

FLOOD MONITORING AND ALERT SYSTEM

BY

KOK HOW MENG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JAN 2024

REPORT STATUS DECLARATION FORM

Title: Flood Monitoring and Alert System

Academic Session: Jan 2024

I KOK HOW MENG
(CAPITAL LETTER)


declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,



(Author's signature)



(Supervisor's signature)

Address:

224, Jalan Awana 12A/1 Blok C,
Taman Cheras Awana, 43200
Batu 9 Cheras, Selangor.

Ts Tan Teik Boon

Supervisor's name

Date: 25/4/2024

Date: 25/4/2024

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 21 JUNE 2011	Page No.: 1 of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 25/4/2024

SUBMISSION OF FINAL YEAR PROJECT /~~DISSERTATION/THESIS~~

It is hereby certified that *Kok How Meng* (ID No: *20ACB02193*)
has completed this final year project/ ~~dissertation/ thesis~~* entitled “ *Flood Monitoring and Alert System* ” under the supervision of Ts Tan Teik Boon (Supervisor) from the Department of
Computer Science , Faculty of Information and Communication Technology , and
_____ (Co-Supervisor)* from the Department of _____,
Faculty/Institute* of _____.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,




(*Kok How Meng*)

*Delete whichever not applicable

DECLARATION OF ORIGINALITY

I declare that this report entitled “**FLOOD MONITORING AND ALERT SYSTEM**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature :  _____

Name : Kok How Meng

Date : 25/4/2024

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Ts Tan Teik Boon, for offering me this bright opportunity to work on a mobile application development project on the topic of “Flood Monitoring and Alert System.” His patience, invaluable guidance, practical advice, and continuous support have helped me tremendously throughout the duration of this project.

I also want to sincerely thank my classmates and friends, who have always motivated me and provided both moral and technical support during hard times. Last but not least, I would like to express my deepest gratitude to my parents and my family for their unwavering support, understanding, and ongoing encouragement throughout the course.

ABSTRACT

Due to the growing frequency and severity of natural disasters, particularly flooding occurrences, the field of environmental monitoring and disaster management has made significant advancements in recent years. This project addresses the critical need for an effective and practical approach for monitoring flood incidents and disseminating alerts by developing a flood monitoring and alert system. One of the most catastrophic natural disasters is flooding, which causes fatalities and significant property damage. Despite the government's efforts to decrease the effects of floods by implementing various structural and non-structural strategies, flood concerns in Malaysia caused by the monsoon and flash floods remain. Besides that, some of the existing systems have limited applicability to cover all areas prone to flooding due to limitations in monitoring infrastructure and Android OS versions. To overcome these challenges, this project aims to design and implement a mobile application for flood monitoring and alert systems to deliver fast and precise flood alerts. The project adopts agile methodology and utilises tools such as Flutter with Dart language, Google Maps, Google Cloud, APIs, and Firebase. This project also integrates data from various sources, including current water level data, rainfall data, weather forecasts, and weather warning data. The system will consist of four major modules: the monitoring module, the alerting module, the map module, and the flood management module. The flood monitoring and alert system is a significant advancement in the flood management process. It provides accurate and real-time information for users to monitor flood risk levels and status. Additionally, users can receive alerts about potential flood events and rising water levels via push notifications. It also provides a user-friendly interface for easy access to flood information using geospatial visualisation.

TABLE OF CONTENTS

TITLE PAGE	i
REPORT STATUS DECLARATION FORM	ii
FYP THESIS SUBMISSION FORM	iii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	x
LIST OF TABLES	xv
LIST OF ABBREVIATIONS	xvii
CHAPTER 1 INTRODUCTION	1
1.1 Problem Statement and Motivation	1
1.2 Project Objectives	2
1.3 Project Scope and Direction	4
1.4 Impact, Significance and Contributions	5
1.5 Report Organisation	6
CHAPTER 2 LITERATURE REVIEW	7
2.1 Review of the Existing Flood Monitoring and Alert System	7
2.1.1 MyPublicInfoBanjir	7
2.1.2 BWDB Flood App	11
2.1.3 FloodAlert	15
2.1.4 My Flood Risk Accra	19
2.2 Summary	23
2.3 Critical Remarks	25
2.4 Review of the Technologies	27
2.4.1 Flutter	27
2.4.2 Google Firebase	27
2.4.3 TensorFlow Lite	28

CHAPTER 3 SYSTEM METHODOLOGY/APPROACH (FOR DEVELOPMENT-BASED PROJECT)	29
3.1 Methodology	29
3.2 System Requirements	31
3.2.1 Functional Requirements	31
3.2.2 Non-functional Requirements	32
3.3 System Design Diagram	33
3.3.1 System Architecture Diagram	33
3.3.2 Use Case Diagram and Description	35
3.3.3 Activity Diagram	45
CHAPTER 4 SYSTEM DESIGN	53
4.1 System Block Diagram	53
4.1.1 Flutter Application	53
4.1.2 Flood Detection	54
4.2 Database Design	58
4.3 User Interface Design	59
CHAPTER 5 SYSTEM IMPLEMENTATION (FOR DEVELOPMENT-BASED PROJECT)	68
5.1 Hardware Setup	68
5.2 Software Setup	69
5.3 Setting and Configuration	74
5.4 System Operation (with Screenshot)	80
5.4.1 Authentication Module	80
5.4.2 Monitoring Module	82
5.4.3 Map Module	89
5.4.4 Alerting Module	93
5.4.5 Flood Management Module	95
5.5 Implementation Issues and Challenges	96
5.6 Concluding Remark	97

CHAPTER 6 SYSTEM EVALUATION AND DISCUSSION	98
6.1 System Testing and Performance Metrics	98
6.2 Testing Setup and Result	100
6.2.1 Flutter Application	100
6.2.2 Flood Detection	123
6.3 Project Challenges	126
6.4 Objectives Evaluation	127
6.5 Concluding Remark	127
CHAPTER 7 CONCLUSION AND RECOMMENDATION	128
7.1 Conclusion	128
7.2 Recommendation	129
REFERENCES	130
WEEKLY LOG	A-1
POSTER	B-1
PLAGIARISM CHECK RESULT	C-1
FYP2 CHECKLIST	E-1

LIST OF FIGURES

Figure Number	Title	Page
Figure 2.1.1	Home Page of MyPublicInfoBanjir	7
Figure 2.1.2	Setting for POI & Auto Alert	8
Figure 2.1.3	Additional Features of MyPublicInfoBanjir	8
Figure 2.1.4	Alert Page and SOS Page of MyPublicInfoBanjir	9
Figure 2.1.5	Dashboard of BWDB Flood App	11
Figure 2.1.6	Water Level Forecast Page of BWDB Flood App	12
Figure 2.1.7	Station Details Page of BWDB Flood App	12
Figure 2.1.8	Current Bulletin Page and Flood Warning Message Page	13
Figure 2.1.9	Additional Features of BWDB Flood App	13
Figure 2.1.10	Setting for Monitoring Location	15
Figure 2.1.11	Main Page of FloodAlert	16
Figure 2.1.12	Measures Page of FloodAlert	16
Figure 2.1.13	Water Level Graph of FloodAlert	17
Figure 2.1.14	Settings for FloodAlert	17
Figure 2.1.15	Home Page of My Flood Risk Accra	19
Figure 2.1.16	Literacy Information based on Flood Intensity	20
Figure 2.1.17	Flood Map of My Flood Risk Accra	20
Figure 2.1.18	Literacy Information in form of Infographics	21
Figure 2.1.19	Rainfall Page of My Flood Risk Accra	21
Figure 3.1.1	Agile Software Development	30
Figure 3.3.1	System Architecture Diagram	33
Figure 3.3.2	Use Case Diagram of the Proposed System	35
Figure 3.3.3	Login/Register Activity Diagram	45
Figure 3.3.4	Monitor Flood Information Activity Diagram	46
Figure 3.3.5	View Reports Activity Diagram	47
Figure 3.3.6	Manage Points of Interest Activity Diagram	48
Figure 3.3.7	View Alerts Activity Diagram	49
Figure 3.3.8	View Map Activity Diagram	50

Figure 3.3.9	View Flood Safety Tips Activity Diagram	51
Figure 3.3.10	Submit Report Activity Diagram	52
Figure 4.1.1	Block Diagram of the Flutter Application	53
Figure 4.1.2	Block Diagram of the Flood Detection	54
Figure 4.2.1	Database Design of the Proposed System	58
Figure 4.3.1	Login Page	59
Figure 4.3.2	Register Page	59
Figure 4.3.3	Reset Password Page	60
Figure 4.3.4	Home Page	60
Figure 4.3.5	Map Page	61
Figure 4.3.6	Alert Page	61
Figure 4.3.7	Guide Page	62
Figure 4.3.8	Report Flood Page	62
Figure 4.3.9	Monitoring Page	63
Figure 4.3.10	Station Details Page	63
Figure 4.3.11	POI Page	64
Figure 4.3.12	Add POI Page	64
Figure 4.3.13	Weather Forecast Page	65
Figure 4.3.14	Weather Warning Page	65
Figure 4.3.15	Report Page	66
Figure 4.3.16	Manage Report Page	66
Figure 4.3.17	Flood Status Page	67
Figure 5.2.1	Installation of Android Studio	70
Figure 5.2.2	Installation of Android Studio (Cont.)	70
Figure 5.2.3	Installation of Visual Studio Code	71
Figure 5.2.4	Installation of Visual Studio Code (Cont.)	71
Figure 5.2.5	Installation of Flutter	72
Figure 5.2.6	Add Flutter to PATH	72
Figure 5.2.7	Add Flutter to PATH (Cont.)	73
Figure 5.2.8	Run “flutter doctor” Command	73
Figure 5.2.9	Creation of Google Firebase	74
Figure 5.3.1	Android Studio Configuration	74
Figure 5.3.2	Flutter SDK Path	75

Figure 5.3.3	Create New Project	75
Figure 5.3.4	Flutter Extension	75
Figure 5.3.5	Dart Extension	76
Figure 5.3.6	Awesome Flutter Snippets Extension	76
Figure 5.3.7	Register App	77
Figure 5.3.8	Download and Add Config File	77
Figure 5.3.9	Add Dependencies	78
Figure 5.3.10	Initialise Firebase	78
Figure 5.3.11	Add Intent Filter	79
Figure 5.3.12	Handle Incoming Messages	79
Figure 5.3.13	Handle Incoming Messages (Cont.)	79
Figure 5.4.1	Login Screen	80
Figure 5.4.2	Register Screen	80
Figure 5.4.3	Reset Password Screen	80
Figure 5.4.4	Logout Screen	80
Figure 5.4.5	Home Screen	82
Figure 5.4.6	Monitoring Screen	82
Figure 5.4.7	Search State Results	82
Figure 5.4.8	Station Details Screen	82
Figure 5.4.9	POI Screen	83
Figure 5.4.10	Add POI Screen	83
Figure 5.4.11	Weather Forecast Screen (1/3)	84
Figure 5.4.12	Weather Forecast Screen (2/3)	84
Figure 5.4.13	Weather Forecast Screen (3/3)	84
Figure 5.4.14	Forecast Details Screen	84
Figure 5.4.15	Weather Warning Screen	85
Figure 5.4.16	Warning Details Screen	85
Figure 5.4.17	Flood Report Screen	86
Figure 5.4.18	My Report Screen	86
Figure 5.4.19	Flood Status Screen (1/6)	86
Figure 5.4.20	Flood Status Screen (2/6)	86
Figure 5.4.21	Flood Status Screen (3/6)	87
Figure 5.4.22	Flood Status Screen (4/6)	87

Figure 5.4.23	Flood Status Screen (5/6)	87
Figure 5.4.24	Flood Status Screen (6/6)	87
Figure 5.4.25	Location Service Disabled	89
Figure 5.4.26	Access Device's Location	89
Figure 5.4.27	User's Current Location	89
Figure 5.4.28	Marker Clustering	89
Figure 5.4.29	Monitoring Station Location	90
Figure 5.4.30	Station Details	90
Figure 5.4.31	Flood Report Location	90
Figure 5.4.32	Report Details	90
Figure 5.4.33	Navigation to Google Maps	91
Figure 5.4.34	Search Location Function	91
Figure 5.4.35	Search Location Results	91
Figure 5.4.36	Notification Disabled	93
Figure 5.4.37	Allow Notification	93
Figure 5.4.38	Alert Screen	93
Figure 5.4.39	Report Flood Screen	95
Figure 5.4.40	Guide Screen	95
Figure 6.1.1	Precision Score Formula	99
Figure 6.1.2	Recall Score Formula	99
Figure 6.1.3	F1-Score Formula	99
Figure 6.1.4	Accuracy Score Formula	99
Figure 6.2.1	Valid Email and Password	102
Figure 6.2.2	Successfully Login	102
Figure 6.2.3	Valid User Information	102
Figure 6.2.4	Successfully Registered	102
Figure 6.2.5	Valid Email Address	103
Figure 6.2.6	Successfully Sent	103
Figure 6.2.7	Password Reset Link	103
Figure 6.2.8	Enter New Password	103
Figure 6.2.9	Successfully Logout	104
Figure 6.2.10	Invalid Email or Password	105
Figure 6.2.11	Invalid User Information	105

Figure 6.2.12	Invalid Email Address	106
Figure 6.2.13	Empty Email and Password	107
Figure 6.2.14	Empty Input Fields	107
Figure 6.2.15	Empty Email Field	108
Figure 6.2.16	Empty State Name	109
Figure 6.2.17	Valid Selection	110
Figure 6.2.18	Successfully Added	110
Figure 6.2.19	Reload Page	110
Figure 6.2.20	POI is Already Added	111
Figure 6.2.21	No POI is Selected	112
Figure 6.2.22	Confirm POI Deletion	113
Figure 6.2.23	Successfully Deleted	113
Figure 6.2.24	Reload Page	114
Figure 6.2.25	Before Clicking Valid	115
Figure 6.2.26	After Clicking Valid	115
Figure 6.2.27	Select Platform	116
Figure 6.2.28	Navigate to Platform	116
Figure 6.2.29	Confirm Report Deletion	117
Figure 6.2.30	Successfully Deleted	117
Figure 6.2.31	Failed to Predict	118
Figure 6.2.32	Failed to Copy	118
Figure 6.2.33	Insufficient Rainfall Data	119
Figure 6.2.34	Invalid Rainfall Data	119
Figure 6.2.35	Empty Rainfall Data	120
Figure 6.2.36	Valid Report Details	121
Figure 6.2.37	Successfully Submitted	121
Figure 6.2.38	Empty Report Details	122
Figure 6.2.39	First Test Result	125
Figure 6.2.40	Second Test Result	125
Figure 6.2.41	Third Test Result	125
Figure 6.2.42	Fourth Test Result	125
Figure 6.2.43	Fifth Test Result	125

LIST OF TABLES

Table Number	Title	Page
Table 2.2.1	Table of Comparison	23
Table 2.2.2	Summary of Existing System	24
Table 3.3.1	Login/Register Use Case Description	37
Table 3.3.2	Monitor Flood Information Use Case Description	38
Table 3.3.3	View Reports Use Case Description	39
Table 3.3.4	Manage Points of Interest Use Case Description	40
Table 3.3.5	View Alerts Use Case Description	41
Table 3.3.6	View Map Use Case Description	42
Table 3.3.7	View Flood Safety Tips Use Case Description	43
Table 3.3.8	Submit Report Use Case Description	44
Table 4.1.1	Data Description of Malaysia Flood Dataset	55
Table 5.1.1	Specifications of Laptop	68
Table 5.1.1	Specifications of Smartphone	68
Table 6.1.1	Confusion Matrix Table	98
Table 6.2.1	Test Case for AUTHENTICATION-01	101
Table 6.2.2	Test Case for AUTHENTICATION-02	105
Table 6.2.3	Test Case for AUTHENTICATION-03	107
Table 6.2.4	Test Case for MONITORING-01	108
Table 6.2.5	Test Case for MONITORING-02	109
Table 6.2.6	Test Case for MONITORING-03	111
Table 6.2.7	Test Case for MONITORING-04	112
Table 6.2.8	Test Case for MONITORING-05	113
Table 6.2.9	Test Case for MONITORING-06	114
Table 6.2.10	Test Case for MONITORING-07	115
Table 6.2.11	Test Case for MONITORING-08	116
Table 6.2.12	Test Case for MONITORING-09	117
Table 6.2.13	Test Case for MONITORING-10	118
Table 6.2.14	Test Case for MONITORING-11	117

Table 6.2.15	Test Case for FLOODMANAGEMENT-01	121
Table 6.2.16	Test Case for FLOODMANAGEMENT-02	121
Table 6.2.17	Hyperparameter Tuning	123
Table 6.2.18	Confusion Matrix of the Base Model	123
Table 6.2.19	Confusion Matrix of the Tuned Model	123
Table 6.2.20	Evaluation Results of the Models	124
Table 6.2.21	Testing Results	124

LIST OF ABBREVIATIONS

<i>ANN</i>	Artificial Neural Network
<i>API</i>	Application Programming Interface
<i>DID</i>	Department of Irrigation and Drainage
<i>FCM</i>	Firebase Cloud Messaging
<i>FN</i>	False Negatives
<i>FP</i>	False Positives
<i>GPS</i>	Global Positioning System
<i>IDE</i>	Integrated Development Environment
<i>MET</i>	Malaysian Meteorological Department
<i>OS</i>	Operating System
<i>POI</i>	Point of Interest
<i>TN</i>	True Negatives
<i>TP</i>	True Positives

Chapter 1

Introduction

This chapter discusses about the report organisation, problem statements, motivations, objectives of the project, project scope and its contributions to the field.

1.1 Problem Statement and Motivation

Flash floods have been a serious threat in Malaysia recently, especially in urban regions. The previous floods were horrible; many were stuck in flooded houses waiting for help to come, unsure of where to go, and some centres were so full that there wasn't room for everyone. The problem statements for this project are:

i. No timely information to monitor flood risk levels and status.

The public needs more accurate and real-time information on water levels, rainfall, and weather forecasts. Flood information is crucial and must be widely disseminated to every community in time to assist people in preparing for flash floods. Early flood warnings enable people who live in flood-prone regions to be informed and evacuate, taking their belongings to the relief centre before the flood arrives. Therefore, the motivation behind this problem statement is to emphasise the importance of flood monitoring to reduce the devastating impact of floods.

ii. Limited early warning system to provide instant flood alerts.

There may be limited early warning systems in place to inform residents about future flash floods in certain areas. These systems often rely on traditional techniques such as sirens or public announcements, which are occasionally difficult to hear by everyone, especially in isolated or densely populated areas. Inefficient preparedness can cause chaos during flood events, making it critical to empower communities with the information needed to protect themselves and their property. Hence, the motivation of this problem statement is to highlight the urgent need for an early alert system that can notify the public to improve public preparedness.

iii. Difficult to manage flood events and coordinate resources efficiently.

Existing flood alert systems need to be improved in their ability to provide emergency measures and location-specific information to the public. In the absence of such information, people cannot immediately access emergency tips as they need to search the Internet by themselves. Besides that, people struggle to understand the precise locations of flood-affected area. Thus, ineffective resource allocation and evacuation to the relief centre can result in displacement or fatalities. For these reasons, the motivation behind this problem statement is to focus on emergency tips and geospatial visualisation to enhance community safety and response efforts during floods.

Based on the current state of affairs, it is imperative to employ digital technology innovation to tackle these unresolved issues. As a result, developing a timely flood monitoring and alert system is a critical, practical, and low-cost solution to reduce flood damage and address these issues.

1.2 Project Objectives

This project's main goal is to implement a flood monitoring and alert system to monitor and notify people of potential floods, mitigating the effects of floods on communities and infrastructure. Several sub-objectives that derived from the primary objective are:

i. To provide accurate and real-time information to monitor flood risk levels and status.

This sub-objective addresses the fundamental aspect of monitoring water levels, rainfall, and weather forecasts, which is critical for an effective flood monitoring and alert system. The flood warning and weather APIs would be used to provide information on water levels, rainfall, and weather in different regions of Malaysia. The innovation behind this system lies in the integration of APIs that leverage advanced sensor data and meteorological information to provide accurate and real-time updates on flood risk levels and current status.

ii. To develop a mobile application that can effectively disseminate alerts on possible flood events.

This sub-objective focuses on building a user-friendly platform that not only monitors but also notifies the public about potential flood incidents using push notifications. The innovation is in the real-time nature of the alerts and the user-centric approach to whether the system is able to deliver timely information and alerts to individuals or communities based on their points of interest.

iii. To develop a mobile application that can provide flood safety tips and geospatial visualisation using geospatial maps.

This sub-objective focuses on the use of geospatial data to present the locations of potential flood area, ensuring that resources are strategically allocated to assist affected communities. The innovation is using geospatial analytics and mapping to plan and make decisions about flood evacuation. As a result, communities may be evacuated more quickly and effectively during flood occurrences, improving the efficiency of relief efforts.

iv. To develop a mobile application that can detect flood based on rainfalls using deep learning techniques.

This sub-objective is concerned with the technical challenge of accurately predicting flood from rainfall data using deep learning techniques such as Artificial Neural Network (ANN). This feature enhances predictive modelling for more precise flood forecasts, enabling proactive measures to be taken. In this case, the innovation is in developing a feature that can reliably detect flood from rainfall data by integrating deep learning techniques.

1.3 Project Scope and Direction

The project scope involves the development of a comprehensive and innovative solution for flood monitoring and timely alert dissemination. The project aims to create a mobile application for flood monitoring and alert systems to contribute to better flood management and disaster response in Malaysia. This project will also integrate data from various sources, including current water level data, rainfall data, weather forecasts, and weather warning data. This mobile application will consist of four major modules: the monitoring module, the alerting module, the map module, and the flood management module.

For the monitoring module, users can monitor real-time updates on current water levels, rainfall intensity, weather forecast, and weather warning based on data obtained from the flood warning and weather APIs. The users will need to grant the system permission to allow it to access their locations. The application also enables users to set up points of interest and view detailed information for a specific monitoring station. Furthermore, the alerting module is crucial for sending push notifications and warnings about potential risks or rising water levels. Users are required to grant notification permission to the system to receive alerts for their points of interest.

Moreover, the map module includes geospatial visualisation, which provides a clear and intuitive representation of flood data on geospatial map. The application automatically displays the appropriate flood map based on user's device location, which includes high-risk areas and nearby flood locations reported by other users. Users can also search for the intended location they want to monitor, such as their hometown, workplaces, or schools. Last but not least, the flood management module provides flood safety tips, which include guidelines, emergency contact information, and access to resources or services offered by the government or relief organisations. For instance, users can get detailed information on flood prevention and emergency response strategies. The application also allows users to report any flood incidents, which enables real-time communication and information sharing with other users.

1.4 Impact, Significance and Contributions

The flood monitoring and alert system holds immense potential to benefit communities and authorities in various ways. Firstly, it makes the dissemination of flood information more accessible to the general public. Nowadays, smartphones are almost everywhere. Compared to a website, a mobile application would reach a wider audience, especially in regions where access to computers and stable internet connections may be limited. Therefore, this application addresses accessibility issues by providing a user-friendly interface to deliver real-time updates and alerts about flood situations, ensuring that users are instantly informed about the latest status.

Secondly, the flood monitoring and alert system provides early warnings to communities, which may save lives and lessen property damage. Floods often occur suddenly, making it nearly impossible for the general public to prepare for the flood event effectively. Thus, this system can be implemented to ensure that responsible authorities can work together and make decisions more efficiently during the flood management process, especially in the flood preparedness phase. For instance, early warning of a flood event can help the public and authorities organise both short-term and long-term preventive actions and prepare for rescue and evacuation efforts.

Thirdly, the flood monitoring and alert system can contribute to the flood response phase in the flood management process by using a geospatial map that makes it easier for the general public to comprehend flood risks and affected areas. The main considerations in most flood administrations are the geographical locations of affected areas and their respective severity levels. The system can help optimise resource allocation by visually representing the location of flood-affected areas, enabling emergency responders to allocate personnel and equipment to the affected regions efficiently.

1.5 Report Organisation

This final year report contains seven chapters, and the details of this project are shown in the following chapters. The report started with Chapter 1, which discusses the project's problem statements, objectives, scope, and contributions. Chapter 2 reviews some existing flood monitoring and alert systems, including a table of comparison and critical remarks for each application. Next, a few technologies used in this project are also reviewed. Furthermore, the requirements for the system, the methodology, and the proposed approach are described in Chapter 3, along with the architectural diagram, use case diagrams, use case descriptions, and activity diagrams. Besides that, Chapter 4 contains the system block diagram, database design, and user interface mock-ups. Chapter 5 covers the system implementation, including hardware and software setup, setting and configuration before development, and system operation. Then, this chapter also explains the problematic issues and challenges in the implementation. Moreover, Chapter 6 involves the system evaluation and testing results from the prototype, which will be discussed to address any limitations and suggest further study. Lastly, Chapter 7 concludes with an overview of the project's accomplishments, implications for the future, and possible areas for improvement.

Chapter 2

Literature Review

2.1 Review of Existing Flood Monitoring and Alert System

2.1.1 MyPublicInfoBanjir

MyPublicInfoBanjir is a system developed by the Department of Irrigation and Drainage (DID) to enable alert notifications for the public, DID internal users, and other agencies on flood forecasts and warnings. Besides that, the system can be utilised to facilitate travel arrangements and monitor the most recent flooding conditions during the monsoon season. As shown in Figure 2.1.1, users can also access relevant data, including current river water levels, current rainfall, and other flood information in 1,500 areas across the nation by placing hydrological telemetry stations, which frequently update the system every 15 minutes [1]. The Home page displays the latest water level and rainfall alerts for the selected points of interest. Furthermore, users can choose to view a graph or detailed information about a particular water level station.

