android Studio is tailor made for Android development. Thus, it provides a natural developer experience. For example, it is an easy process to connect Android emulators or physical Android devices to the IDE to test the application. Another example of this is the GUI it provides to make layouts. Instead of writing XML, the developer can simply drag and drop views onto the screen which automatically generates XML code. Although other IDE's can be used such as VSCode, there are no advantages in doing so.

We are using a NoSQL database because it allows users to use unstructured data such as photos, videos, PDF files, emails and other format data. It doesn't require any predefined schema ,. Also, scaling a NoSQL database is cheaper than a relational database, because a NoSQL database can add more servers to scale horizontally while relational databases scale vertically, by changing to a more powerful server. Our solution for this project is to use Firebase database, because it provides a platform for developing mobile application, it includes a lot of features and functionalities for developers, such as hosting, authentication, analytics and tracking application usage in the console. It is a good choice for developing a small project.

Of course, the above languages will not be used for the web app. Instead, we will use PHP for the backend and HTML, CSS, and JavaScript on the frontend. The choice of PHP on the backend is due to the simplicity of the web app. PHP allows us to quickly develop an app due to its native features such as database connection. This is unlike other languages such as Python where a framework must be used.

4 Conclusion

This project is for building a solution for the council to manage its assets efficiently. On the one hand, the council wants the public to be able to report an issue precisely and conveniently, and the public can also track the issue that has been reported. On the other hand, the council wants to identify, track, and communicate on specific assets or issues unambiguously, in addition, each issue reported by the public could provide a state and time point for the council to clearly understand the current state of the issue.

5 Resources

Android Studio and SDK tools | Android Developers. (2022). https://developer.android.com/studio

Build Your First Android App in Java, https://developer.android.com/codelabs/build-your-first-android-app#0

Fabio Staiano. (2022). *Designing and Prototyping Interfaces with Figma*. Packt Publishing.

Firebase. (2019b). Cloud Firestore | Firebase: https://firebase.google.com/docs/firestore.

Java Vs Kotlin: Which one is better to learn in 2022? https://codersera.com/blog/java-vs-kotlin/

Kotlin vs Java: Which is better for android app development? https://www.xenonstack.com/blog/kotlin-andriod

Kotlin vs Java: Important Differences That You Must Know, https://hackr.io/blog/kotlin-vs-java

MongoDB. (n.d.). *NoSQL vs Relational Databases*: https://www.mongodb.com/scale/nosql-vs-relational-databases#:~:text=Relational%20databases%20are%20table%2Dbased

Moroney. (2017). *The Definitive Guide to Firebase Build Android Apps on Google's Mobile Platform* (1st ed. 2017.). Apress. https://doi.org/10.1007/978-1-4842-2943-9

Spolsky. (2001). *User Interface Design for Programmers* (1st ed. 2001.). Apress. https://doi.org/10.1007/978-1-4302-0857-0

Trivedi. (2020). *Android Application Development with Kotlin : Build Your First Android App in No Time*. BPB Publications.

Appendix

Functional requirements

1. Issue Report Submission (Client)

Req 1: Categorise Issues

Requirement: When submitting an issue, users shall be able to put it under a category chosen from a list.

Rationale: This will make it easier for the council to sort through data.

Req 2: Address Submission

Requirement: When submitting an issue, users shall be able to also enter the address where the issue was found, either manually or using GPS.

Rationale: For the issue to be fixed, the location is paramount.

Req 3: Camera Use When Submitting Issue

Requirement: When submitting an issue, the user shall be able to submit a photo of the issue.

Rationale: This gives the council an idea of what the issue is. Additionally, it may assist the council in prioritising the issue.

Req 4: Issue Severity

Requirement: Users shall be able to submit a severity rating with the report. This shall be selected from a list of options.

Rationale: This will help the council prioritise which issues to fix.

2. Report history (Client)

Req 2: Viewing Report History

Requirement: Users shall be able to view a history of reports made by everyone. These shall be sortable by metrics such as time, category, location, or ticket number.

Rationale: This can ensure that the user does not submit a duplicate report.

Req 3: Searching Report History

Requirement: Users shall be able to search through the report history (as discussed in 1.6) using some sort of search engine.

Rationale: As the number of reports grows, it may be harder to filter through reports. Thus, a search engine is necessary to make it easier.

3. User Identification (Client)

Req 1: User Authentication

Requirement: Users shall be able to login and register to the app.

Rationale: This allows the council to know who submits a report. This can help them ban users who use the app maliciously. Additionally, should the council need to contact a user who submits an issue report, they can do so.

Req 2: Anonymous Issue Submission

Requirement: Users shall be able to submit an issue without creating an account.

Rationale: Most users are going to be using this app while on the go. They may not be bothered to create an account. They may simply want to submit an issue without having to worry about creating an account.

Req 3: Add Contact Details to Submission

Requirement: When a user submits an issue report, they shall have the choice to add contact details.

Rationale: Some users may not want to create an account, but still submit contact details. Thus, this requirement is the solution for that.

Req 4: Add Ticket Number

Requirement: When a user submits an issue report, a ticket number should be created and given to the user.

Rationale: This gives the user a way to keep track of the issue. (This means we do not need user authentication, although it is still useful).

4. Viewing User Reports (Council Worker)

Req 1: Identifying Users who Submit Issue Reports

Requirement: Council workers shall be able to view the contact details of the person who submitted a report (if the person who submitted the report did so with an account or if the person who submitted added contact details manually).

Rationale: This allows the council workers to contact the person who submitted the report if necessary. Additionally, should anyone use the app maliciously, assuming the malicious user did so under an account, they can be banned by council workers.

5. Managing Issue Reports (Council Worker)

Req 1: Changing Report Status

Requirement: Council workers shall be able to change the status of an issue report. Once a report is completed, they can change it to complete.

Rationale: This helps keep track of which reports are completed and which are not.

Req 2: Moving Completed Reports to Log

Requirement: Council workers shall be able to move completed reports to a completed reports log.

Rationale: This helps the council workers keep track of which issues reports are completed and which aren't.

Req 3: Location of Issues

Requirement: The system shall be able to identify issues from the same area and present them to the user so that the user won't report the same issue.
Rationale: Some issues might be submitted by different multiple users and cause confusion.

Req 4: Issue Severity

Requirement: Council workers shall be able to change report severity.