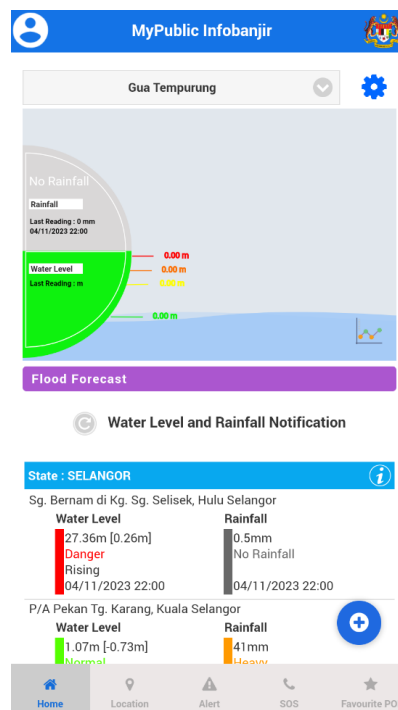


Figure 2.1.1: Home Page of MyPublicInfoBanjir

Users can configure Point of Interest (POI) through the system and enable automatic alert notification for the flood forecast location (Figure 2.1.2). If the water level rises or falls, an alert or warning will be signalled at that specific location.

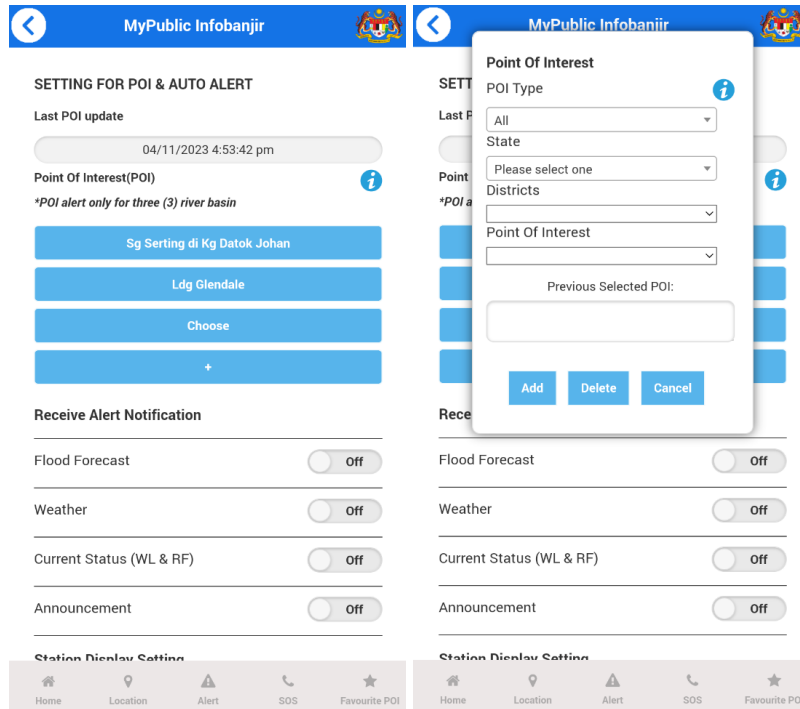


Figure 2.1.2: Setting for POI & Auto Alert

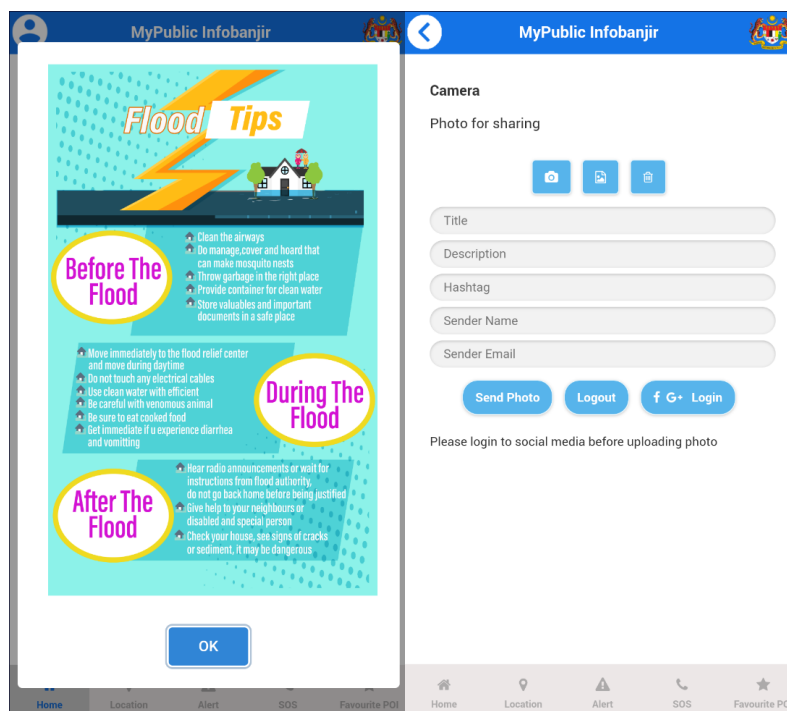


Figure 2.1.3: Additional Features of MyPublicInfoBanjir

Figure 2.1.3 demonstrates the additional features of MyPublic InfoBanjir. Users can view emergency advice and flood preparations before, during, and after flood events. The users can also upload images of recent floods and other disasters for information sharing. On the alert page, users can receive push notifications for announcements, weather updates, and flood forecasts. Apart from that, there is an SOS page that includes the contact information of various district offices for civil defence across the country [1]. The alert page and SOS page are shown in Figure 2.1.4.

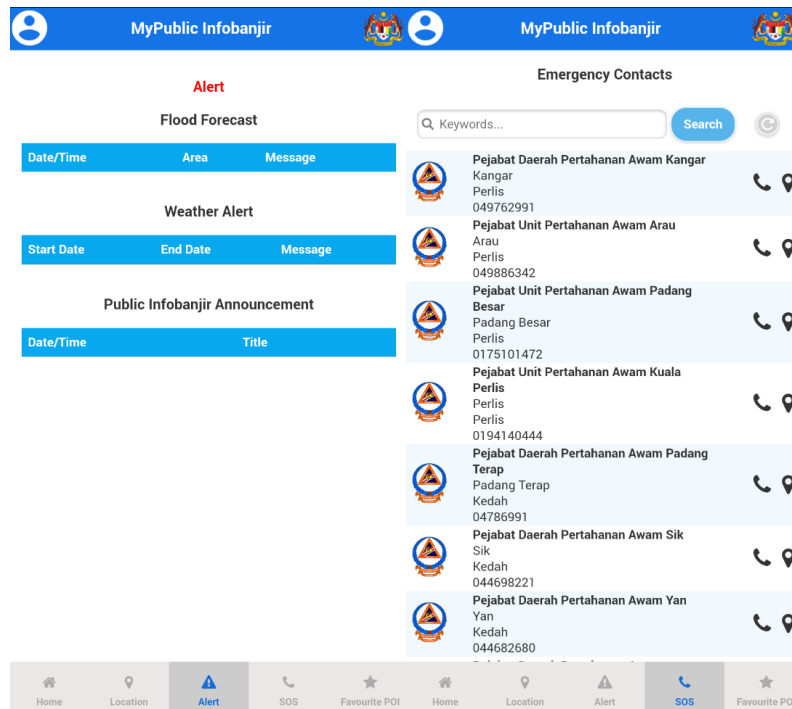


Figure 2.1.4: Alert Page and SOS Page of MyPublicInfoBanjir

Based on the system, one of its strengths is its use of diverse data, such as meteorological, hydrological, and geospatial data. Integrating data from multiple sources provides a comprehensive view of the flood scenario, facilitating more precise forecasts and improved decision-making. For example, it allows the DID, external agencies, and the general public to better plan for possible risks, allocate resources, and coordinate response efforts. Moreover, another strength of the system is that it allows users to contribute the latest flood images as crowdsourced data. This feature promotes community involvement in flood monitoring and response, which improves situational awareness and cultivates a sense of shared responsibility for disaster management.

However, the system also has several weaknesses. Based on the latest review in October 2023, one of the weaknesses is that it only supports older versions of the Android OS and cannot accommodate versions like Android 11 and above. Therefore, the system excludes a significant portion of the potential user base because it only works with previous Android versions. As new Android versions are released, many users upgrade their smartphones, which may cause their devices to become incompatible with the system. This clearly shows how more and more Android users, who are frequently the intended audience of such important public safety alerts, are unable to use the service.

Next, another area for improvement in this system is the setting for POI, which has been reported by several users. Setting POIs is essential because it allows users to customise their flood-related alerts to their requirements and preferences. For instance, these may include their residences, workplaces, schools, or travel locations. If the system has issues adding certain POIs, users cannot receive warnings in the places where they are most needed during an emergency. This can result in people missing flood alerts and not reacting quickly enough to potential disasters. Hence, users who encounter such difficulties in configuring POIs may get frustrated with the system. When people become less satisfied with the design, they may decide not to use it in the future. One of the recommendations for this problem is to perform user testing, identify the underlying causes of the issues, and implement software updates and enhancements.

2.1.2 BWDB Flood App

The BWDB Flood App is a mobile application that was developed by the Flood Forecasting and Warning Centre (FFWC), Bangladesh Water Development Board (BWDB) to enhance public accessibility to flood warning information. With the increasing rates of mobile phone adoption in South Asia, the application aims to disseminate information and alerts to a broader audience during the flood season [2]. The application dashboard is demonstrated in Figure 2.1.5, which consists of eight different components. There are two water level options, namely current water level and water level forecast. The current water level interface contains historical water levels, whereas the water level forecast interface comprises historical and forecast water information for each station. As shown in Figure 2.1.6, users can choose to access the water level information in a map or list view. If the user clicks on any station on the map, the details view will appear with various information such as river name, division, district, water level, highest level, and danger level (Figure 2.1.7). To avoid constantly scrolling up and down, users can create a favourites list from the total stations list based on their needs or the catchment region to observe within a short list view.

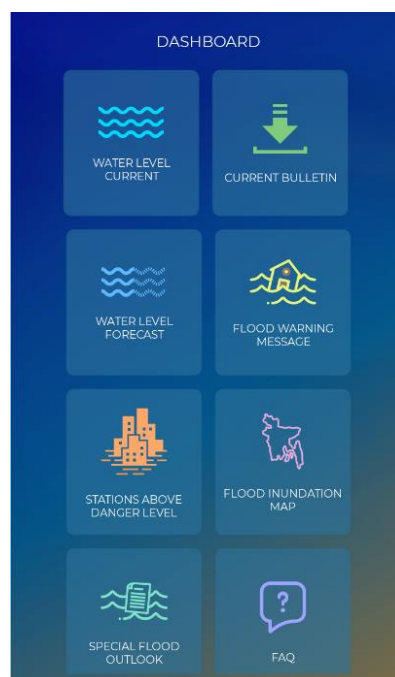


Figure 2.1.5: Dashboard of BWDB Flood App

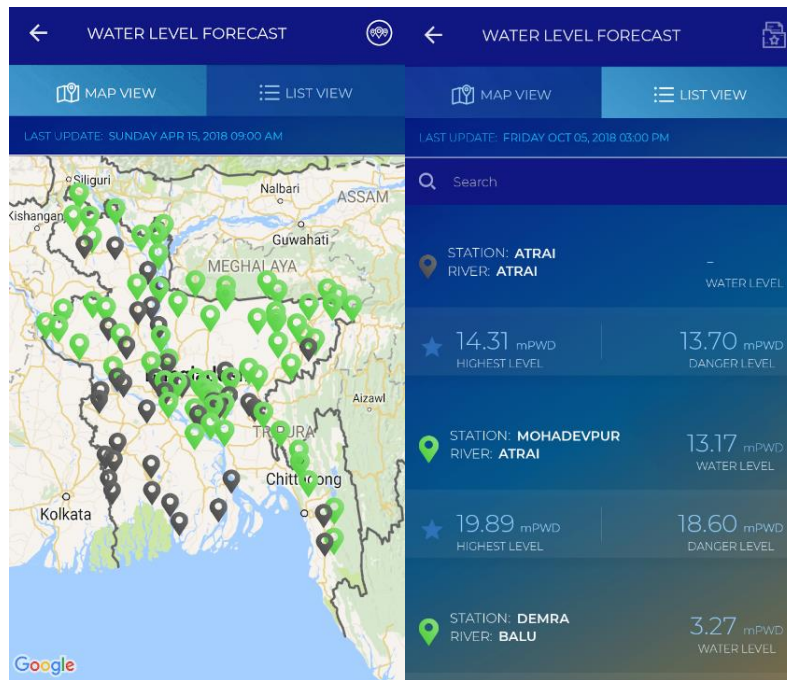


Figure 2.1.6: Water Level Forecast Page of BWDB Flood App

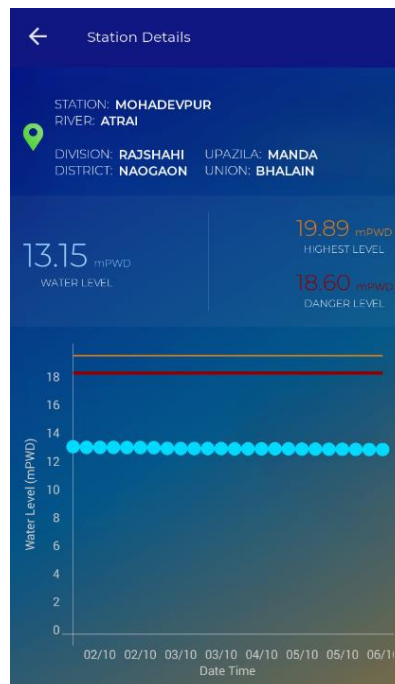


Figure 2.1.7: Station Details Page of BWDB Flood App

Figure 2.1.8 illustrates the current bulletin page and flood warning message page. Users can download information such as the flood summary and the observed morning and afternoon bulletins from the current bulletin page. Users can also access the most recent flood alert messages in the flood warning message page.

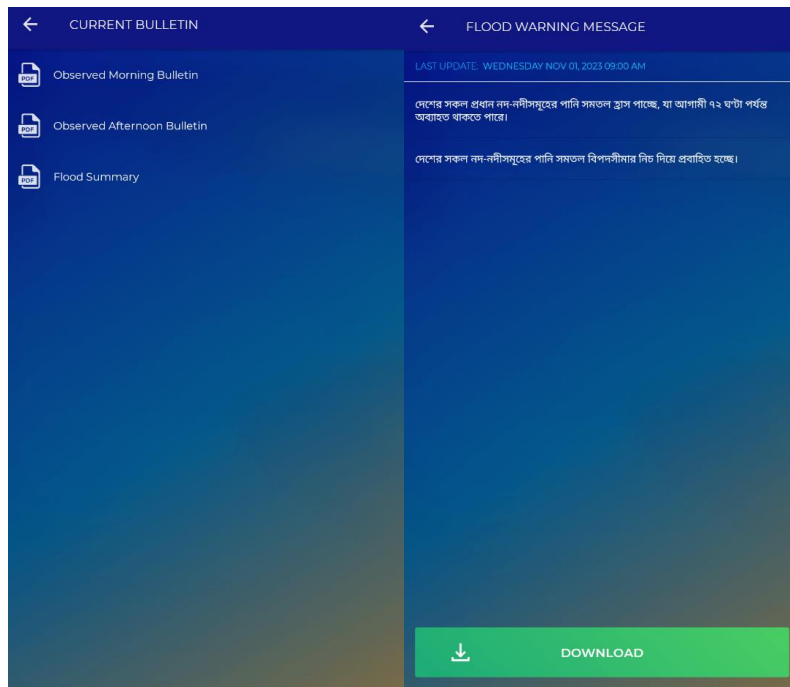


Figure 2.1.8: Current Bulletin Page and Flood Warning Message Page

Aside from that, users can view the stations above the danger level immediately, along with detailed information like station name, river name, danger level, rise or fall in centimetres, and river height above the danger level. Next, several views are available for the Flood Inundation Map, including Today, 24 Hour Forecast, 48 Hour Forecast, 72 Hour Forecast, 96 Hour Forecast, and 120 Hour Forecast (Figure 2.1.9).

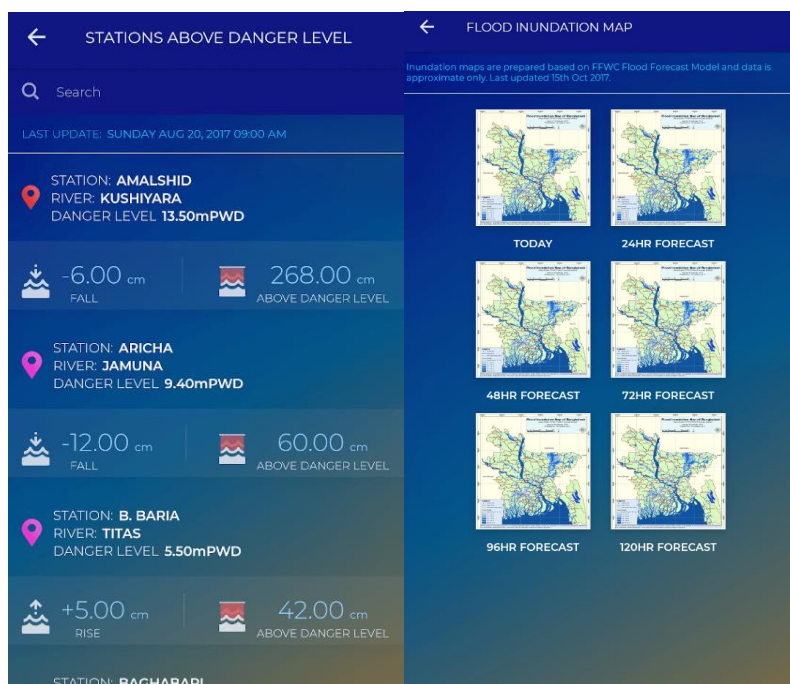


Figure 2.1.9: Additional Features of BWDB Flood App

In addition, a new feature called “Special Flood Outlook” enables users to receive special alerts regarding the state of the current floods. The FAQ page contains a few terminologies related to flood forecasting as well as an option for users to provide feedback and learn more about the application.

One of the strengths of this app is that it presents water level forecasts in multiple views, primarily through the map or list view. The map view feature in the BWDB Flood App allows users to visualise flood-related data on an interactive map. This feature is beneficial as it offers a geographical representation of predicted water levels and flood-prone places. For instance, users may easily understand the risk in specific locations by using color-coded markers on the map that indicate the severity of the flood scenario. In contrast, the list view provides an organised format of water level forecasts, which is helpful for users who prefer a structured presentation of information. Users can quickly scan the list to identify high-risk locations, prioritise their attention, and obtain detailed information as needed. Another strength of this app is its ease of use, where it is simple to learn and navigate. The app features a user-friendly interface with intuitive navigation and well-arranged menus. Thus, its uncomplicated design ensures that users can easily find the required information without confusion.

However, the application also has a few weaknesses. One of the weaknesses is the flood map representation, where users may find it challenging to view nearby stations. The flood map is significant for observing the water level stations and understanding the severity of flooding in a particular location. Therefore, the flood map representation can be adjusted to the device’s current location using the Global Positioning System (GPS), allowing the user to view adjacent points of interest or become instantly notified of flood events. Another improvement for this application is the development of a weather alert and forecast feature. To effectively prepare for and respond to flood disasters, users require flood information and weather forecasts. For example, weather conditions, including heavy rain, storms, or cyclones, are often the major causes of flooding incidents. With the help of timely weather alerts, people can take preventive actions, such as evacuating high-risk regions, reinforcing flood defences, or securing belongings.

2.1.3 FloodAlert

FloodAlert is a mobile application produced by SOBOS to provide current water levels and forecasts to users at approximately 30,000 water monitoring points worldwide [3]. Besides that, it quickly warns users of emergencies when the water level hits a dangerous level. As a result, users can take preventive action and respond early to severe events like floods. The gauge application also enables users to modify thresholds for various water levels of relevant water bodies in the USA and Europe. With a large number of measuring stations, it offers a comprehensive picture of the flood situation and has a precise impact on the accuracy of future water level forecasts. When users first launch the app, the application will seek permission to access the device's location. Users will then have to choose a location to monitor in order for the app to show stations nearby (Figure 2.1.10). The main page is shown in Figure 2.1.11, where users can add monitoring points, create notes, and monitor water levels from the map or selected stations. Furthermore, the application provides a catalogue of measures and a flood notebook, allowing users to learn from previous alerts and prepare for upcoming flood disasters. As demonstrated in Figure 2.1.12, the action catalogue helps users respond to the initial warning of a critical level by providing a step-by-step, officially approved guide with safety measures.

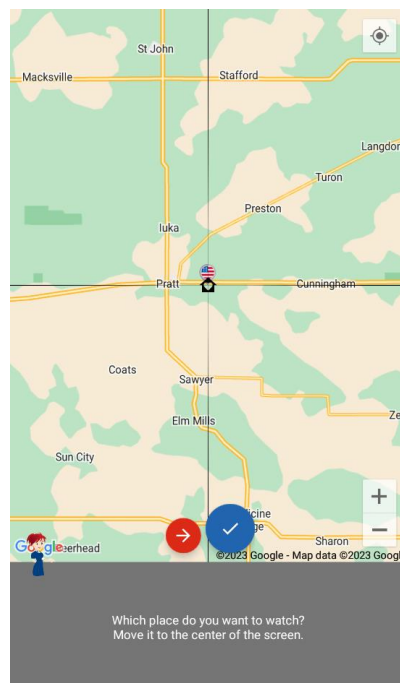


Figure 2.1.10: Setting for Monitoring Location

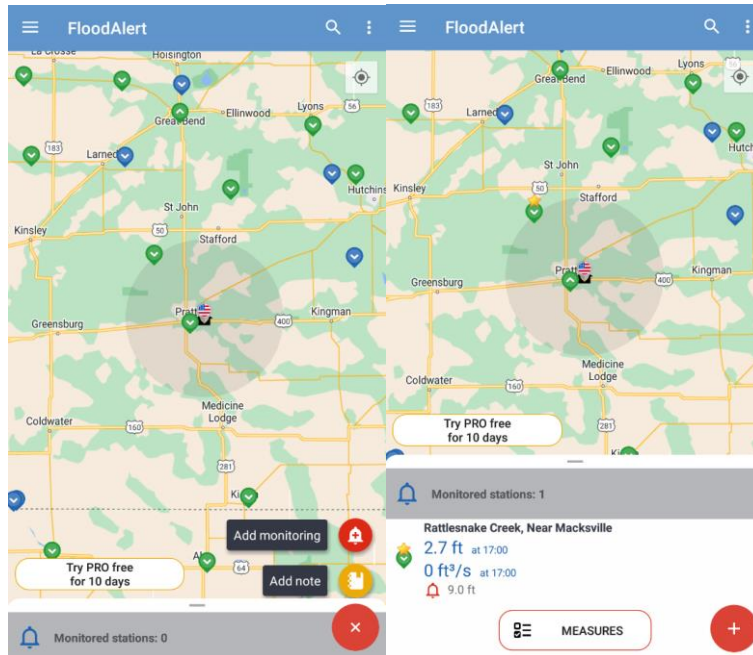


Figure 2.1.11: Main Page of FloodAlert

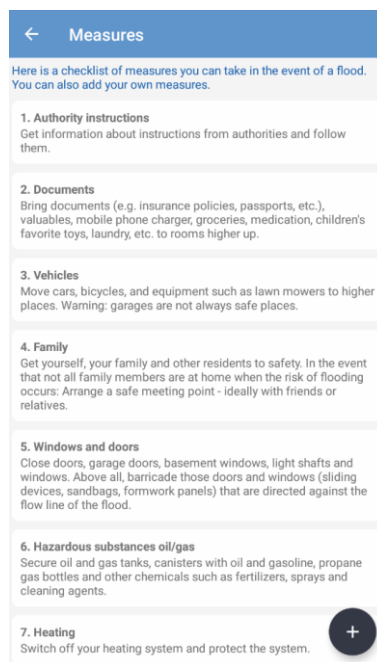


Figure 2.1.12: Measures Page of FloodAlert

Moreover, users can search among 30,000 stations for the appropriate gauges by entering the name of a town or body of water on the main page. The figure below illustrates the water level graph for a specific station. Users can drag the red slider to adjust the warning threshold. Then, the application will send real-time notifications to them when the water level rises above their personal warning limit.

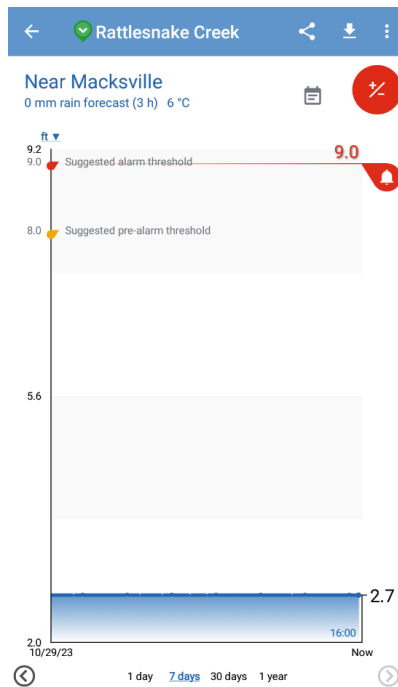


Figure 2.1.13: Water Level Graph of FloodAlert

As shown in Figure 2.1.14, users can customise their notifications by selecting different tones, vibrations, screen output, and LED flashing lights. Additionally, users may optimally enable the rain radar to assess the current weather conditions by observing the color-coded areas on the map that indicate heavy rain locations.

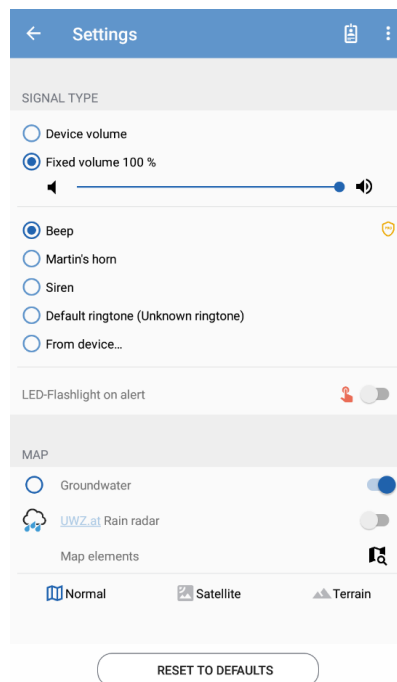


Figure 2.1.14: Settings for FloodAlert

One of its strengths is that the emergency alerts app allows users to simply configure one or more threshold values for warnings for each gauging station. By setting the warning threshold, an alarm signal is sent to their smartphone whenever the river or flood level rises above or falls below their personally defined threshold value. This feature enables users to respond quickly to urgent situations such as heavy rain and flood disasters. Another strength is that users can modify their warning signals. For example, users can choose their preferred alert signal to notify them of flood disasters and upcoming emergencies. Hence, users will be able to prepare for storms or rain-related disasters with the help of water gauge warnings in the application. Users can also choose to turn off the alarm sound for less urgent early warnings that do not require to be responded immediately.

However, the application has several weaknesses. One of the weaknesses is that critical functionalities may not be available to free users. Those who cannot afford or refuse to pay for the pro version may not have access to advanced features, such as customisable warnings, warning messages via SMS and email even without an internet connection, and access to historical water level data. This limitation might be problematic since having access to such data can be significant for personal safety and preparedness. Another area of improvement is the location search in the application, where users frequently need help searching nearby monitoring points. Although the application requests access to the device's location, it misplaces users in other locations instead of using their geolocation as the centre point. Thus, one of the recommendations is to implement more accurate and intuitive location search functionality that centres users based on their location.

2.1.4 My Flood Risk Accra

My Flood Risk Accra is a flood monitoring system that HKV developed to provide citizens with clear and detailed information on their exposure to flood risks [4]. The system contains several flood maps for various statistical return periods. On the home page, the water depth is presented as metres above mean street level and acts as an indicator of flooding. In addition, it recognises the user's current location, enabling them to monitor water levels and estimate the potential rise in floodwaters around their homes (Figure 2.1.15). For instance, users can make better decisions based on the water level, such as staying on an upper, dry floor in an emergency or leaving their homes. Users can also use the textbox above to enter their intended location or the icon above to obtain their current location. As demonstrated in Figure 2.1.16, the literacy information regarding measures that may be performed to prevent flooding is categorised based on the flood level. Users can select a location on the map and click on the star to save the location as a favourite location (Figure 2.1.17). Next, the system allows users to observe a specific water level category by integrating geolocation with the chosen risk class.

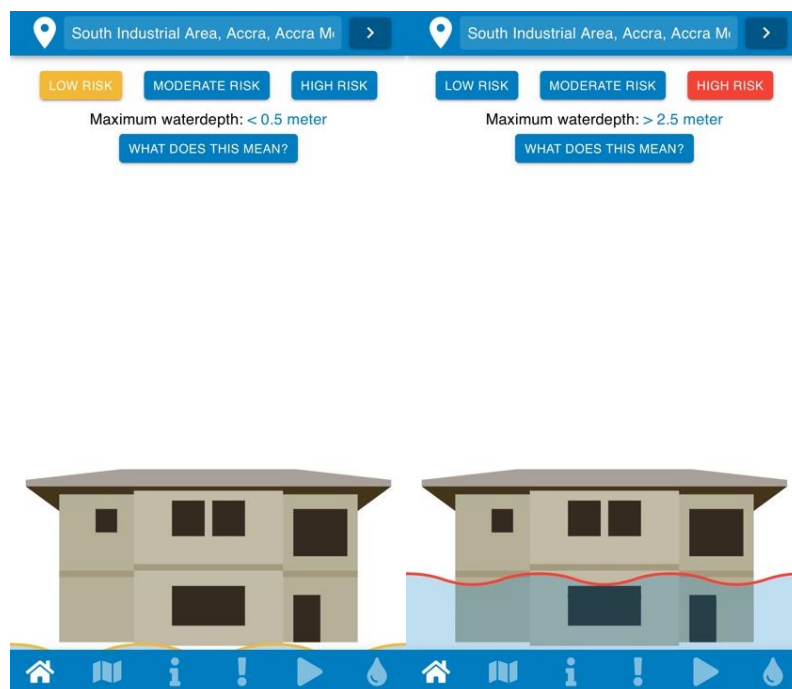


Figure 2.1.15: Home Page of My Flood Risk Accra

What does this mean?

The literacy work is categorized based on the flood intensity (low, moderate and high). This includes risk management - financial and non-financial (waste management, easy apply tips, building codes)

- Pre-flooding tips
- Common (Low risk)
- Moderate (Moderate risk)
- Extreme (High risk)
- Post-flooding tips
- NADMO Emergency numbers

COMMON (Low risk):

With this situation, there is a slight or minimal amount of water coverage. Limited or minimal damage is expected. Citizens are advised to be calm, since it's a common water level.

Precautions:

- Keep important documents waterproof, for instance in a plastic bag.
- Disconnect any electrical appliance; this could help prevent fires and electric shock.
- Move all valuable items to a higher floor.
- Pack a bag and move to a higher ground. The bag could contain toiletries, touch/flashlight, drinking water, some edibles.

OK

Figure 2.1.16: Literacy Information based on Flood Intensity

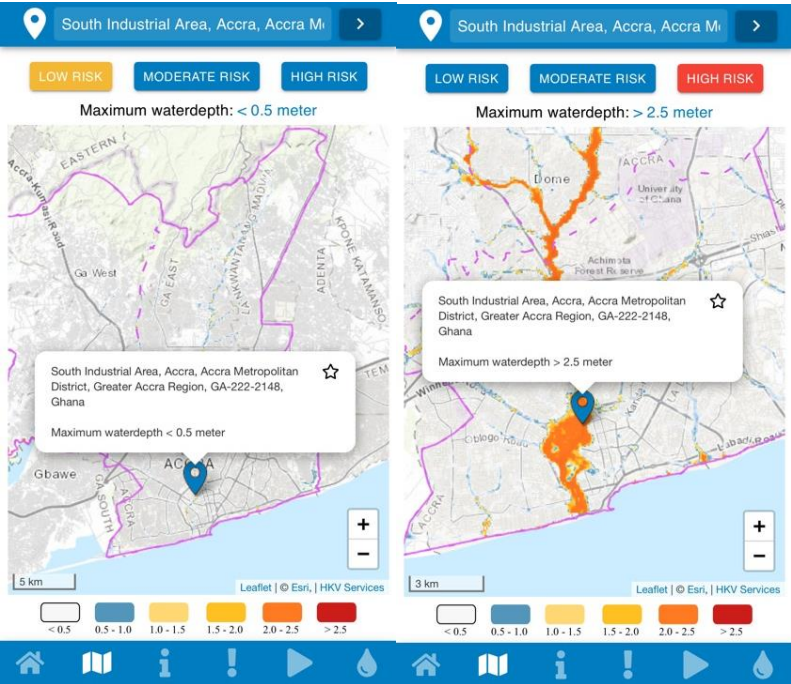


Figure 2.1.17: Flood Map of My Flood Risk Accra

Based on Figure 2.1.18, the system provides literacy information in the form of infographics on how to minimise the impacts of floods, including prevention, preparedness, emergency response, risk or transfer insurance, and recovery strategies. It also displays detailed tutorial instructions to guide users in navigating the application.

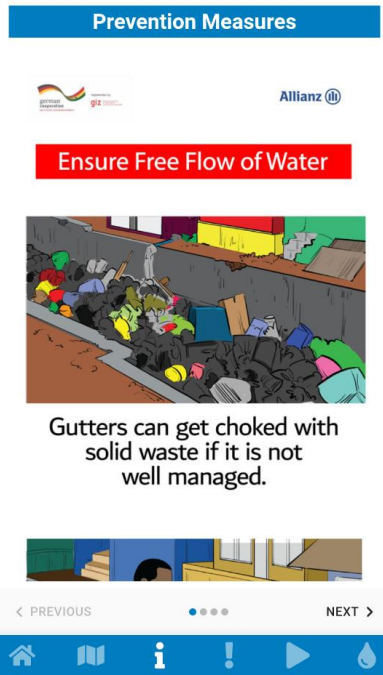


Figure 2.1.18 : Literacy Information in form of Infographics

Moreover, the system disseminates official forecasts from the Ghana Meteorological Agency (GMet) in the event of heavy rain. Therefore, users can check the weather forecast and the estimated rainfall amount for efficient planning and decision-making (Figure 2.1.19). It also allows users to give a specific rainfall event as input, which is an additional approach to get water depths for a certain place.

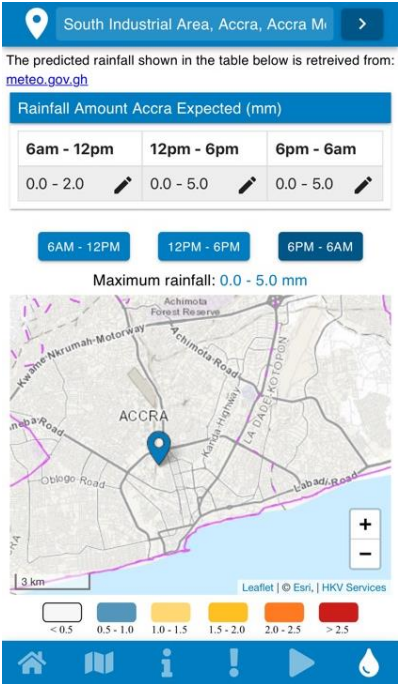


Figure 2.1.19: Rainfall Page of My Flood Risk Accra

The strength of this system is that it provides comprehensive emergency tips and flood management strategies that cover prevention, preparation, response, and recovery. Floods have the potential to be fatal, and having access to emergency tips is crucial for public safety. The reason is that it gives users guidance on what to do in the case of a flood and how to react after a flood event, which can help save lives and reduce the risk of injury. Apart from that, another notable strength of the system is its provision of a tutorial guide. The tutorial instructions are important for a flood disaster management system as they serve a variety of users, including those who may not be familiar with using such an application. Thus, the system needs to be more accessible to a broader audience so that people with varying age groups and technical backgrounds can understand and utilise it effectively.

However, the system also has a few weaknesses. One of the weaknesses is its limited applicability, as it is designed specifically for the flood risk scenarios in Accra. This demonstrates that individuals who live in or are concerned about flood risks in other cities within Ghana or adjacent countries will not benefit from this platform. By concentrating exclusively on Accra, the system loses the opportunity to reach a wider audience and assist other communities facing similar difficulties. Therefore, sharing knowledge and resources can result in more effective flood disaster management, and a narrowly focused system limits the potential for cross-regional collaboration and support. Another improvement of this system is the absence of monitoring points on the map, which forces users to search for locations to monitor on their own. This weakness can affect the usability and effectiveness of the platform since people unfamiliar with the geography of Accra would need help locating and choosing appropriate monitoring points. Hence, in an emergency, users may waste more time looking for relevant locations if monitoring points are not shown on the map. Lastly, human error in selecting monitoring points can also lead to inaccurate information, which may cause one to monitor the incorrect areas and miss early warnings when a flood is approaching.

2.2 Summary

Features	MyPublicInfoBanjir	BWDB Flood App	FloodAlert	My Flood Risk Accra
Water level monitoring	✓	✓	✓	✓
Rainfall monitoring	✓	✗	✓	✓
Flood forecast	✓	✓	✓	✓
Weather forecast	✓	✗	✓	✓
View graphs	✓	✓	✓	✗
Provide geographic map	✓	✓	✓	✓
Alerts/notifications	✓	✓	✓	✗
Provide flood safety tips	✓	✗	✓	✓
Provide emergency contacts	✓	✗	✗	✗
Setting point of interest	✓	✓	✓	✓
Flood information sharing	✓	✗	✗	✗
Download flood information	✗	✓	✓	✗
Create notes	✗	✗	✓	✗
Alert customisation	✓	✗	✓	✗

Table 2.2.1: Table of Comparison

Existing System	Strengths	Weaknesses
MyPublicInfoBanjir	<ul style="list-style-type: none"> • Use of diverse data sources • Allow users to upload and share flood images 	<ul style="list-style-type: none"> • Only support older versions of the Android OS • Difficulties in setting POI
BWDB Flood App	<ul style="list-style-type: none"> • Present water level forecasts in multiple views • Ease of use 	<ul style="list-style-type: none"> • Poor flood map representation • No weather alert and forecast
FloodAlert	<ul style="list-style-type: none"> • Configure threshold values for warnings • Customise warning signals 	<ul style="list-style-type: none"> • Critical functionalities may not be available to free users • Difficulties in location search
My Flood Risk Accra	<ul style="list-style-type: none"> • Provide comprehensive emergency tips and flood management strategies • Provide tutorial instructions 	<ul style="list-style-type: none"> • Limited applicability • No monitoring points on the map

Table 2.2.2: Summary of Existing System

2.3 Critical Remarks

This section highlights a few critical issues that were present in existing systems to identify the areas of improvement so that they can be improved in this project. The critical issues are:

i. The existing systems have limited applicability for monitoring floods and disseminating flood alerts.

The existing system has demonstrated its ability to monitor water levels and forecast flood events. However, applications such as MyPublicInfoBanjir are incompatible with the latest Android OS versions. The application should accommodate a broader range of versions to enhance its applicability so that more people can benefit from the safety features, potentially minimising risks during flood situations. Besides that, the My Flood Risk Accra app is only limited to a single state in Ghana. Therefore, the application should be adaptable to be used in different locations of the country by integrating a location-based service that can monitor floods in various regions. Based on the critical issue, this project focused on developing a mobile application that is applicable to users from all around Malaysia, regardless of their OS version or geographic location.

ii. The existing systems could be improved to present real-time flood situations on a geographic map.

The ability of a system to present flood situations on a map is crucial for providing timely and actionable information to users during flood events. Furthermore, the visual representation of flood situations makes it easier for users to understand and increases accessibility and user engagement. However, their poor flood map representation is a critical issue in the BWDB Flood App and My Flood Risk Accra. The BWDB Flood App did not have real-time adaptation to the user's location and had trouble identifying nearby monitoring stations. In contrast, My Flood Risk Accra did not have monitoring points on the map, which forced users to search manually. Hence, this project focused on integrating spatial data with other data and incorporating the Global Positioning System (GPS) to customise real-time flood information for users on the map.

iii. The existing systems have limited and poor designed features for monitoring floods and disseminating flood alerts.

The current system, such as the BWDB Flood App, needs essential weather-related information, which is significant for effectively forecasting and managing flood situations. In addition, some features in existing systems could be better designed, making it easier for users to use them. The MyPublicInfoBanjir app has issues setting specific locations that users want to monitor or receive alerts about, whereas the FloodAlert app has issues finding or identifying certain locations. Thus, this project focused on integrating a reliable weather forecast API that provides real-time weather updates. Apart from that, it also focused on enhancing user experience when setting POIs or conducting location searches. The project made critical functionalities available to all users, including water level monitoring, rainfall monitoring, and flood forecasts.

2.4 Review of Technology Used

2.4.1 Flutter

Flutter is a flexible mobile app SDK (software development kit) developed by Google to allow developers to construct high-fidelity and high-performing apps for Android and iOS [5]. Flutter uses a programming language called Dart, which is simple to learn and enables faster development than traditional approaches. One of Flutter's strengths is its ability to create applications for several platforms from a single codebase. This method drastically reduces development time and expenses while ensuring consistency in the user experience across various devices. Besides that, Flutter's hot reload feature promotes an agile development environment by allowing instant previews and real-time code changes without the need to restart the application. Flutter's enhanced rendering engine also ensures better performance, such as smooth animations, seamless navigation, and swift loading times. It gives users a native-like experience because the Flutter apps are compiled into native code. Lastly, developers can create visually attractive and customisable user interfaces with Flutter using its vast selection of widgets and flexible design framework.

2.4.2 Google Firebase

Google Firebase is a collection of cloud-based development tools that assist mobile app developers in creating scalable, feature-rich, and robust apps [6]. One of the features provided by Firebase is its authentication services, which include reliable identity verification options like Facebook, Google, and email logins. Therefore, it provides users with a secure and trustworthy environment in which to sign into the application. In addition, Firebase's real-time database and cloud storage features offer a consistent and scalable solution for storing and syncing app data in real-time across all of the users' devices. Apart from that, the Firebase Cloud Messaging (FCM) service simplifies the implementation of push notifications, enabling personalised and targeted communication with app users. This functionality enables developers to update app content and deliver timely or relevant messages to users. Firebase's analytics and performance monitoring tools also provide insights into user behaviour, engagement, and app performance. For instance, the developers can use performance monitoring to monitor metrics such as CPU usage, memory usage, and network traffic.

2.4.3 TensorFlow Lite

TensorFlow Lite is a cross-platform, open-source deep learning framework released by Google for inference on mobile devices. It is designed to work with a variety of platforms, such as microcontrollers, embedded Linux, Android and iOS smartphones. Pre-trained TensorFlow models can be transformed into unique formats that are optimised for storage or speed [7]. The primary strength of TensorFlow Lite is its ability to integrate machine learning models seamlessly into mobile applications. It reduces dependency on server-side processing by allowing on-device machine learning model inference with a tiny binary scale and fast startup. Aside from that, since there is no requirement for data to be transferred back and forth from the server, the optimised architecture of TensorFlow Lite results in low latency and high performance when executing machine learning models on mobile devices. Moreover, the framework also enhances data security by performing computations locally rather than sending data to external servers. This capability ensures that any sensitive personal data remains on the user's device and secures user privacy.

Chapter 3

System Methodology/Approach

This chapter covers the project's methodology, system requirements, and system design diagram. The processes of the project were divided into several phases, including project initiation, planning, sprint execution, review and demo, release, and feedback.

3.1 Methodology

The methodology used in the development of this project is Agile. Agile methodology is an incremental and iterative approach to project management and software development that emphasise on fast delivery, customer feedback, and continuous improvement. The method itself is centred on the growth and learning processes of its members [8]. Smaller iterations, or sprints, are used to divide the development, which promotes flexibility and adaptability. Agile is suitable for projects with changing or unclear requirements since it enables modifications and adjustments to project requirements even late in the development process. Therefore, agile methodology is appropriate for developing the flood monitoring and alert system because it must adjust to changing environmental conditions and technological considerations. The flexibility of Agile allows quick modifications to flood monitoring parameters and data sources.

Moreover, the Agile methodology was chosen due to the short software development time. Agile places a higher priority on delivering the most valuable features early in the project, providing faster regions of interest, and addressing critical needs sooner. Besides that, the flood monitoring and alert system is considered a complex project because it involves diverse data sources, emergency alerts, and geospatial visualisation. Hence, Agile is suitable for this project since it divides the project into smaller, more manageable iterations, each of which produces a potential shippable product increment and can help reduce complexity. As shown in Figure 3.1.1, the typical phases of agile software development are project initiation, planning, sprint execution, review and demo, release, and feedback.

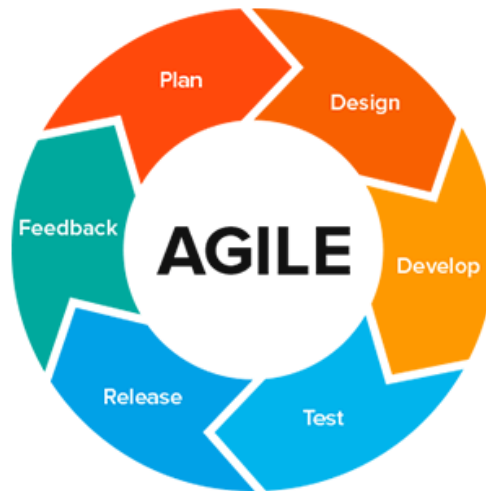


Figure 3.1.1: Agile Software Development [8]

Firstly, the project scope, objectives, and high-level requirements will all be specified at the project initiation phase. After that, an initial backlog of tasks will be created, including data collection, system requirements, system design, and system implementation. Next, planning sessions will be conducted to prioritise tasks in the backlog based on their significance for the project's success and any potential considerations. The planning phase will also determine the goals for the project's initial sprint and the amount of work needed to complete each task. During the sprint, we will begin designing and developing the system based on the tasks listed in the sprint backlog. Regular meetings with the supervisor will also be conducted for this project to discuss progress, direction, and difficulties. At the end of each sprint, the work completed will be tested and reviewed, which includes a system prototype to be demonstrated. Furthermore, the system component will be released, and feedback will be gathered from the supervisor to determine areas for improvement. Lastly, this cycle of planning, execution, and review is repeated for subsequent sprints in order to deliver a complete final prototype.

3.2 System Requirements

3.2.1 Functional Requirements

i. Authentication Module

- The system should allow users to create accounts with unique email addresses and passwords.
- The system should enable users to sign in with their registered accounts.
- The system should enable users to reset their passwords with their registered email addresses.
- The system should validate and ensure the uniqueness of email addresses.

ii. Monitoring Module

- The system should allow users to monitor water levels, rainfalls, and flood risk levels of monitoring stations.
- The system should allow users to set up their points of interest for monitoring.
- The system should allow users to monitor weather forecasts and live weather warnings.
- The system should allow users to view flood-related information reported by other users.
- The system should allow users to view graphs or charts for better comprehension.
- The system should allow users to select or enter rainfall data to detect the possibilities of flood.

iii. Map Module

- The system should request permission to access user's current location.
- The system should enable users to see their current location.
- The system should enable users to view nearby flood affected areas.
- The system should enable users to view current state of monitoring stations.
- The system should enable users to search for other locations to monitor.

iv. Alerting Module

- The system should request permission to send alerts when predefined water level thresholds of their POIs are crossed.
- The system should allow users to review alerts about potential risks or rising water levels.

v. Flood Management Module

- The system should enable users to read flood safety tips.
- The system should enable users to report any flood-related incidents.

3.2.2 Non-functional Requirements

i. Operational Requirements

- The system should work in Android environment and compatible with latest versions of Android.
- The system should be available all the time, ensuring continuous monitoring and alerting capabilities.
- The system should be able to import JPEG or PNG graphic files.

ii. Performance Requirements

- The system should respond to user inputs within 2 second.
- The database must be updated in real time.

iii. Security Requirements

- The system should only allow registered users to sign in and access to the features.
- New users are required to sign up an account before they can sign into the system.

3.3 System Design Diagram

3.3.1 System Architecture Diagram

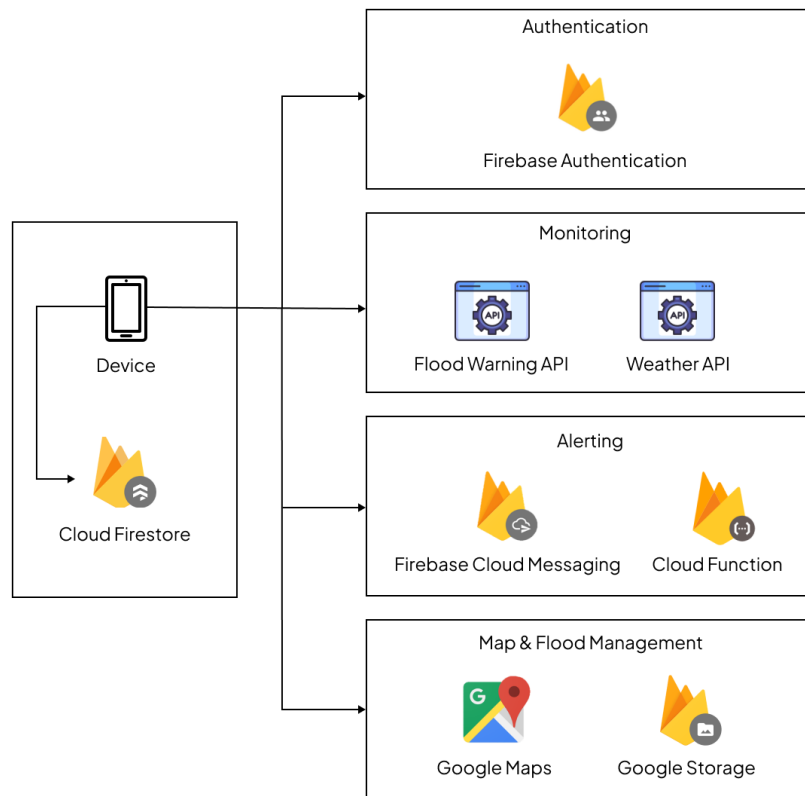


Figure 3.3.1: System Architecture Diagram

Figure 3.3.1 illustrates the architecture diagram of the proposed application. The proposed architecture for the flood monitoring and alert system includes several parts: the device, the Cloud Firestore, an authentication module, a monitoring module, a map and flood management module, and an alerting module. The device serves as the front-end where users interact with the system. It communicates with the Cloud Firestore to store and retrieve user information, points of interest, alerts and flood reports.

The authentication module is responsible for user authentication and authorization within the mobile application. It makes use of Firebase Authentication, which is provided by Google’s Firebase platform, to simplify the user authentication process by offering a variety of sign-in options, such as email or password, Google account, and social media logins. Users must authenticate themselves through the module before they can access the application and its features. It also manages the user registration and password reset processes.

By integrating the Flood Warning API and Weather API, the monitoring module allows users to monitor water levels, rainfall, weather forecasts, and flood conditions through the external real-time APIs. The Flood Warning API provides up-to-date information on water levels and flood alerts that is updated every 15 minutes, enabling users to stay informed about potential flood risks in specific locations. Simultaneously, the Weather API delivers real-time weather warning and data, including rainfall predictions and atmospheric conditions.

The alerting module uses Firebase Cloud Messaging (FCM) and Cloud Function to send users alerts based on the status of their points of interest. It is a robust cloud solution that delivers real-time messages via push notifications to targeted devices. These notifications promptly inform users about the specific nature of the alert, such as arriving floods or specific instructions, enabling them to take immediate action. The seamless integration of FCM into the alerting module ensures that users are informed in real-time, enhancing the effectiveness of the system by providing timely and actionable information to mitigate potential risks.

The map and flood management module utilises Google Maps to provide users with flood-related information on a map and nearby affected regions based on their current location. The Google Maps API is used to visualise real-time flood data and reports, providing users with a geographical representation of flood-prone areas and their severity. For instance, the Places API, Maps SDK for Android, and Geocoding API are used with Google Maps. Concurrently, the Firebase Storage complements the information for flood management by storing the images of flood-related incidents reported by the users.

3.3.2 Use Case Diagram and Description

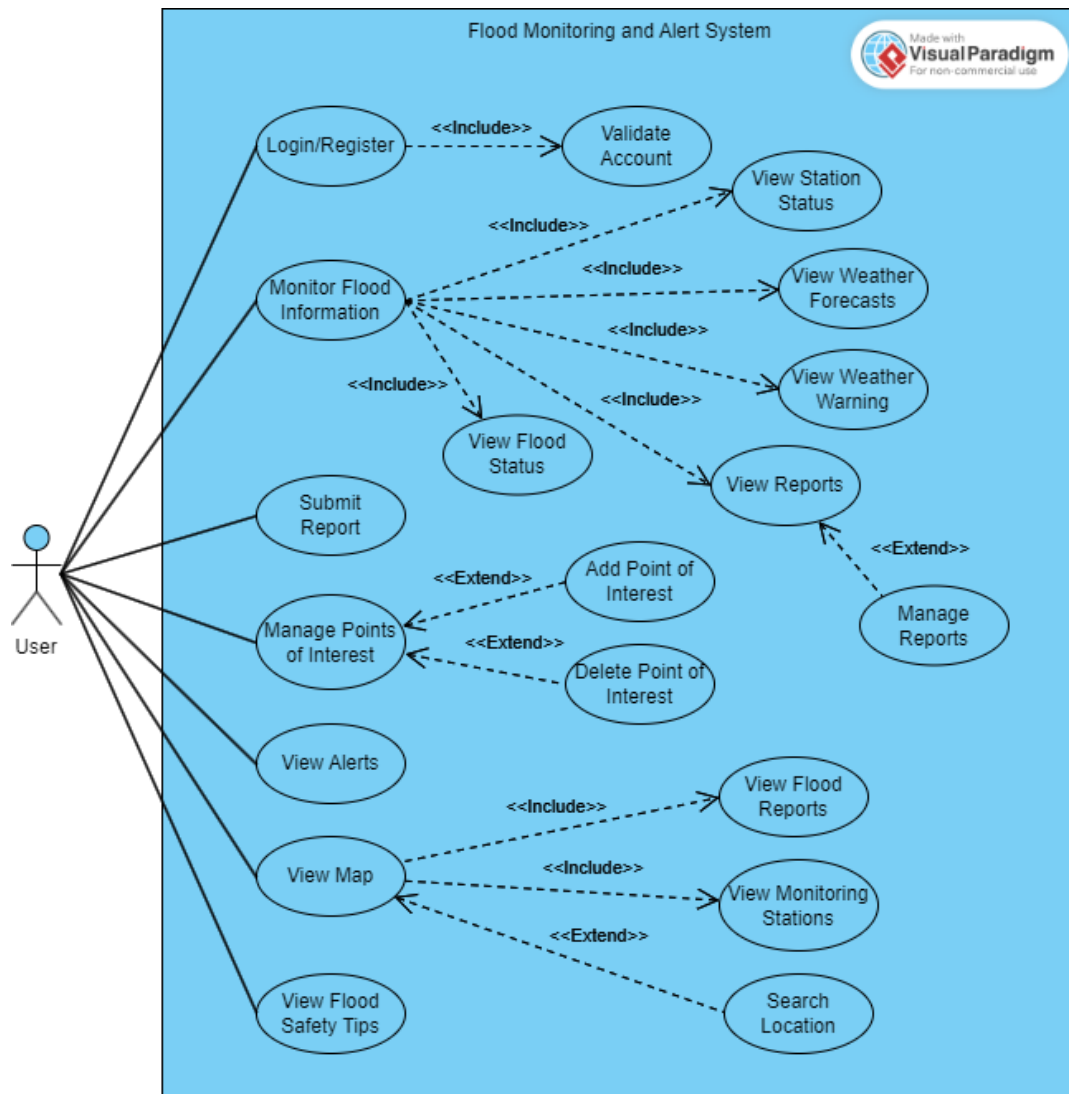


Figure 3.3.2: Use Case Diagram of the Proposed System

Figure 3.3.2 has demonstrated the use case diagram for the flood monitoring and alert system, which includes various functionalities to facilitate a comprehensive user experience. Firstly, users can access the system through the “Login/Register” use case, which involves a verification process for account security. The “Monitor Flood Information” use case serves as the main feature, which branches out to provide the ability to view station status, view weather forecasts, view weather warnings, view reports, and view flood status. The “View Reports” use case further extends to include a feature that enables users to manage their reports. Next, managing points of interest involves two extended functionalities: adding and deleting points of interest. The “View Alerts” use case allows users to catch up on missed notifications by providing them with a way to access and review alerts associated with their POIs.

The “View Map” use case offers essential geographic context by integrating the display of monitoring stations and flood-affected locations through include relationships. Furthermore, the “View Map” extends to include a search location feature for enhanced navigation. The “View Flood Safety Tips” use case enables users to view guidelines and recommendations intended to minimise the risks associated with flooding and respond appropriately during flood events. Lastly, the “Submit Report” use case allows users to report any flood-related incidents. Therefore, the use case diagram presents a full overview of a robust flood monitoring and alert system that includes features such as user authentication, comprehensive flood data monitoring, efficient points of interest management, timely alerts, and an interactive map.

Use Case Name: Login/Register	ID: <u>1</u>	Importance Level: <u>High</u>
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interest: User – wants to login or register an account.		
Brief Description: This use case describes the process of user authentication and registration. Users input their credentials to login or create a new account.		
Trigger: User accesses the system or attempts to perform a secured action. Type: External		
Relationships: Association: User Include: Validate Account Extend: - Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system displays the login page. 2. If the User has an existing account, they input their email address and password and click the login button. 3. If the User does not have an account, they click the register button on the login page. 4. The system displays the register page, which requests a username, email address, and password. 5. The User enters the required information in the form and clicks the register button. 6. If the User forgot their password, they click the forgot password button on the login page. 7. The system displays the reset password page. 8. The User inputs their email address and clicks the reset password button. 9. If the inputs are correct, the system validates the credentials and displays the home page. 		
SubFlows: Not applicable		
Alternate/Exceptional Flows: <ol style="list-style-type: none"> 9a. If the User enters incorrect inputs, the system displays an error message and prompts the User to enter again. 		

Table 3.3.1: Login/Register Use Case Description

Use Case Name: Monitor Flood Information	ID: <u>2</u>	Importance Level: <u>High</u>
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interest: User – wants to monitor flood-related information.		
Brief Description: This use case involves monitoring various aspects of flood-related information, including water levels, rainfall, flood risk levels, weather forecasts, weather warning, and user-generated reports.		
Trigger: User accesses the home page. Type: External		
Relationships: Association: User Include: View Station Status, View Weather Forecasts, View Weather Warning, View Reports, View Flood Status Extend: - Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system displays the dashboard, which includes six options. 2. If the User clicks on monitoring option, the system displays a list of locations with their corresponding water levels and rainfall. 3. If the User clicks on point of interest option, the system displays points of interest with their flood risk levels and current status. 4. If the User clicks on weather forecasting option, the system displays current weather conditions with their corresponding weather forecasts. 5. If the User clicks on weather warning option, the system displays a list of warnings. 6. If the User clicks on flood reporting option, the system displays flood information reported by other users. 7. If the User clicks on flood status option, the system displays a detailed analysis using visualisations such as graphs and charts. 8. The User navigates between different options to monitor flood information. 		
SubFlows: Not applicable		
Alternate/Exceptional Flows: Not applicable		

Table 3.3.2: Monitor Flood Information Use Case Description

Use Case Name: View Reports	ID: <u>3</u>	Importance Level: <u>High</u>
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interest: User – wants to view flood reports.		
Brief Description: This use case focuses on providing users with information on flood in specific areas reported by other users.		
Trigger: User accesses the “Flood Reporting” section. Type: External		
Relationships: Association: - Include: - Extend: Manage Reports Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system retrieves data and displays a list of reports with their corresponding date, description, image, location, and flood hazard level. 2. If the User wants to validate a report, they click the valid button to confirm the accuracy of the information provided. 3. The system updates the validation count of the report in the database. 4. If the User wants to share a report, they click the share button and select the desired platform to share. 5. The system navigates the User to the selected platform along with flood information. 6. If the User clicks on manage my report option Execute the Manage Reports use case. 		
SubFlows: Not applicable		
Alternate/Exceptional Flows: <ol style="list-style-type: none"> 1a. If the system is unable to retrieve report data or no report available, it displays a message to notify the User. 		

Table 3.3.3: View Reports Use Case Description

Use Case Name: Manage Points of Interest	ID: <u>4</u>	Importance Level: <u>Moderate</u>
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interest: User – wants to manage their points of interest.		
Brief Description: This use case describes how users can add or delete points of interest, helping in highlighting significant locations for general awareness.		
Trigger: User accesses the “Points of Interest” section. Type: External		
Relationships: Association: User Include: - Extend: Add Point of Interest, Delete Point of Interest Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system displays a list of points of interest with their corresponding name, water levels, and rainfall. 2. If the User wants to add a new point of interest Execute the Add Point of Interest use case. 3. If the User wants to remove any point of interest Execute the Delete Point of Interest use case. 4. The User clicks the reload button to update the list based on the new modification. 		
SubFlows: Not applicable		
Alternate/Exceptional Flows: <ol style="list-style-type: none"> 1a. If the system is unable to retrieve points of interest or no data available, it displays a message to notify the User. 2a. If the system is unable to add new point of interest, it displays an error message and prompt the User to select again. 		

Table 3.3.4: Manage Points of Interest Use Case Description

Use Case Name: View Alerts	ID: <u>5</u>	Importance Level: <u>High</u>
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interest: User – wants to view the list of alerts.		
Brief Description: This use case describes how users can review alerts based on their points of interest, enhancing their awareness of potential flood events.		
Trigger: User accesses the alert page or receives a push notification. Type: External		
Relationships: Association: User Include: - Extend: - Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system requests notification permission to send a push notification. 2. The User manages notification permission settings. 3. The system stores the User's Firebase Cloud Messaging token into the database. 4. The system displays a list of alerts with their corresponding title, description, date and time. 		
SubFlows: Not applicable		
Alternate/Exceptional Flows: <ol style="list-style-type: none"> 2a. If the notification permission is not granted, it displays a message to notify the User. 2b. The User opens the notification setting and allows permission. 3a. If the system is unable to retrieve the alert data or no data available, it displays a message to notify the User. 3b. The User reloads the system or tries again later. 		

Table 3.3.5: View Alerts Use Case Description

Use Case Name: View Map	ID: <u>6</u>	Importance Level: <u>High</u>
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interest: User – wants to view the flood map.		
Brief Description: This use case describes how users can visualise flood-related data on an interactive map, including flood reports, monitoring stations, and search locations.		
Trigger: User accesses the map page. Type: External		
Relationships: Association: User Include: View Flood Reports, View Monitoring Stations Extend: Search Location Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system requests permission to access the User's current location. 2. The User manages location permission settings. 3. The system displays an interactive map with overlays for current location and monitoring stations. 4. If the User wants to view flood reports location, they select the flood reports option. 5. The User pan across the map to explore different regions. 6. The User clicks the marker they are interested on the map. 7. If the User wants to view marker information, they click View Details. 8. The system displays additional information about the selected marker. 9. If the User wants to navigate to Google Maps, they click the navigation button. 10. The system navigates the User to the Google Maps application. 11. If the User wants to search for a specific location Execute the Search Location use case. 		
SubFlows: Not applicable		
Alternate/Exceptional Flows: <ol style="list-style-type: none"> 2a. If the permission is not granted, the system shows a message to notify the User. 2b. The User allows location access or tries again later. 		

Table 3.3.6: View Map Use Case Description

Use Case Name: View Flood Safety Tips	ID: <u>7</u>	Importance Level: <u>Moderate</u>
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interest: User – wants to view the flood safety tips.		
Brief Description: This use case describes how users can access essential flood safety tips and guidelines to enhance their awareness and preparedness for flood events.		
Trigger: User accesses the tips page. Type: External		
Relationships: Association: User Include: - Extend: - Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system displays the page views of flood safety tips. 2. The User browses through various views, such as preparation, response, post-flood recovery, and emergency hotlines. 3. The User scrolls down, reads, and follows the provided safety guidelines. 4. The system navigates the User to the different views for additional information based on their selection. 		
SubFlows: Not applicable		
Alternate/Exceptional Flows: <ol style="list-style-type: none"> 2a. If the User wants to browse to the next page, they can click the next button. 2b. If the User wants to browse to the previous page, they can click the previous button. 		

Table 3.3.7: View Flood Safety Tips Use Case Description

Use Case Name: Submit Report	ID: <u>8</u>	Importance Level: <u>High</u>
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interest: User – wants to report any flood-related incidents.		
Brief Description: This use case describes how users can submit reports and contribute real-time information about flood-related incidents or observations.		
Trigger: User accesses the add report page. Type: External		
Relationships: Association: User Include: - Extend: - Generalization: -		
Normal Flow of Events: <ol style="list-style-type: none"> 1. The system displays a form for entering relevant details of the flood-related incident. 2. The User provides information such as a brief description, severity level, date, time, and location of the incident. 3. The User has the option to attach images from camera or gallery to supplement their report. 4. If the User wants to attach from camera, they take a photo from camera directly. 5. If the User wants to attach from gallery, they select image from their gallery. 6. The User clicks the submit button to submit the flood report. 7. The system validates the submitted report and stores them in the database. 8. The system notifies the User that their report has been successfully submitted. 		
SubFlows: Not applicable		
Alternate/Exceptional Flows: <ol style="list-style-type: none"> 7a. If the system identifies that the submitted report lacks essential information, it prompts the User to fill in the necessary details. 7b. The User completes the missing information before resubmitting the report. 		

Table 3.3.8: Submit Report Use Case Description

3.3.3 Activity Diagram

Login / Register

If users do not have an account, they enter registration information, including their username, email address, and password. The system validates the information and direct users to the home page if the information is valid. If the information is invalid, users will be prompted to input the registration details again. If they have an existing account, users key in their email address and password. The system verifies the login credentials and redirects users to the home page if the credentials are valid. If users forget their password, they enter their email address and reset their password by clicking on the link sent by the system. After resetting their password, users can sign in to the system with their new password (Figure 3.3.3).

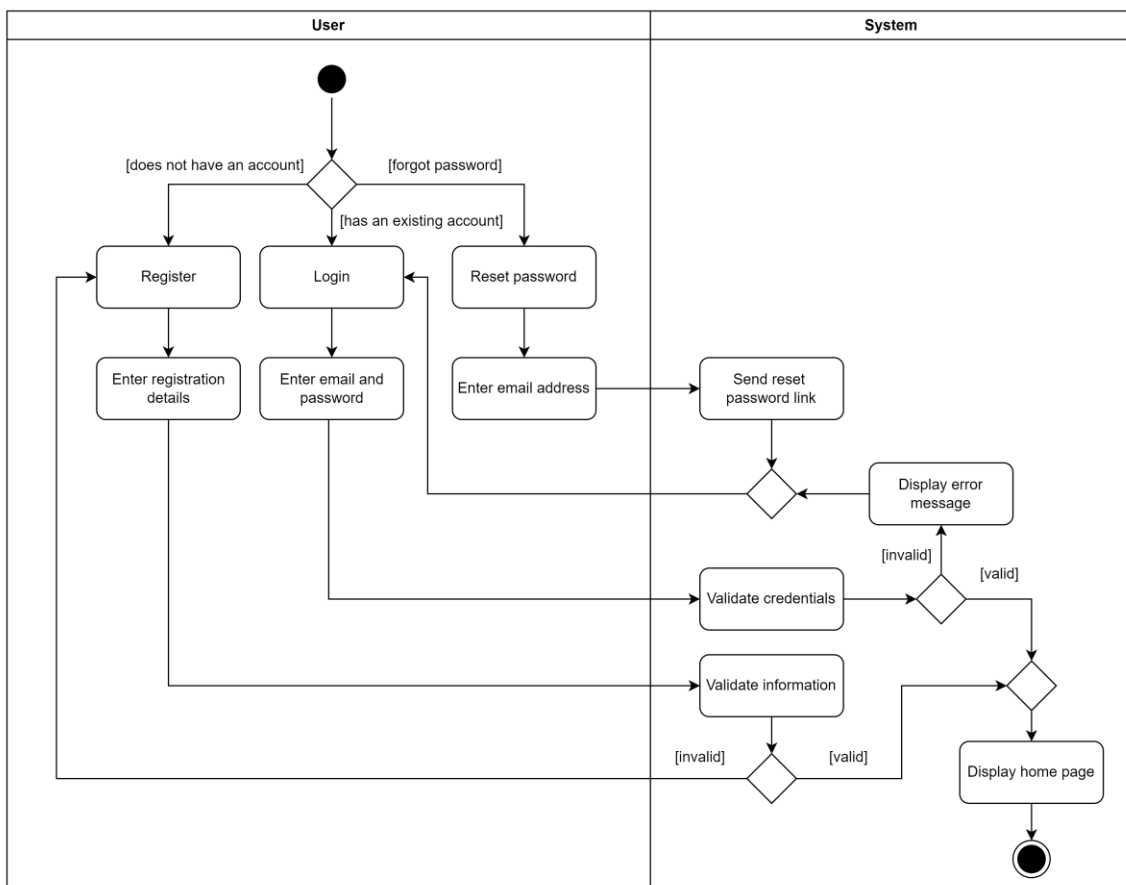


Figure 3.3.3: Login/Register Activity Diagram

Monitor Flood Information

Upon redirecting users to the home page, the system displays a dashboard with several options (Figure 3.3.4). If users select the monitoring option, the system displays a list of monitoring stations and their corresponding water levels and rainfall. Next, the system displays current weather conditions with corresponding weather forecasts if users click the weather forecasting option. When users click the weather warning option, the system displays a list of warnings. By clicking the flood reporting option, the system displays flood information reported by other users. If users select the flood status option, the system displays a detailed analysis using charts.

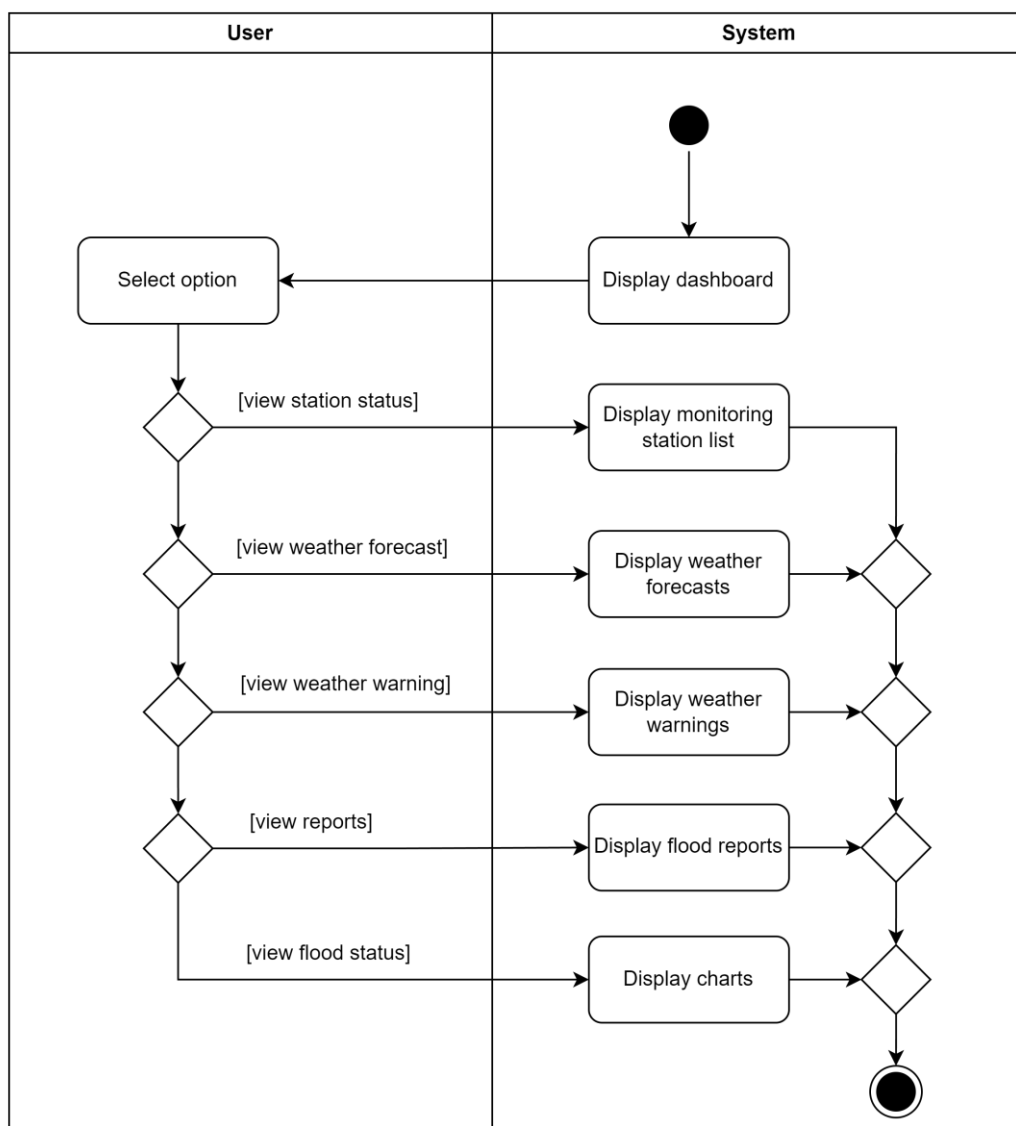


Figure 3.3.4: Monitor Flood Information Activity Diagram

View Reports

When users click the reporting page, the system retrieves data and displays reports with their corresponding date, description, image, location, and flood hazard level. As shown in Figure 3.3.5, the system displays a message to notify users if no report is available. When a user presses the valid button, the system changes the validation count, and the confidence level of the information increases. Next, users click the share button and choose the platform to disseminate the report. If users click the manage report button, the system will arrange and display their reports based on the most recent date and time.

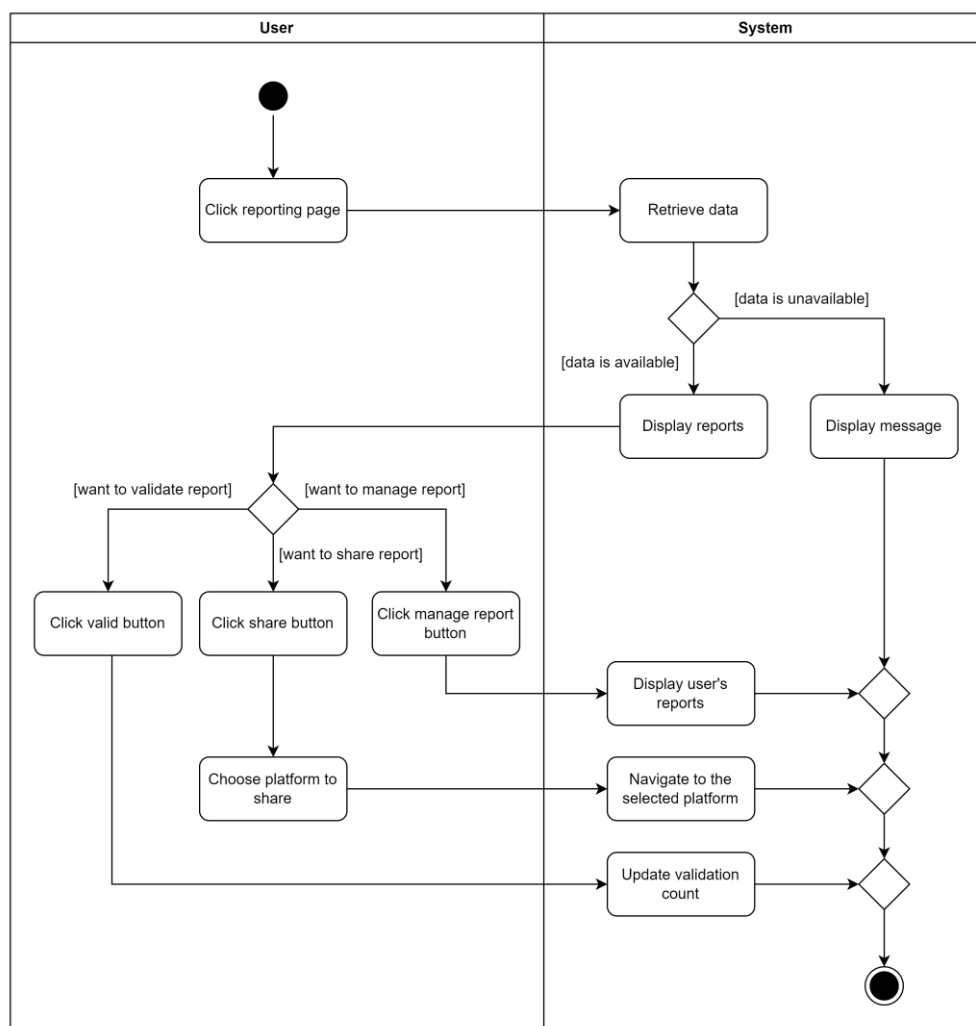


Figure 3.3.5: View Reports Activity Diagram

Manage Points of Interest

As illustrated in Figure 3.3.6, the user selects the points of interest option, and the system lists the user’s points of interest along with their associated name, water levels, and rainfall. If data is unavailable, the system displays a message informing users. Users have the option to add a new POI or delete any POI from the list. The system will check the selection when users want to add a new POI. If the system cannot add a new POI, it will display an error message and prompt users to select again. If users want to delete POI, they click the delete button and confirm deletion. The system will then remove specific data from the database. Lastly, users click the reload button to update the list based on the new modification.

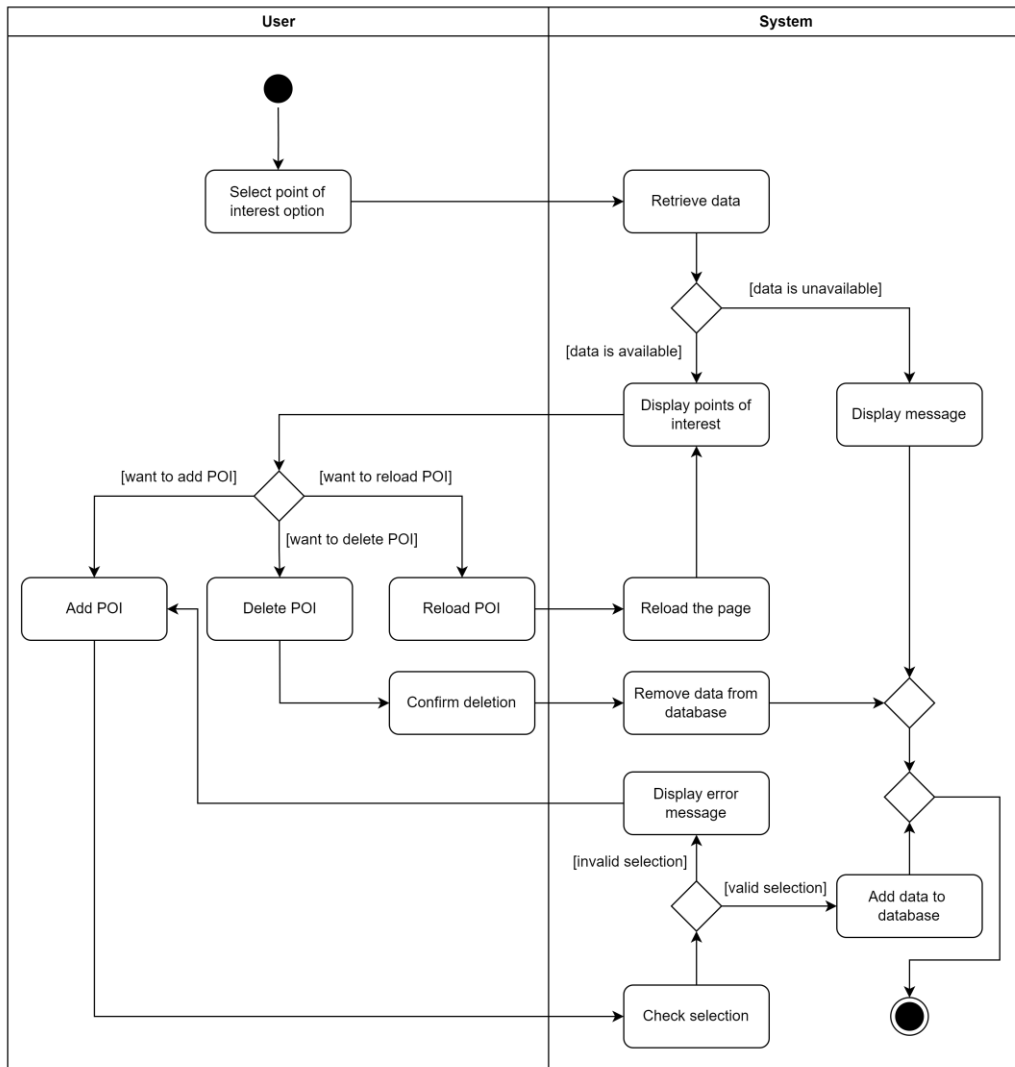


Figure 3.3.6: Manage Points of Interest Activity Diagram

View Alerts

When users click the alert page, the system asks them for permission to receive notifications (Figure 3.3.7). If permission is granted, the system will store the Firebase Cloud Messaging token in the database; if not, it will display a message to inform users. Next, the system displays a list of alerts with their corresponding title, description, date, and time. If data is unavailable, it notifies the user with a pop-up message.

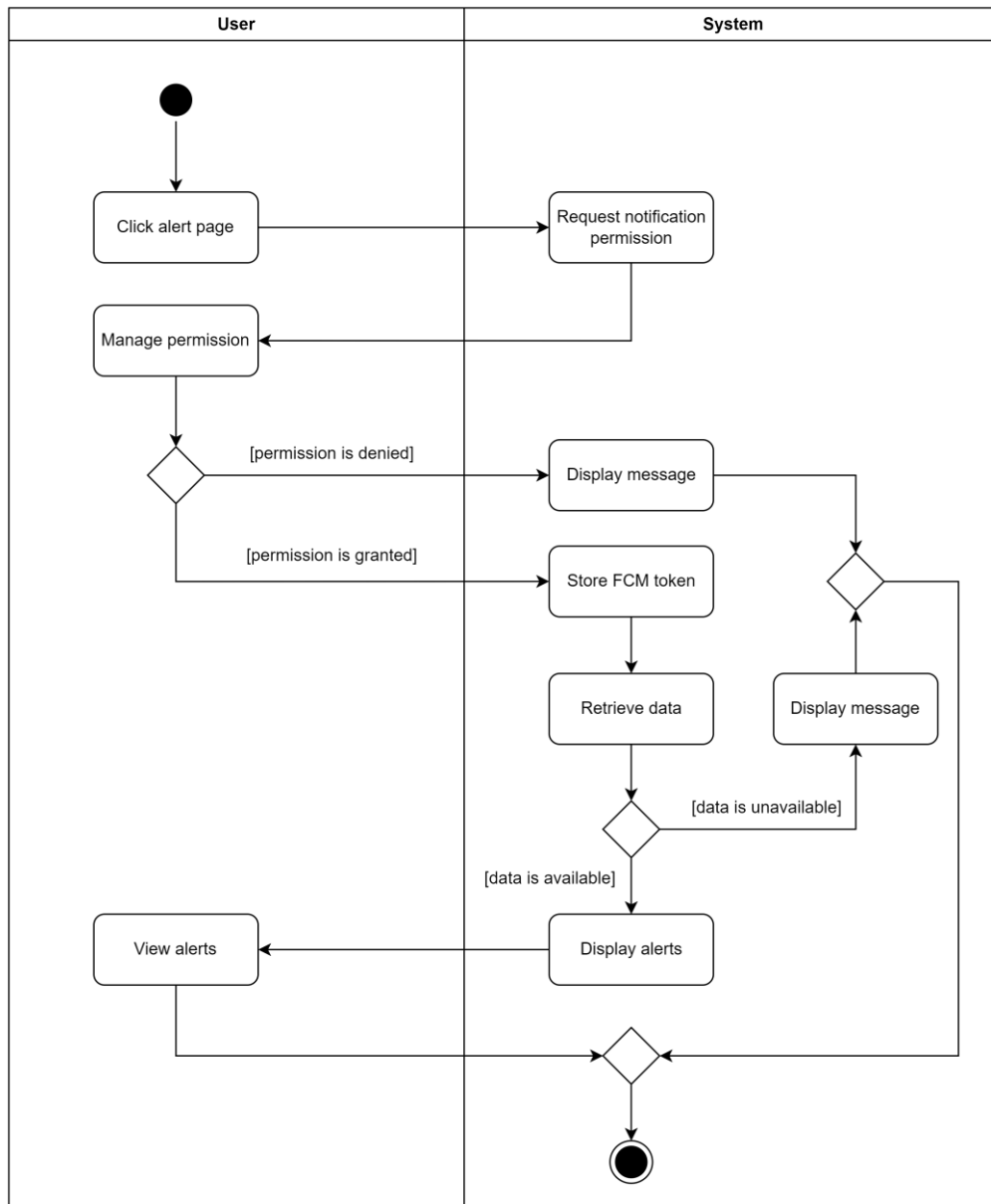


Figure 3.3.7: View Alerts Activity Diagram

View Map

Based on Figure 3.3.8, the system requests permission to access their current location when users click the map page. If the permission is denied, users will see a notification from the system. Subsequently, the system displays an interactive map featuring the user's current location and monitoring stations. If the users want to see the location of flood reports, they choose the flood reports option. Users pan around the map to explore various regions and click on the marker they are interested in. Then, the system displays additional information depending on the marker selected if users want to view details. When users click the navigation button on the map, the system navigates them to the Google Maps application. If users want to search for a specific location, they key in the location name.

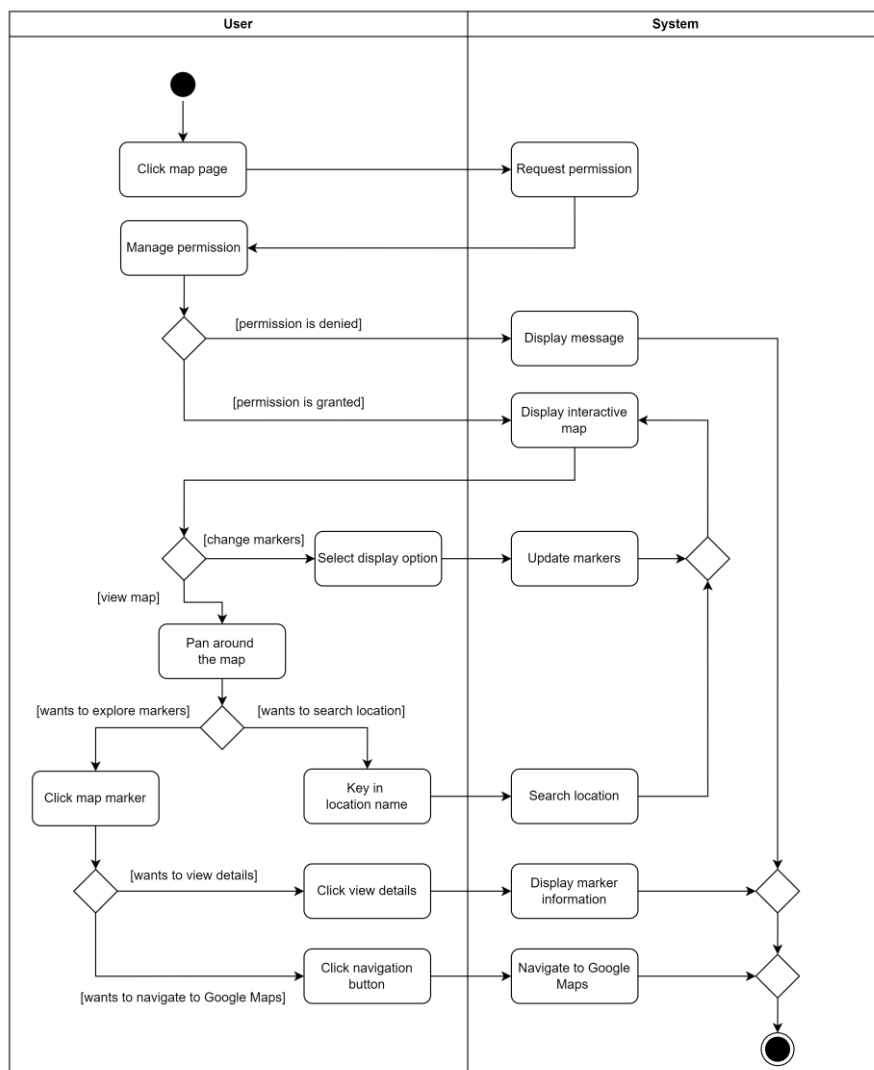


Figure 3.3.8: View Map Activity Diagram

View Flood Safety Tips

When users click the guide page, the system displays the flood safety tips on multiple page views (Figure 3.3.9). Users browse through different views, scroll down, read, and follow the safety guidelines given. Next, users select the next button to view the next page and the previous button to view the previous page. The system navigates the users to different views according to their selection.

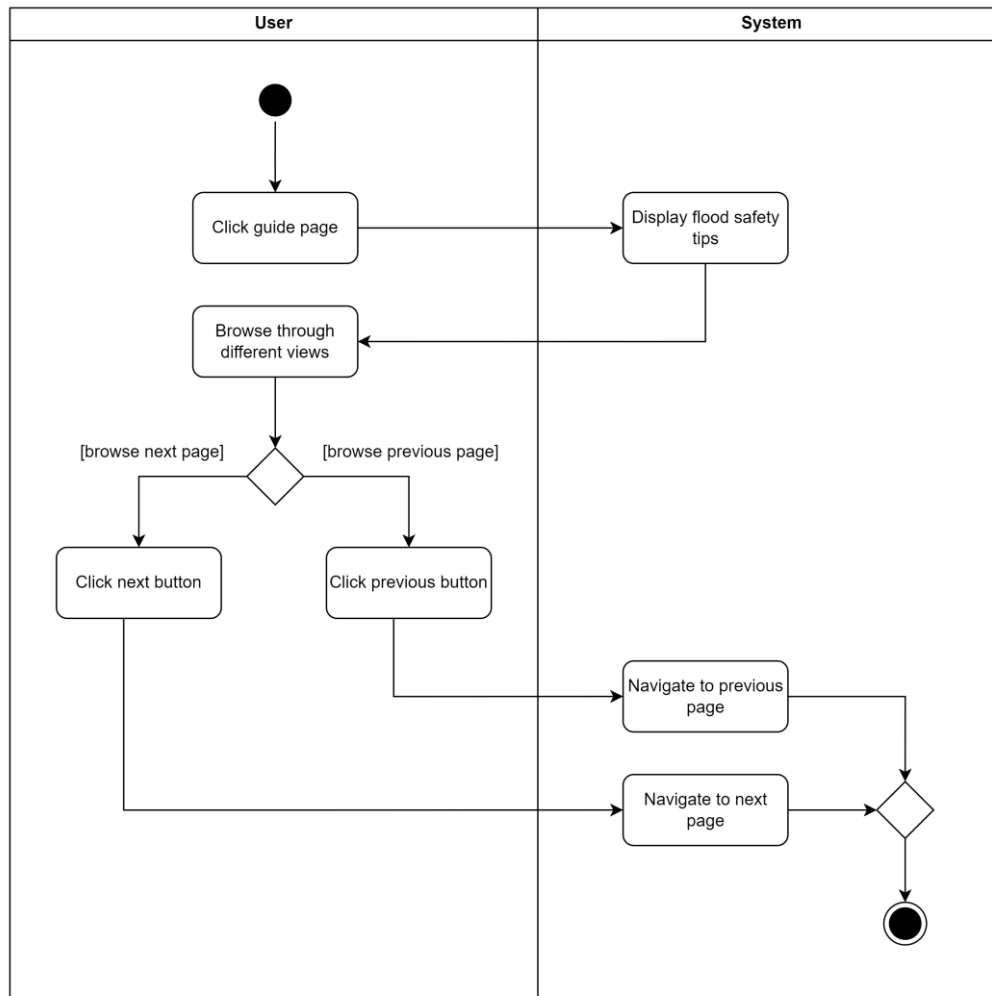


Figure 3.3.9: View Flood Safety Tips Activity Diagram

Submit Report

When users click add report button, the system will display a report form for them to fill out with relevant information about the flood-related incident. Users enter details about the incident, including a brief description, severity level, date, time, and location. Users have the option to attach photos from their camera or gallery. Next, the system validates the submitted report before storing it in the database. If the system identifies that the submitted report is missing important information, it will display a message informing the users. In contrast, the system notifies users that their report has been successfully submitted (Figure 3.3.10).

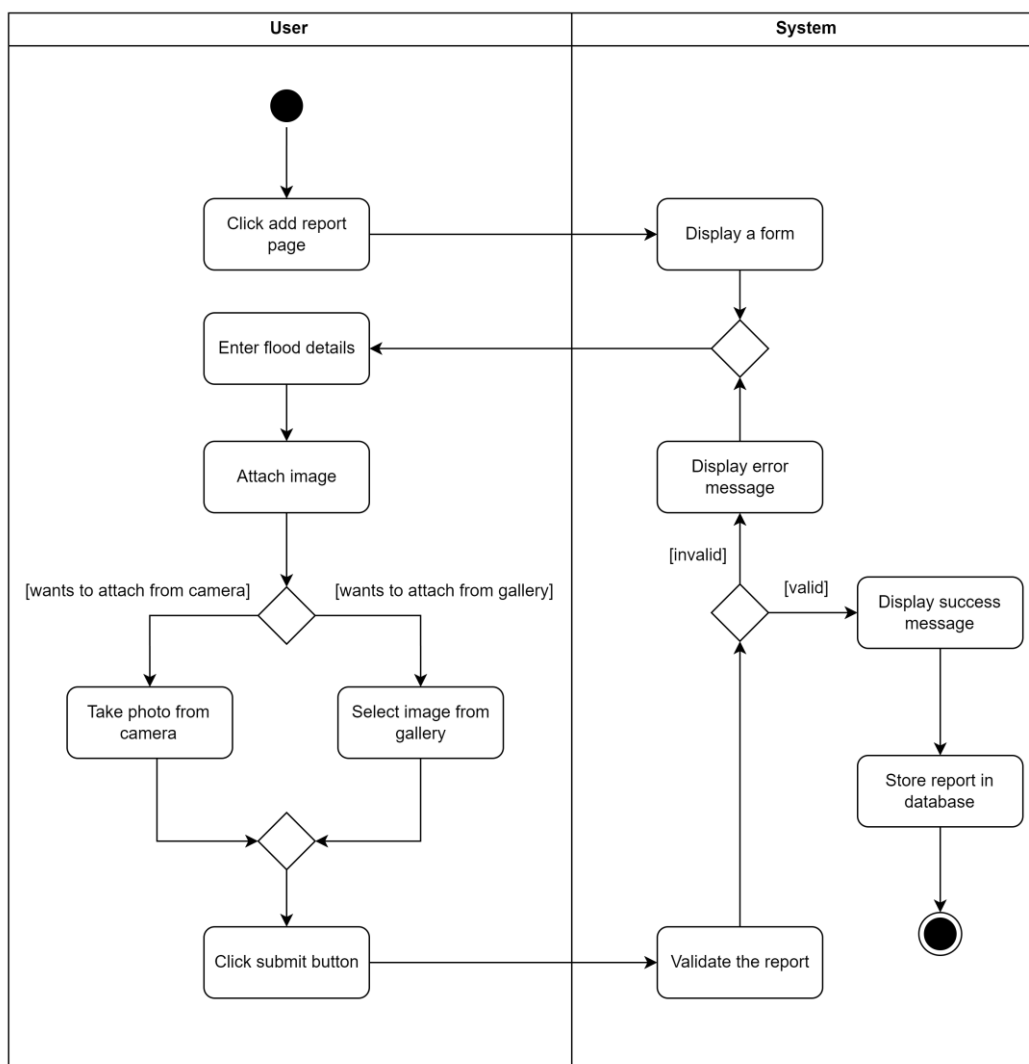


Figure 3.3.10: Submit Report Activity Diagram

Chapter 4

System Design

This chapter provides a comprehensive overview of the system, including its functionality, system flow, database, and user interface design. It starts with a top-down approach, presenting a system block diagram that summarises the primary components of the system.

4.1 System Block Diagram

4.1.1 Flutter Application

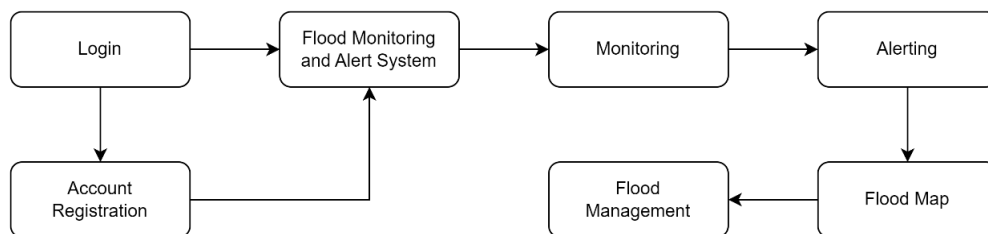


Figure 4.1.1: Block Diagram of the Flutter Application

Figure 4.1.1 shows a block diagram of the flood monitoring and alert system, which involves four primary modules such as monitoring, alerting, flood mapping, and flood management. First of all, users must first securely login to access the system with their registered email address and password. When a user attempts to sign in, Firebase Authentication safely handles the authentication process by verifying user credentials, securely storing user data, and managing user sessions. Users can also register an account with a username, email address, and password. Validation checks are included into the system to ensure the uniqueness of the registration information.

In the monitoring component, users can monitor and receive regular updates on current flood information like water levels, rainfall, and weather conditions. By integrating the Flood Warning API and Weather API, the system processes the data and presents it through a user-friendly interface, enabling users to visualise and understand the current state of water-related parameters based on accurate and timely information from these external APIs. Besides that, users can get access to flood-related information reported by other users.

In the alerting component, the system sends notifications to users through FCM based on the flood conditions monitored at their points of interest. When the monitoring module detects potential flood conditions or other critical events, the Cloud Function triggers FCM to send push notifications directly to users' devices. Furthermore, users can view the alert history about potential risks or rising water levels within the alerting module. Users can also choose to disable push notifications for receiving alerts.

In the map component, the system visualises real-time flood data on Google Maps, integrating with the Flood Warning API for additional information. Users can interact with the map interface, obtaining a comprehensive view of the flood situation and crucial information about potential flood-affected areas reported by other users. Apart from that, users can view the current status of each monitoring station around the country. The system also enables users to search for the locations they want to monitor.

In the flood management component, the system provides safety guidelines and information on evacuation procedures, emergency contacts, and general precautions before, during, and after floods. Additionally, users can report any observed flood situations, damages, and emergencies. For instance, the reports may include text descriptions, flood severity, date and time, images, and location data. This information can be valuable for emergency responders to prioritise and coordinate their efforts.

4.1.2 Flood Detection

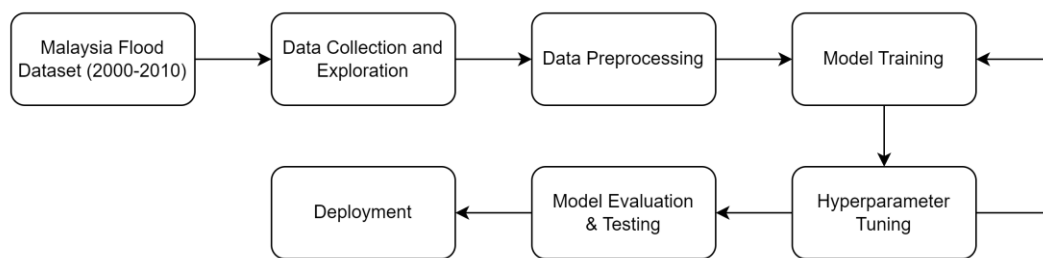


Figure 4.1.2: Block Diagram of the Flood Detection

The diagram above illustrates the backend functionality that uses deep learning techniques to detect floods (Figure 4.1.2). It involves various components such as data collection, exploration, preprocessing, model training, hyperparameter tuning, model evaluation, testing, and deployment.

Data Collection and Exploration

To begin with the implementation of the flood detection, the Malaysia Flood Dataset was downloaded from the Omdena website. The National Centre for Environment Information provided rainfall data for many states and districts in Malaysia from 2000 to 2010, which was the data source for this dataset [9]. Next, the dataset consists of 825 samples, and each sample consists of 17 variables. Columns in the dataset include the following:

Column	Description
STATE	The state in which the amount of rainfall was recorded.
DISTRICT	The district in which the amount of rainfall was recorded.
YEAR	The year where the rainfall data was measured.
JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC	The total precipitation measured in millimetres (mm) for every month of the year.
ANNUAL RAINFALL	The total rainfall reported for the year.
FLOOD	A binary variable (1 = flood, 0 = no flood) that indicates if there was a flood in the district that year.

Table 4.1.1: Data Description of Malaysia Flood Dataset

However, three columns, namely “STATE”, “DISTRICT”, and “YEAR”, have been dropped from the dataset because they might not contribute useful information for the current learning task. As a result, removing these irrelevant features can help the model perform better by reducing noise. Besides that, data exploration is carried out to understand its characteristics, patterns, and relationships, which involves visualisation techniques and data analysis to find any missing values or anomalies in the data. For example, a bar chart will be plotted to determine if the dataset is imbalanced. Handling imbalanced data is an essential part of designing a deep learning model. When the number of instances in one class significantly exceeds those in other classes, the dataset is considered imbalanced. This can make training a deep learning model complicated due to the potential for bias towards the majority class and poor performance of the minority class. Other than that, a few histograms are displayed to observe how the rainfall index changes during the rainy season, from September to January.

Data Preprocessing

The next component is data preprocessing, which includes handling missing values, feature scaling, and feature engineering tasks. Data preprocessing is essential for preparing the raw data for training the neural network. Since “FLOOD” is the target variable, it is separated from the rest of the dataset. In addition, a 9:1 random sample ratio is used to split the dataset into two sets: the training and testing sets. By dividing the dataset, the model can be trained on one part of the dataset, and its performance can be validated on a different subset. This practice can help assess its robustness to variations in the data distribution and avoid overfitting. When a model learns to memorise the training data rather than identifying the underlying patterns that generalise well to new data, this phenomenon is known as overfitting. Then, the input features of the dataset are standardised by centring the data around zero and scaling it to a comparable range, which can prevent features with bigger scales from dominating the learning process.

Model Training

For model training, a 3-layer artificial neural network is constructed and trained using the pre-processed data. The neural network will learn the patterns and correlations in the data to make accurate flood predictions. During the training process, the training loss, validation loss, training accuracy, and validation accuracy for each epoch are stored in the history object. This information can be used to analyse the model’s performance and visualise the training process. For instance, the model’s initial performance is assessed using training and validation accuracy before performing hyperparameter tuning.

Hyperparameter Tuning

For hyperparameter tuning, the RandomSearchCV is utilised to experiment with several combinations of hyperparameters, such as learning rate, optimiser, batch size, and number of epochs. Tuning is performed to identify the ideal configuration that maximises the model’s generalisation ability. After finishing hyperparameter tuning, the training data is trained on the model using the best parameters found. Next, the accuracy of the training and validation is calculated and compared with the initial model.

Model Evaluation & Testing

As the optimal model has been selected and trained, it is evaluated using a variety of performance metrics to assess its performance. For example, this is commonly done using metrics like accuracy score, confusion matrix, precision, recall, and F1-score. After evaluating the model's performance, a few samples from the testing set are applied to test the model's effectiveness in actual situations. This stage helps ensure the model can generalise well to new, unseen instances and perform reliably in real-world environments.

Deployment

Once the model has been trained, evaluated, and tested successfully, it is saved and converted to TensorFlow Lite format. Then, the TensorFlow Lite model is stored as a file that can be deployed and executed on other platforms, such as mobile devices and web applications. By integrating the model into the application, users can now detect the likelihood of a flood based on rainfall data input.

4.2 Database Design

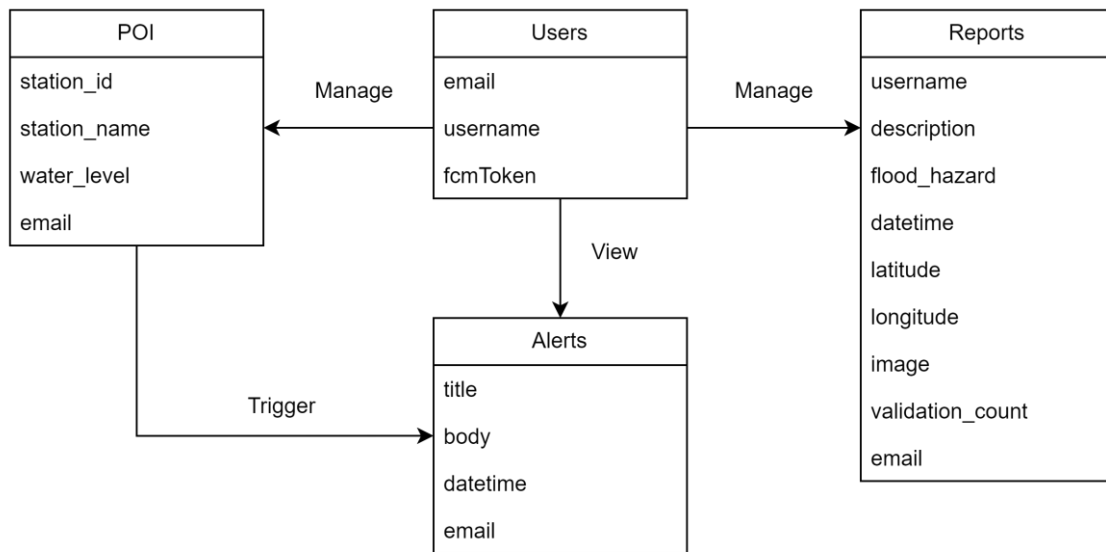


Figure 4.2.1: Database Design of the Proposed System

In this project, a NoSQL document database named Cloud Firestore was used in the flood monitoring and alert system to organise and manage data efficiently. The database design includes several classes to represent key entities within the system (Figure 4.2.1). The “Users” class includes attributes such as username, email address, and fcmToken. Personalised flood monitoring is facilitated by the “POI” class, which consists of station ID, station name, water level, and user’s email. The “Alerts” class records any alerts triggered, with the alert’s title, description, datetime, and email. Lastly, user-generated flood reports are organised through the “Reports” class, including report description, flood hazard level, datetime, coordinates, image URL, validation count, username, and email. These classes are interconnected through relationships, ensuring data integrity and enabling a comprehensive representation of the system. A user can manage one or many points of interest, but each point of interest is associated with one user. Next, a user can view one or many alerts, but each alert is associated with one user only. A user can manage one or many reports, but each report belongs to one user. Lastly, each POI record update can trigger only one alert, and that alert is created based on that specific update.

4.3 User Interface Design

This section focuses on the layout, features, and user interactions of the Flutter application, which includes wireframes or UI mock-ups for the four important modules: monitoring, alerting, flood map, and flood management.

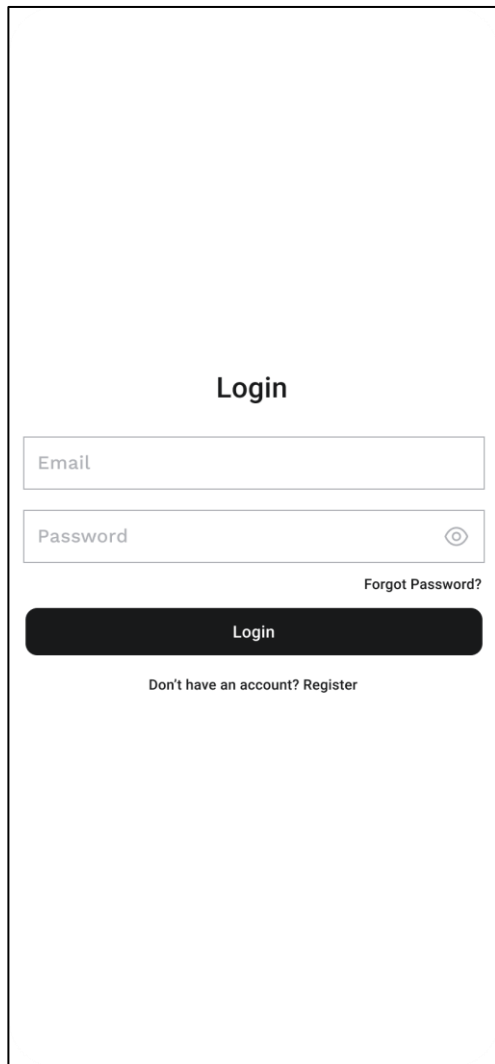


Figure 4.3.1: Login Page

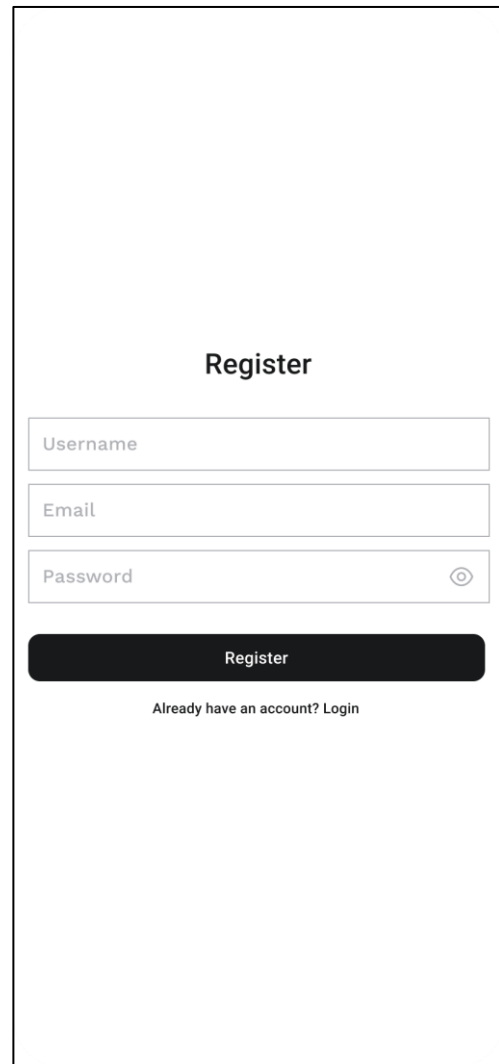


Figure 4.3.2: Register Page

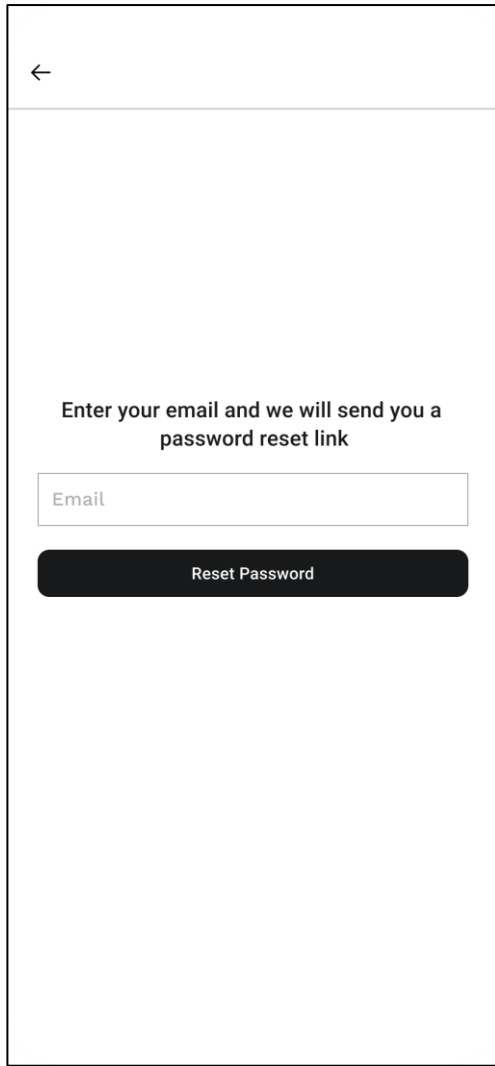


Figure 4.3.3: Reset Password Page

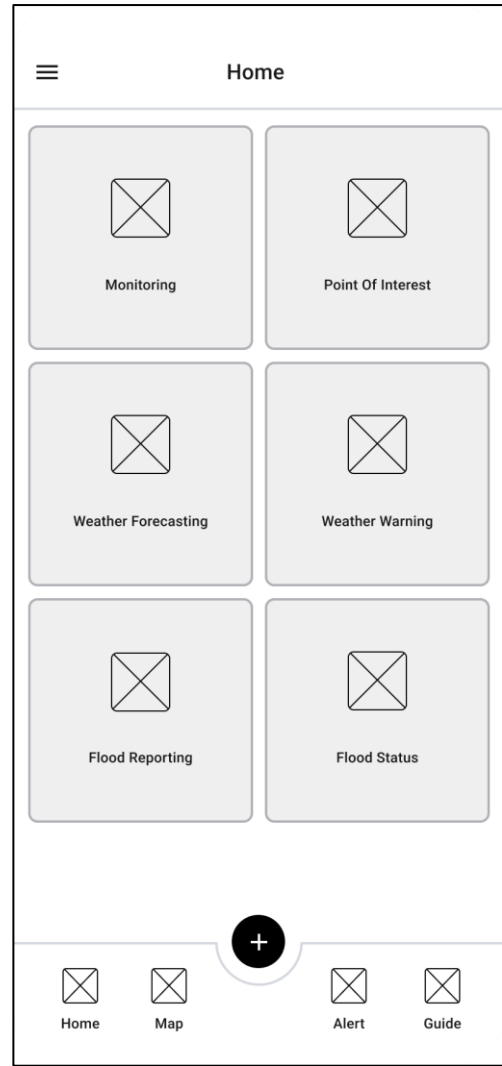


Figure 4.3.4: Home Page

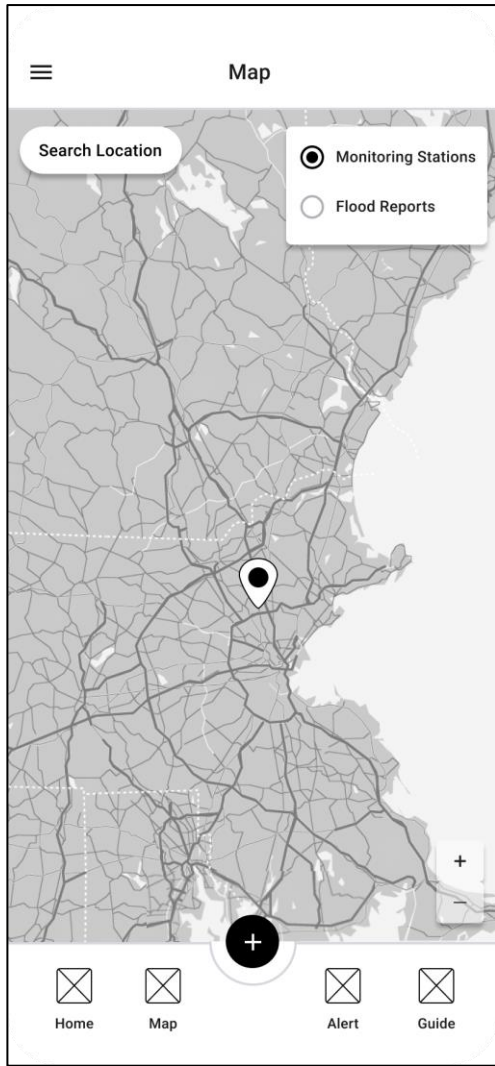


Figure 4.3.5: Map Page

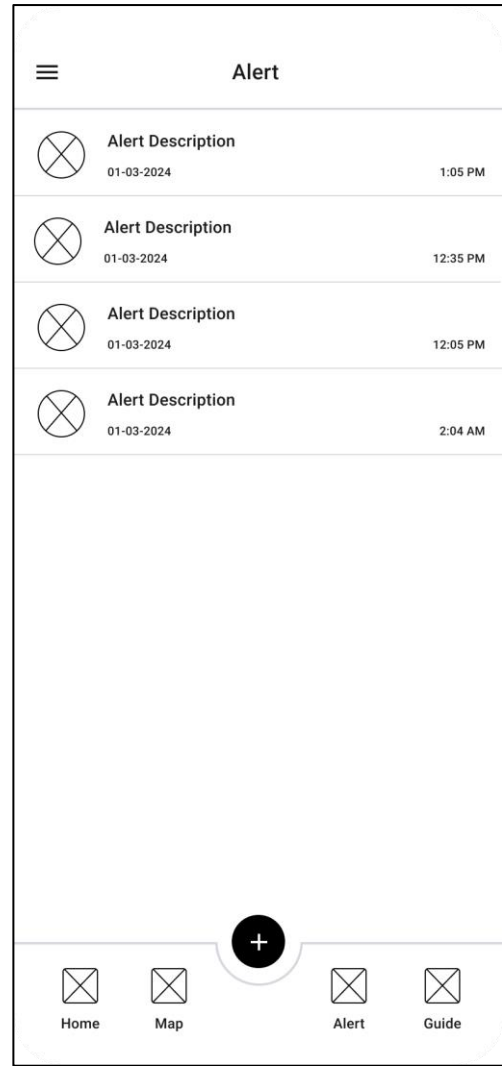


Figure 4.3.6: Alert Page

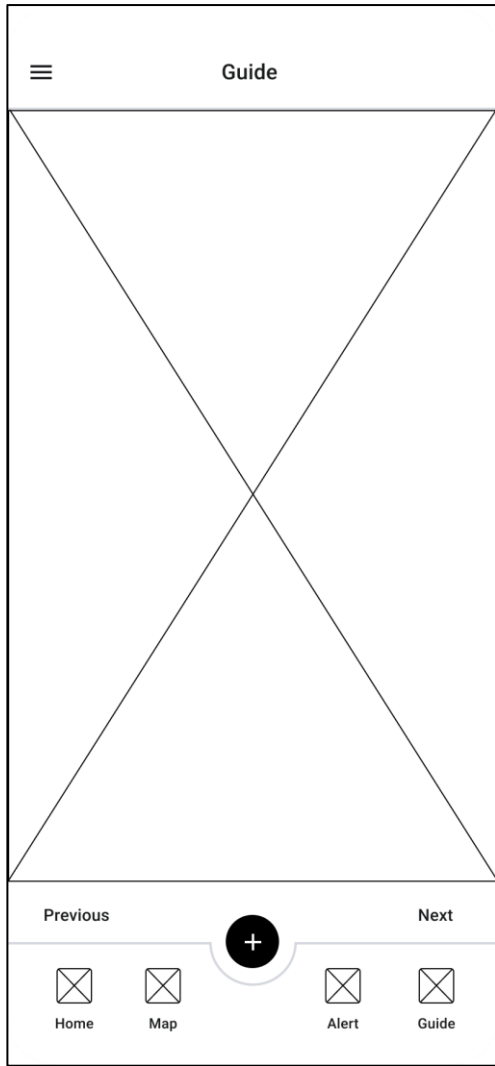


Figure 4.3.7: Guide Page

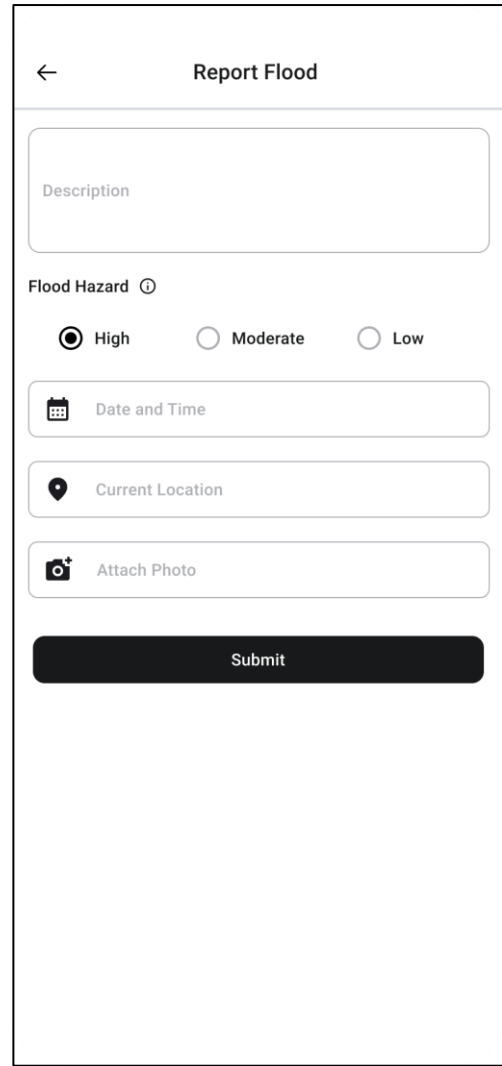


Figure 4.3.8: Report Flood Page

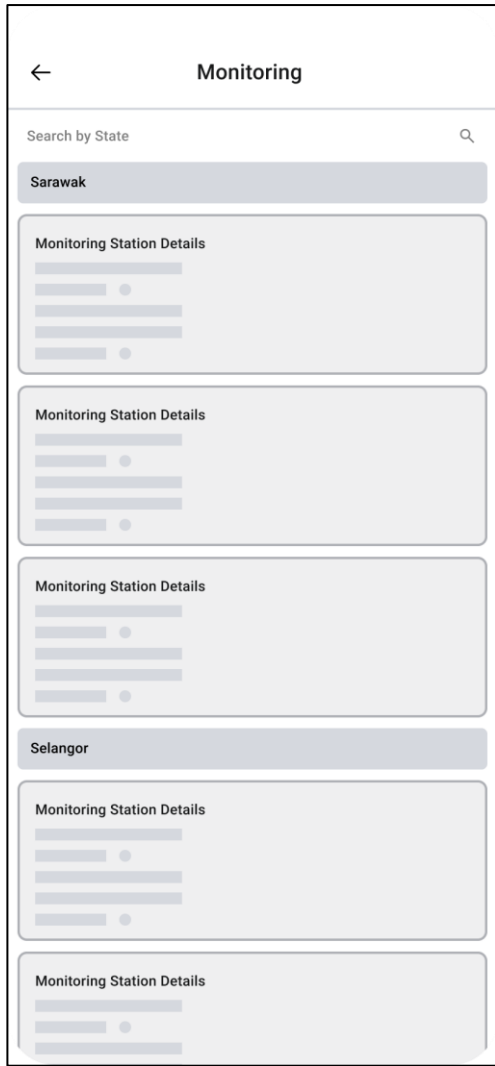


Figure 4.3.9: Monitoring Page

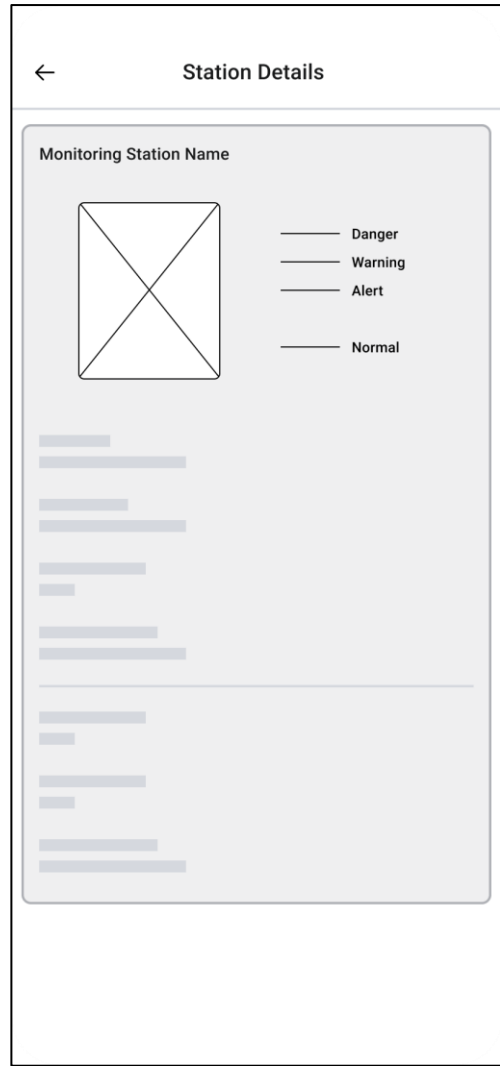


Figure 4.3.10: Station Details Page



Figure 4.3.11: POI Page

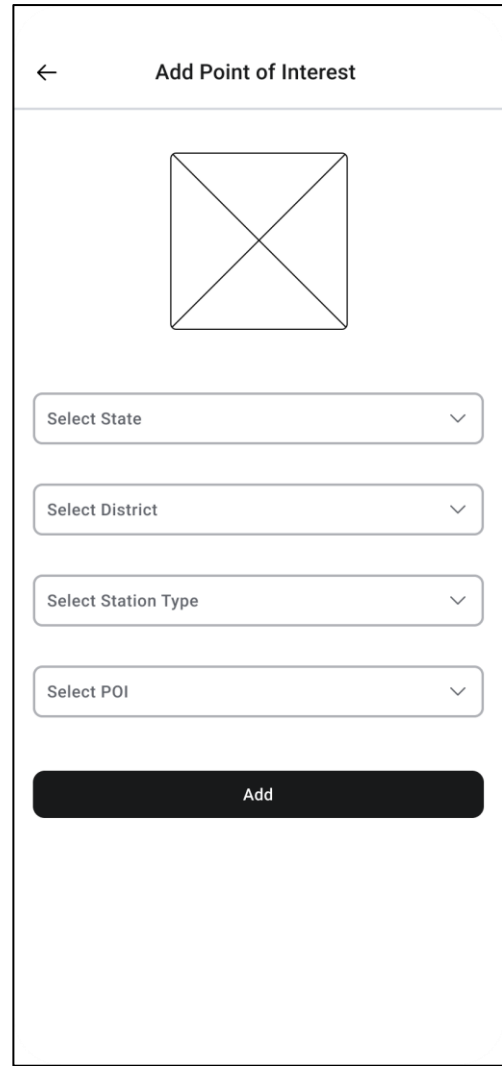


Figure 4.3.12: Add POI Page

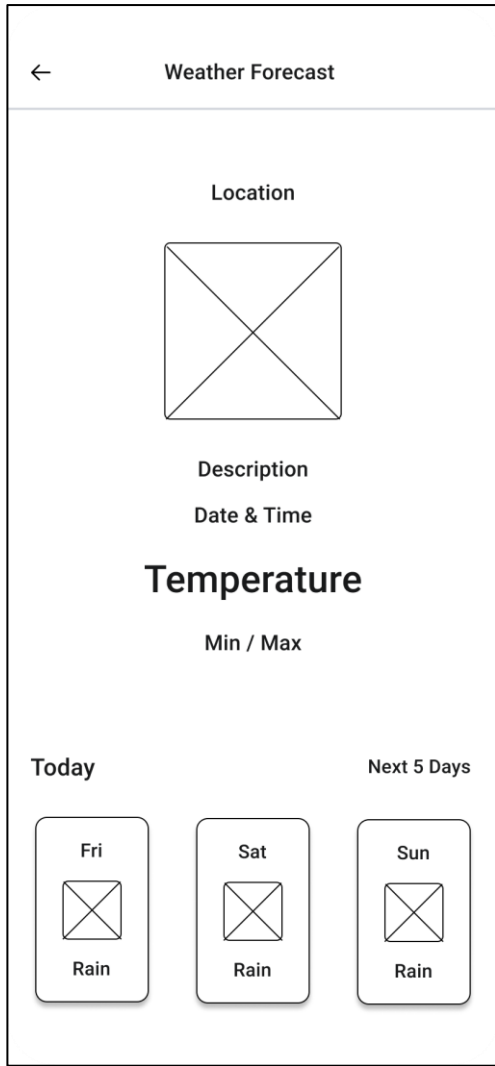


Figure 4.3.13: Weather Forecast Page

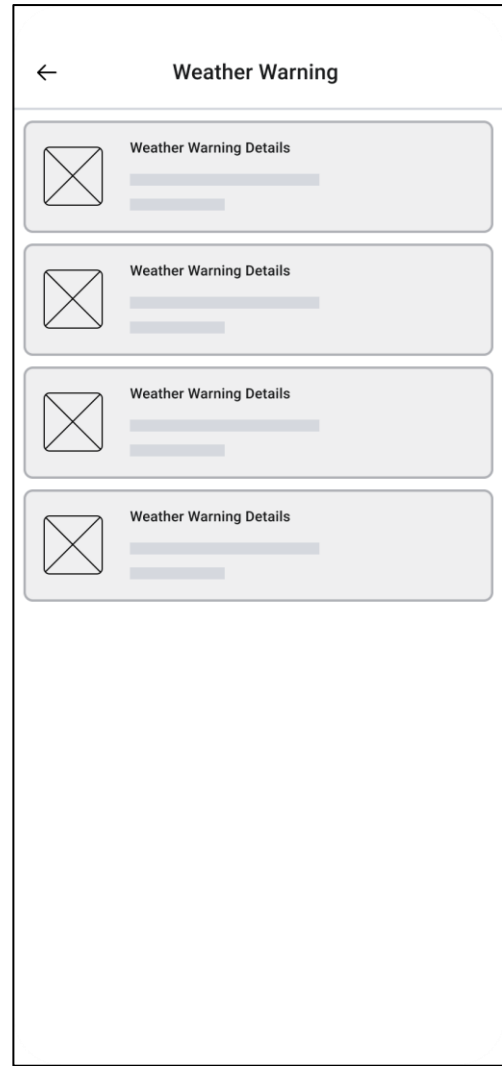


Figure 4.3.14: Weather Warning Page

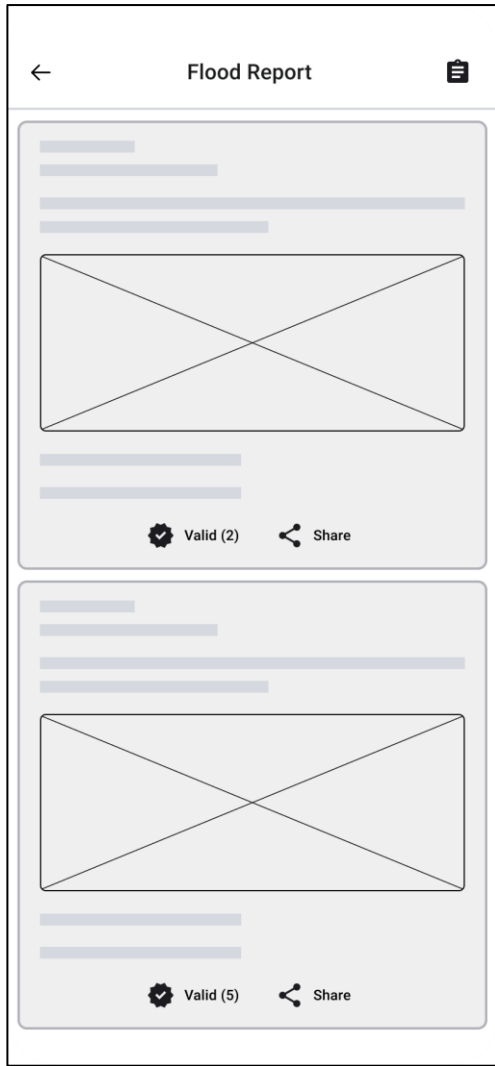


Figure 4.3.15: Report Page

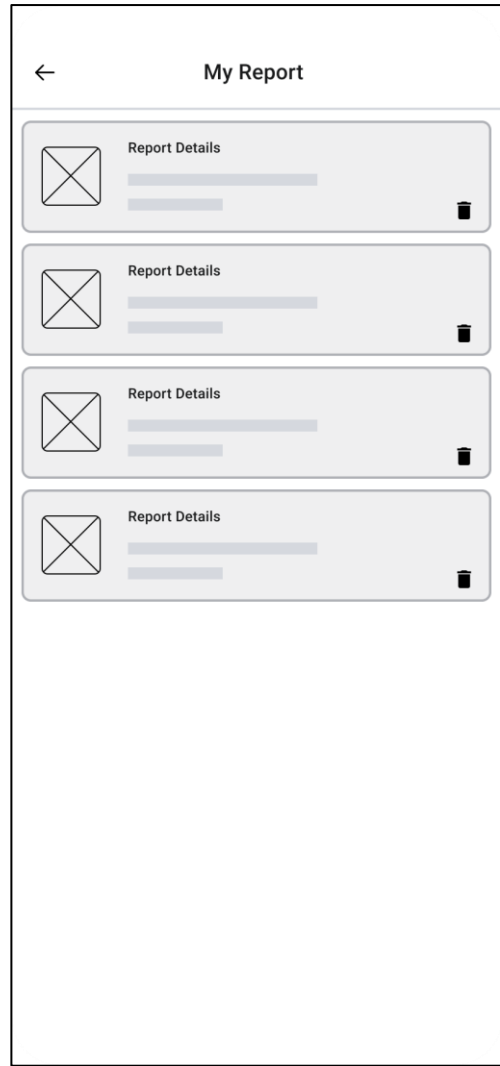


Figure 4.3.16: Manage Report Page



Figure 4.3.17: Flood Status Page

Chapter 5

System Implementation

The practical foundation of the project is system implementation, which describes how the proposed system or solution is implemented in the actual world. This chapter will include hardware setup, software setup, setting and configuration, system operation, implementation challenges, and a concluding remark to summarise the entire implementation process.

5.1 Hardware Setup

The hardware used in the development is a computer and a mobile phone. A laptop computer is utilised to code and develop the mobile application using Flutter with Dart language, whereas a smartphone is used in USB Debugging for in-device demo of Flutter code.

Description	Specifications
Model	HUAWEI MateBook D 15
Processor	AMD Ryzen 5 3500U
Operating System	Windows 10
Graphic	Radeon Vega 8 Graphic
Memory	8GB DDR4 2400MHz
Storage	256GB NVMe PCIe SSD

Table 5.1.1: Specifications of Laptop

Description	Specifications
Model	POCO X3 NFC
Processor	Octa-core Max 2.30GHz
Operating System	Android 12
Memory	128GB 6GB RAM

Table 5.1.2: Specifications of Smartphone

5.2 Software Setup

The software used in this project are:

i. Android Studio Giraffe

An integrated development environment (IDE) called Android Studio is made specifically for developing Android applications [13]. It is the official IDE for Android development and is offered by Google. A Gradle-based build system, an Android emulator, code editing, debugging, and other capabilities designed specifically for Android development are all included in Android Studio, which is built on top of the IntelliJ IDEA IDE.

ii. Visual Studio Code

Microsoft produced a free and open-source code editor called Visual Studio Code (VS Code) [14]. Most developers use it extensively because it is a lightweight, cross-platform editor that works with many different programming languages and frameworks. As opposed to full-fledged IDEs, VS Code focuses on providing a streamlined and customisable code editing experience with essential features for modern development.

Before I could begin developing the system, there were two software needed to be downloaded and installed on my laptop, which are Android Studio and Visual Studio Code. First, go to the official Android Studio website and download the Android Studio installer (Figure 5.2.1). Then, follow the installation instructions provided by the installer. As shown in Figure 5.2.2, users will be asked to select the installation location, choose the components to be installed, and agree to the terms and conditions. Android Studio will also prompt users to install any required SDKs, build tools, and other dependencies. Once the setup process is finished, users can start developing Android applications using Android Studio.

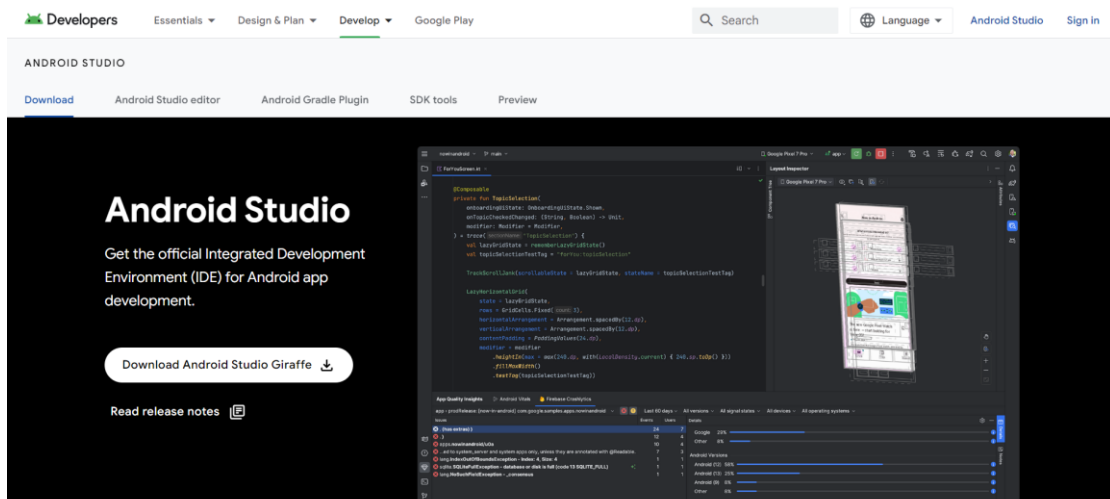


Figure 5.2.1: Installation of Android Studio

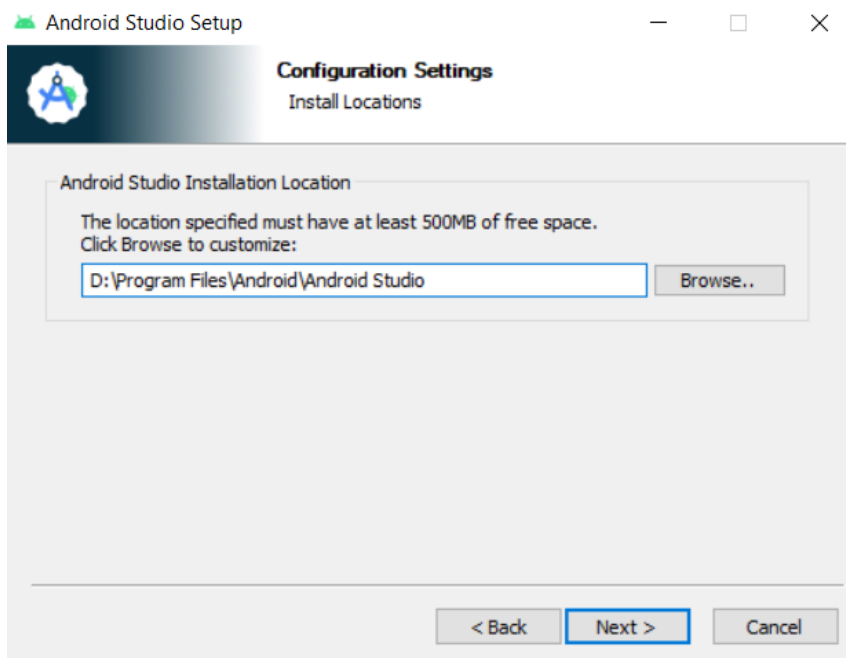


Figure 5.2.2: Installation of Android Studio (Cont.)

Next, visit the official Visual Studio Code website and download the installer (Figure 5.2.3). Once the download is complete, run the installer and follow the on-screen installation instructions to install VS Code. As demonstrated in Figure 5.2.4, users will be required to read and accept the terms of agreement before proceeding with the installation. After the setup process is completed, users can start developing Flutter applications using VS Code.

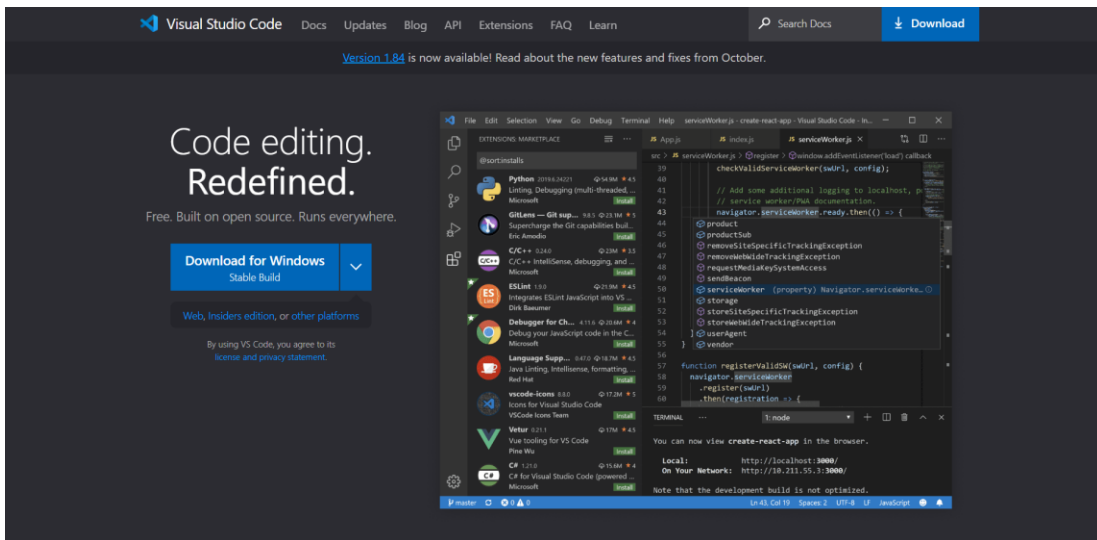


Figure 5.2.3: Installation of Visual Studio Code

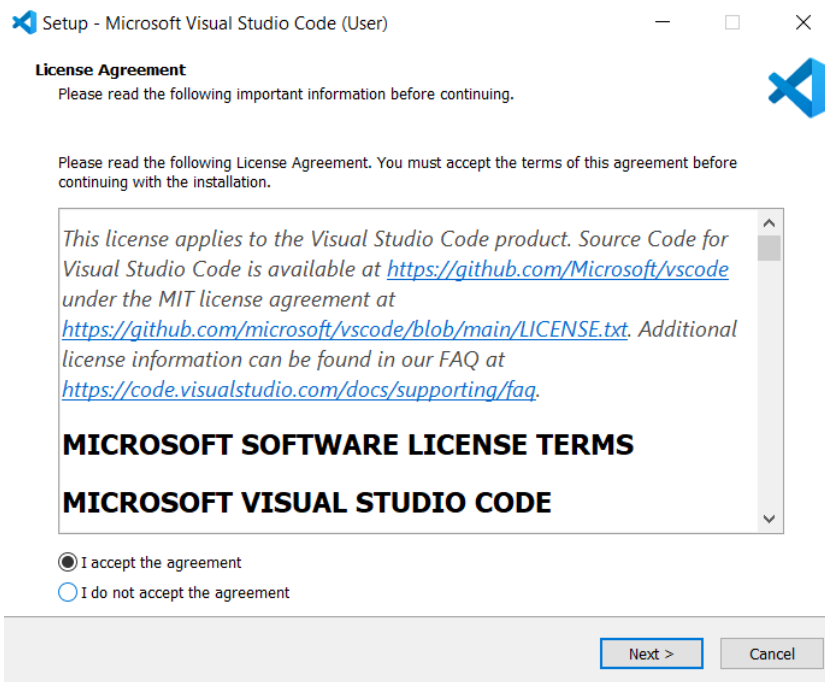


Figure 5.2.4: Installation of Visual Studio Code (Cont.)

To develop the Flutter application, the Flutter SDK needs to be installed on my laptop. First, select the operating system and download the latest stable version of Flutter for that operating system from the official Flutter website (Figure 5.2.5). Once the installation is complete, extract the downloaded ZIP file to a directory.

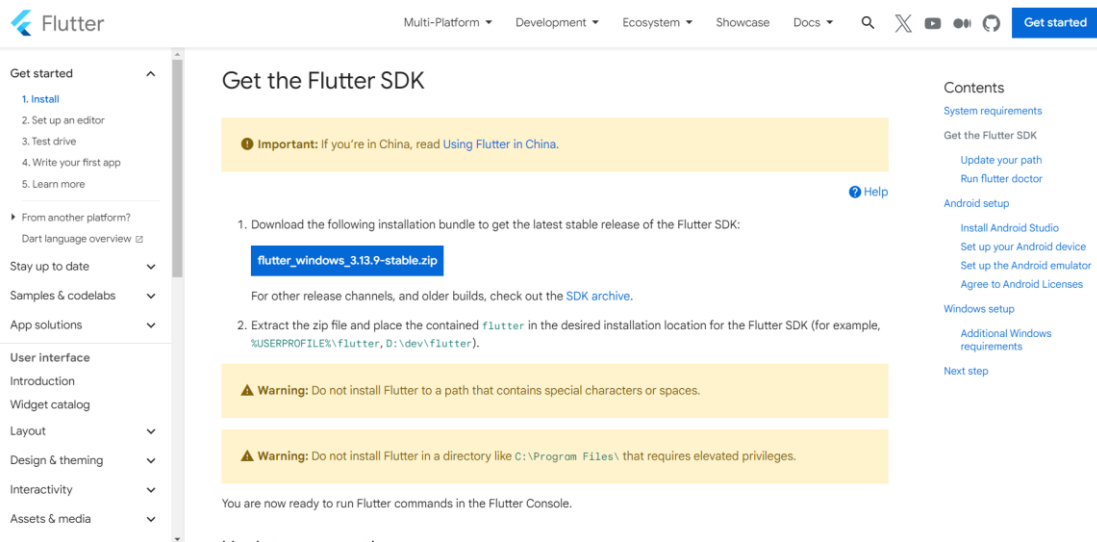


Figure 5.2.5: Installation of Flutter

To run Flutter commands globally, add the Flutter binary directory to the system PATH. In the settings, search for “Edit the system environment variables.” Next, select “path” and click the “edit” button (Figure 5.2.6). As shown in Figure 5.2.7, add the full path to the “flutter/bin” directory to the system PATH. Then, click “OK”.

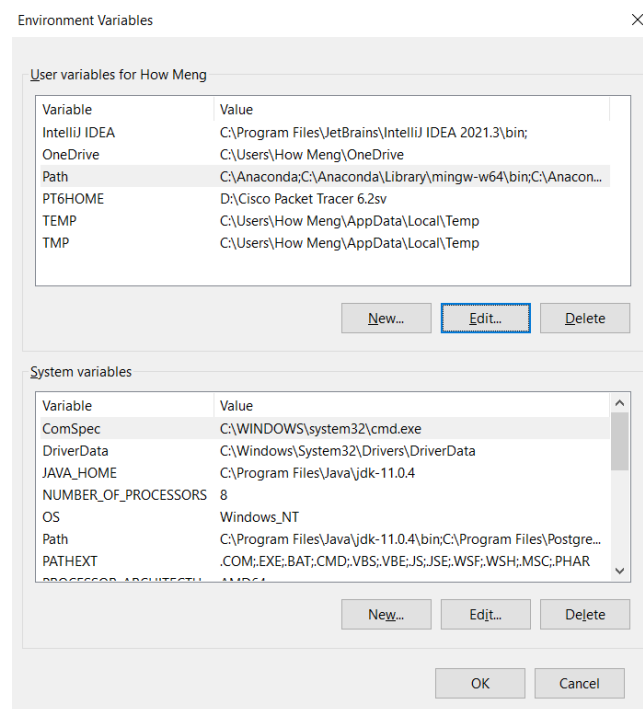


Figure. 5.2.6: Add Flutter to PATH

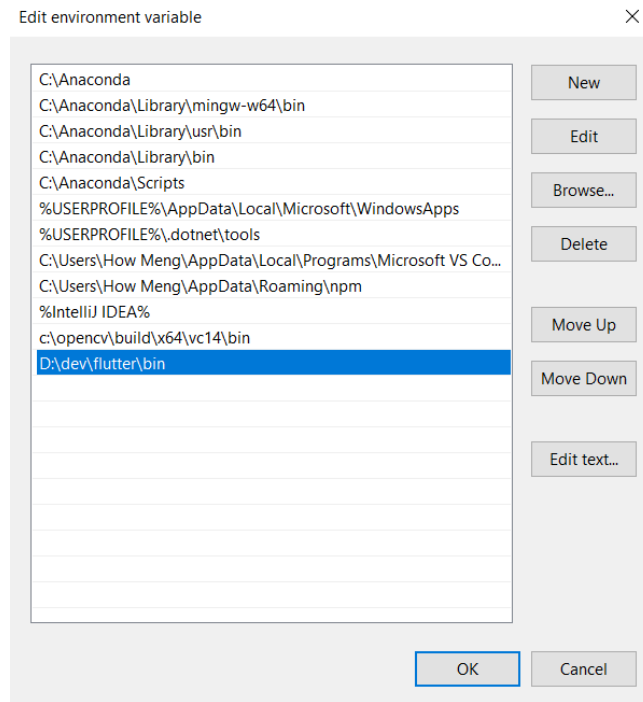


Figure 5.2.7: Add Flutter to PATH (Cont.)

After that, open a terminal or command prompt and run the command “flutter doctor” to check for dependencies and to ensure that everything is configured correctly (Figure 5.2.8). Based on the summary, address any issues reported by the “flutter doctor” command, such as installing additional dependencies if needed.

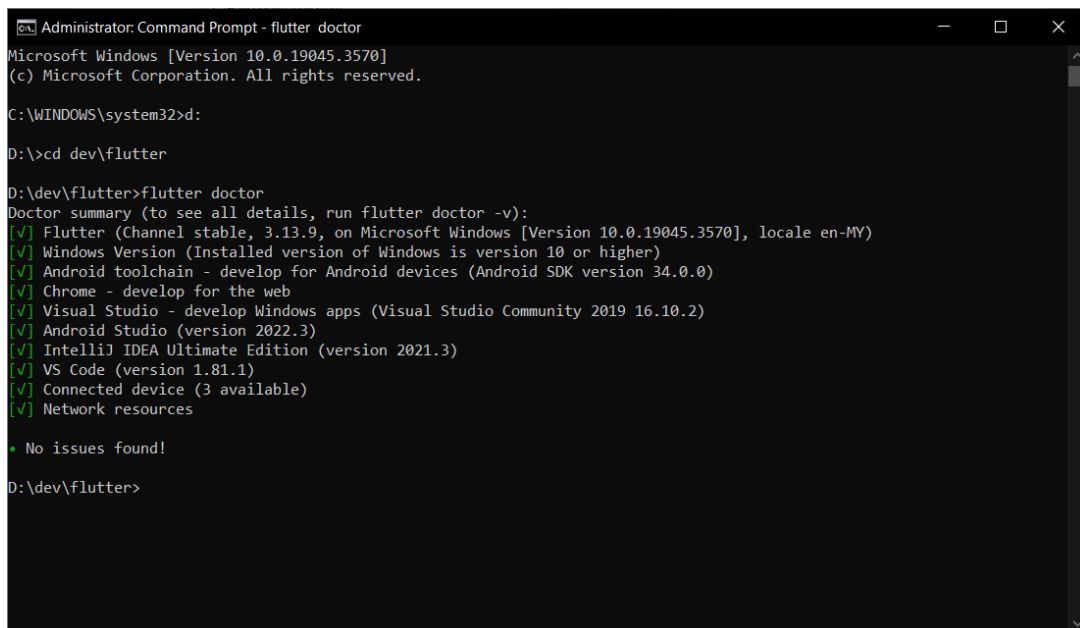


Figure 5.2.8: Run “flutter doctor” Command

To use Google Firebase, go to the Firebase website and sign in using a Google account. Next, go to the Firebase Console and click on the “Add Project” button, and follow the setup wizard to create a new project (Figure 5.2.9). Once the project is created, users will be redirected to the project dashboard.

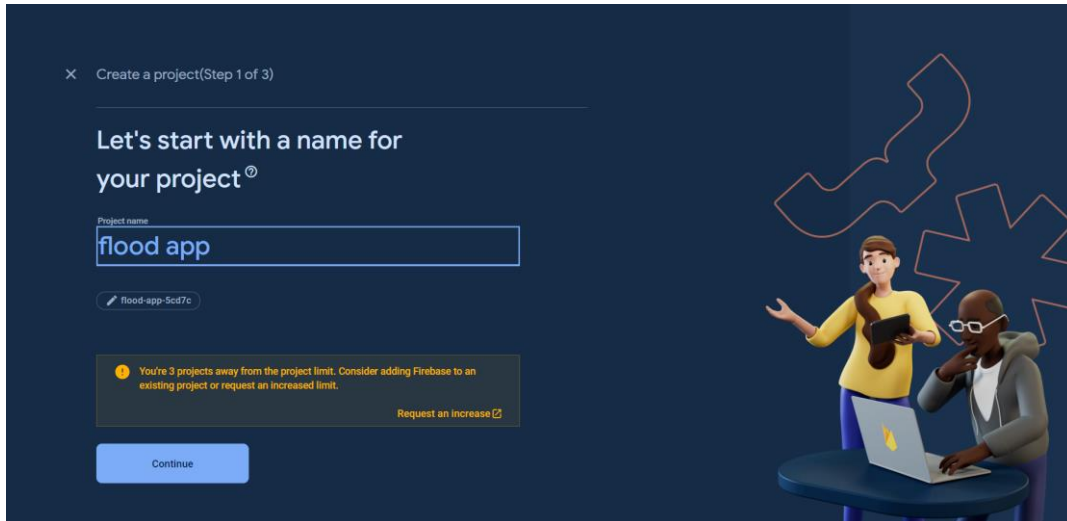


Figure 5.2.9: Creation of Google Firebase

5.3 Setting and Configuration

There are a few settings and configurations that must be made to develop the program using Android Studio and VS Code. To create a Flutter project, open Android Studio and click “New Flutter Project” (Figure 5.3.1). Next, ensure that the Flutter SDK path is assigned to the correct directory (Figure 5.3.2). As demonstrated in Figure 5.3.3, enter the project name and description and select platforms. Lastly, click the “Create” button, and the project is successfully created.

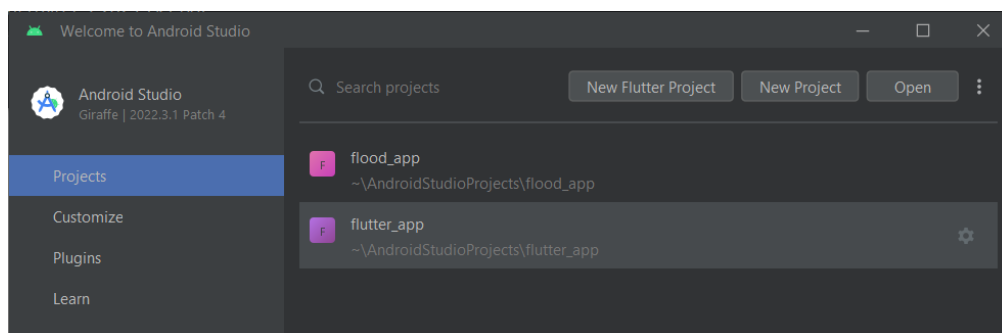


Figure 5.3.1: Android Studio Configuration

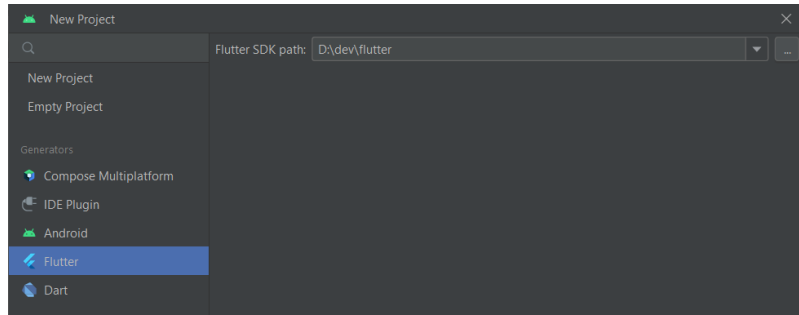


Figure 5.3.2: Flutter SDK Path

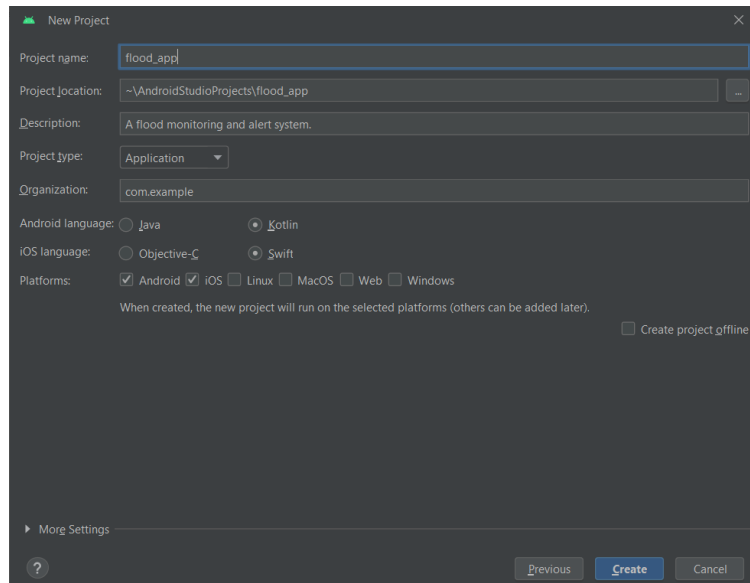


Figure 5.3.3: Create New Project

To code the Flutter project in VS Code, several extensions need to be installed, such as Flutter, Dart, and Awesome Flutter Snippets. The figures below show the extensions needed for the Flutter project.

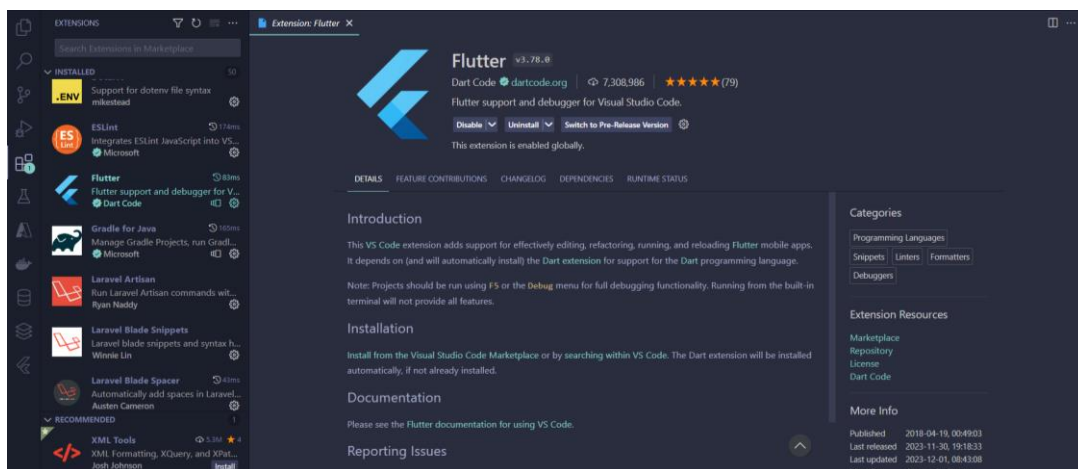


Figure 5.3.4: Flutter Extension

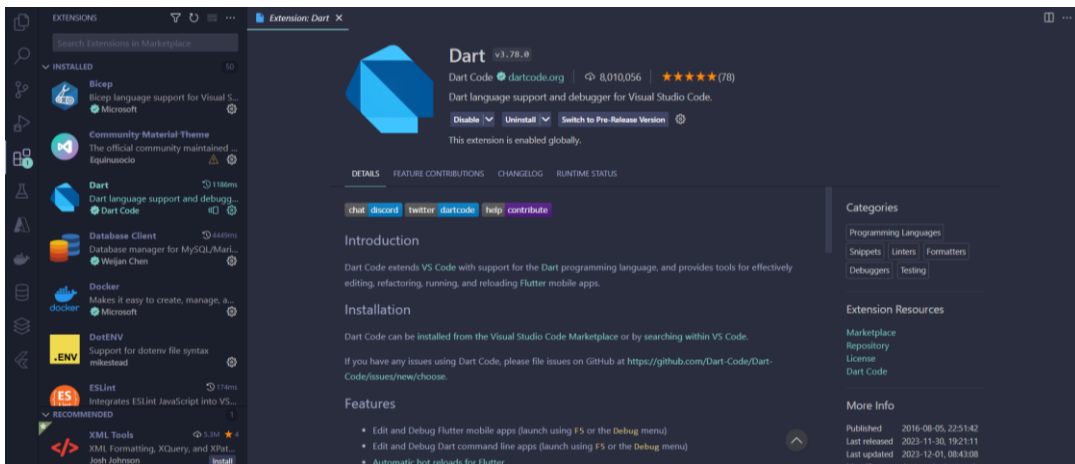


Figure 5.3.5: Dart Extension

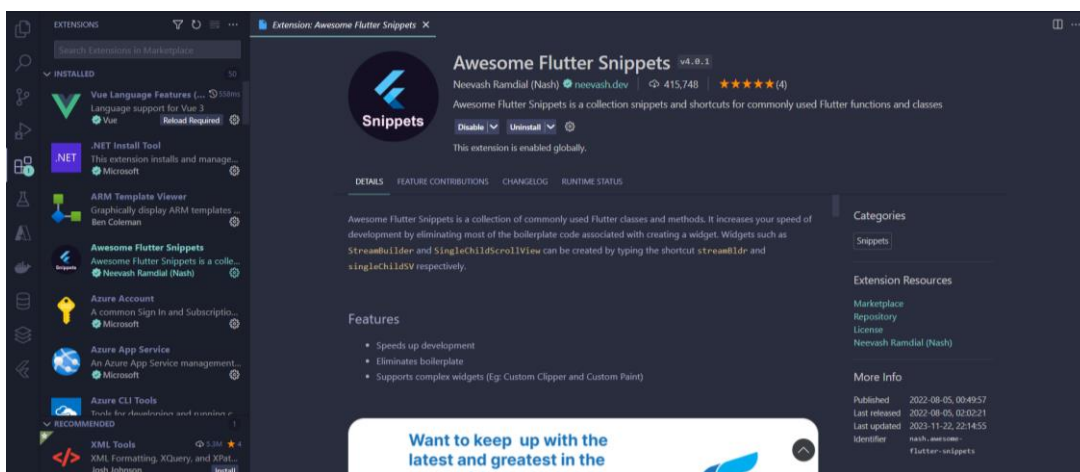


Figure 5.3.6: Awesome Flutter Snippets Extension

To add Firebase to the Flutter project, a few configurations need to be made. First, click on the “Add Firebase to your Android app” button in the Firebase dashboard. Next, enter the android package name, which can be found in the build.gradle file under the android app folder (Figure 5.3.7). As demonstrated in Figure 5.3.8, download the “google-services.json” configuration file and move it into the app folder of the Android Studio project.

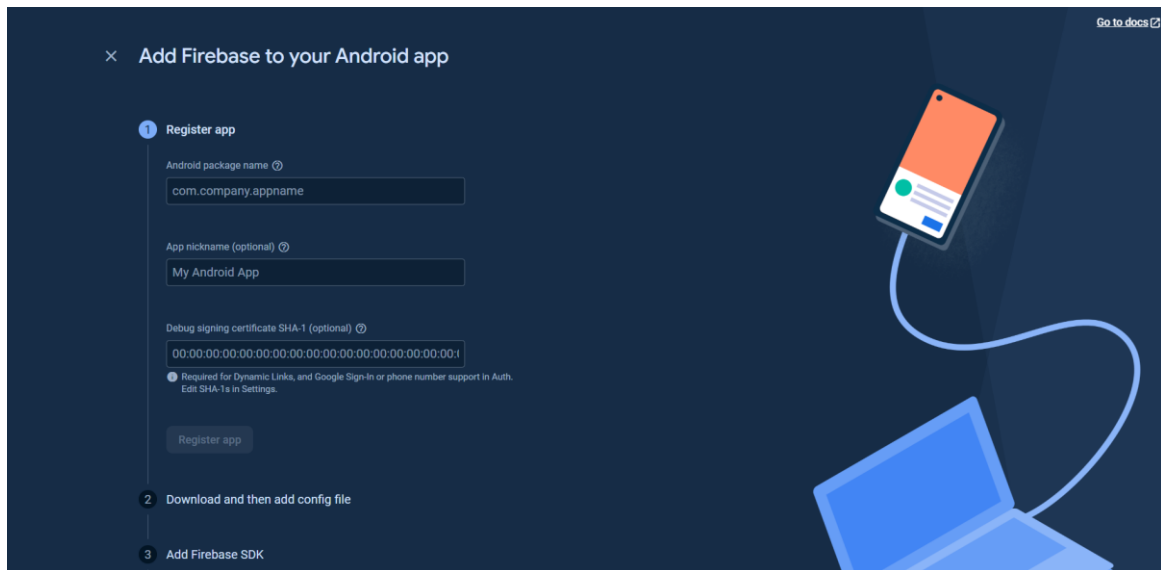


Figure 5.3.7: Register App

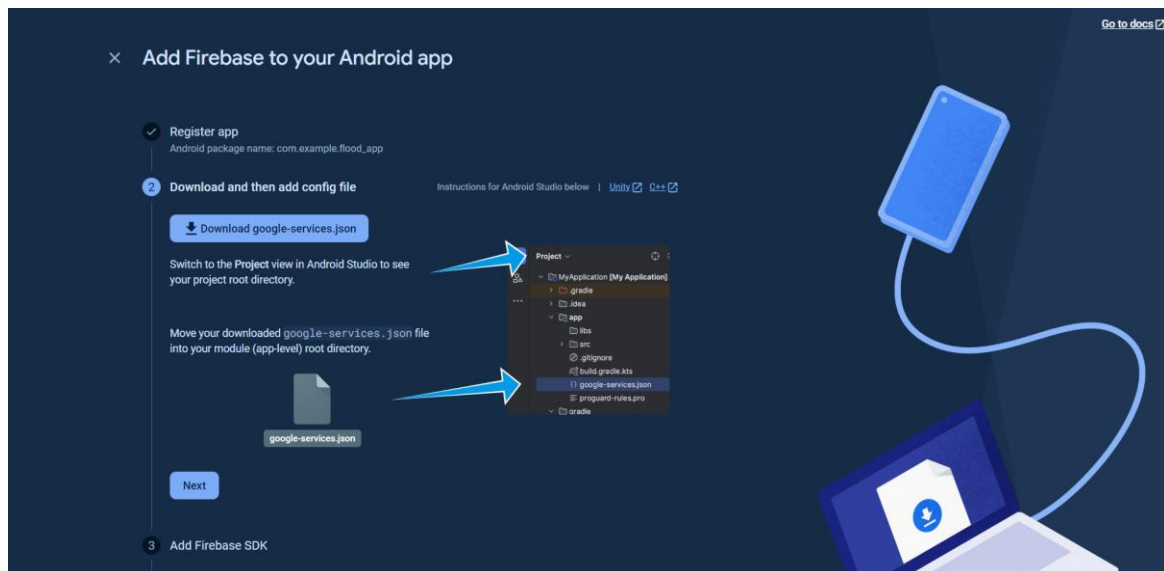


Figure 5.3.8: Download and Add Config File

Then, add the Firebase SDK to the app by adding the following dependencies to the build.gradle file (Figure 5.3.9). Finally, initialise Firebase in the app by adding the code to the main.dart file (Figure 5.3.10).

```
build.gradle x
android > build.gradle > buildscript
1  buildscript {
2      ext.kotlin_version = '1.9.21'
3      repositories {
4          google()
5          mavenCentral()
6      }
7  }
8  dependencies {
9      classpath 'com.android.tools.build:gradle:7.3.0'
10     classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
11     classpath "com.google.gms:google-services:4.3.15"
12 }
13 }
14
15 allprojects {
16     repositories {
17         google()
18         mavenCentral()
19     }
20 }
21
22 rootProject.buildDir = '../build'
23 subprojects {
24     project.buildDir = "${rootProject.buildDir}/${project.name}"
25 }
26 subprojects {
27     project.evaluationDependsOn(':app')
28 }
29
30 tasks.register("clean", Delete) {
31     delete rootProject.buildDir
32 }
```

Figure 5.3.9: Add Dependencies

```
Run | Debug | Profile
Future main() async {
    // Initialize Firebase in your app
    WidgetsFlutterBinding.ensureInitialized();
    await Firebase.initializeApp();
    runApp(const MyApp());
}
```

Figure 5.3.10: Initialise Firebase

A few setups must be completed to use Firebase Cloud Messaging to send push notifications. Firstly, the platform-specific configuration is handled by updating “AndroidManifest.xml” to include the intent filter (Figure 5.3.11). This enables us to execute code when a notification is clicked from the device tray. Next, create a function called “_configureFirebaseMessaging” to handle incoming FCM messages in the background and the foreground (Figure 5.3.12). As shown in Figure 5.3.13, three callbacks, such as “onMessage,” “onMessageOpenedApp,” and “onBackgroundMessage,” are written in the function. The “onMessage” event occurs when an application is open and running in the foreground, whereas the “onMessageOpenedApp” event occurs when an application is closed but remains active in the background. In addition, the “onBackgroundMessage” event is triggered if the application is completely terminated [12].

```

<manifest xmlns:android="http://schemas.android.com/apk/res/android">
  <application
    <activity
      android:windowSoftInputMode="adjustResize">
        <!-- Specifies an Android theme to apply to this Activity as soon as
            the Android process has started. This theme is visible to the user
            while the Flutter UI initializes. After that, this theme continues
            to determine the Window background behind the Flutter UI. -->
        <meta-data
          android:name="io.flutter.embedding.android.NormalTheme"
          android:resource="@style/NormalTheme"
        />
        <intent-filter>
          <action android:name="android.intent.action.MAIN"/>
          <category android:name="android.intent.category.LAUNCHER"/>
        </intent-filter>
        <intent-filter>
          <action android:name="FLUTTER_NOTIFICATION_CLICK"/>
          <category android:name="android.intent.category.DEFAULT"/>
        </intent-filter>
      </activity>
    </application>
  </manifest>

```

Figure 5.3.11: Add Intent Filter

```

Run | Debug | Profile
Future main() async {
  // Initialize Firebase in your app
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  _configureFirebaseMessaging();
  runApp(const MyApp());
}

```

Figure 5.3.12: Handle Incoming Messages

```

void _configureFirebaseMessaging() {
  FirebaseMessaging.onMessage.listen((RemoteMessage message) {
    // Handle received message
    print('Got a message whilst in the foreground!');
    print('Message data: ${message.data}');

    if (message.notification != null) {
      print('Message also contained a notification: ${message.notification}');
    }
    showAlertToasts(message: message.notification!.title!);
  });

  // Handle message when app is in background and opened by tapping on the notification
  FirebaseMessaging.onMessageOpenedApp.listen((RemoteMessage message) {
    if (message.notification != null) {
      print('Message also contained a notification: ${message.notification}');
    }
  });

  // Handle message when app is in background or terminated
  FirebaseMessaging.onBackgroundMessage(_firebaseMessagingBackgroundHandler);
}

Future<void> _firebaseMessagingBackgroundHandler(RemoteMessage message) async {
  print("Handling a background message: ${message.messageId}");
}

```

Figure 5.3.13: Handle Incoming Messages (Cont.)

5.4 System Operation (with Screenshot)

5.4.1 Authentication Module

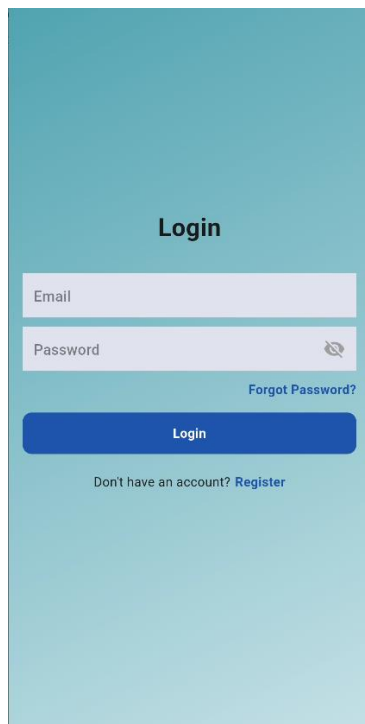


Figure 5.4.1: Login Screen

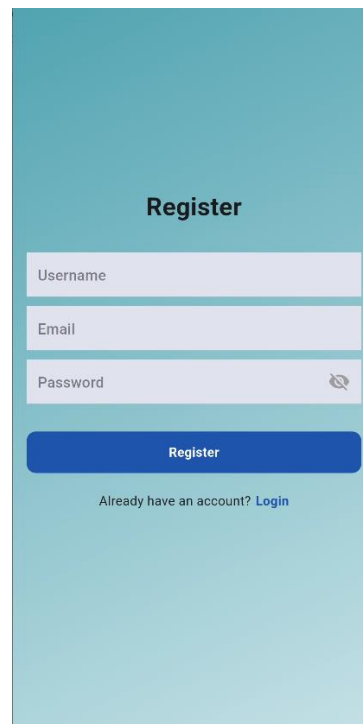


Figure 5.4.2: Register Screen

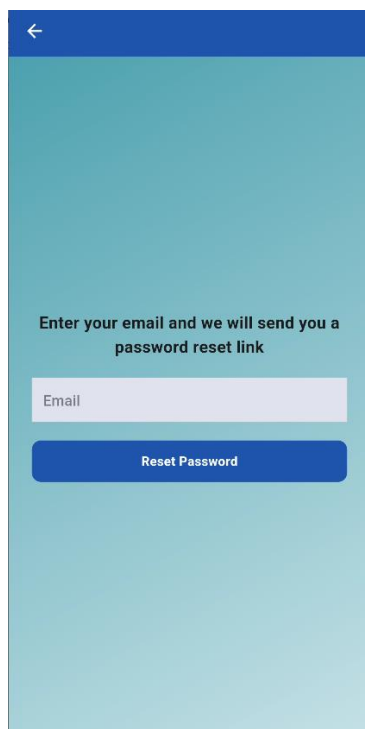


Figure 5.4.3: Reset Password Screen

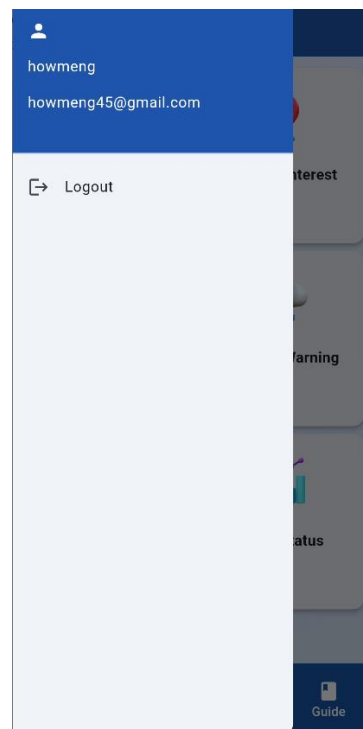


Figure 5.4.4: Logout Screen

The authentication module plays an essential role in ensuring that only authorised users are permitted to access the system's features. First of all, the users fill in the provided text fields with their email address and password (Figure 5.4.1). Firebase Authentication will be used by the system to verify the credentials after they are submitted. Subsequently, new users can create an account on the register screen by entering the required details, including a username, email address, and password (Figure 5.4.2). The system validates the data entered to ensure that a unique email address is used for the registration process. The user account is created, and their information is safely stored in the Firestore Database if the validation is successful.

Furthermore, the reset password screen enables users to reset their passwords if they forget them (Figure 5.4.3). When a user inserts the email address linked to their account, the system checks to see if the email is indeed kept in the database. If the email is valid, the system emails the user with a password reset link. After clicking the reset link, the user is redirected to a page where they can change their password. The system modifies the user's password and shows a confirmation message when the new password is submitted. Lastly, the users simply click the logout button after pressing the menu button on the upper left corner of the screen to sign out of the system (Figure 5.4.4). The system will redirect the users to the login screen, and any following attempts to access restricted pages will require them to log in again.

5.4.2 Monitoring Module



Figure 5.4.5: Home Screen



Figure 5.4.6: Monitoring Screen

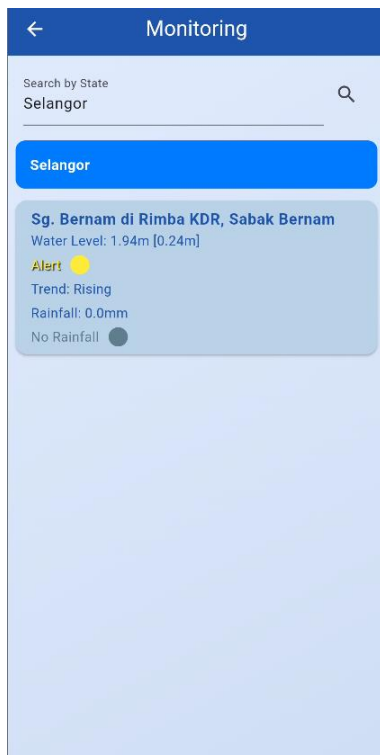


Figure 5.4.7: Search State Results



Figure 5.4.8: Station Details Screen

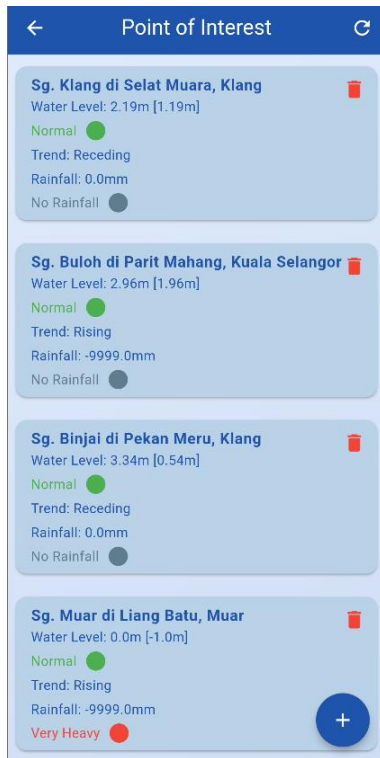


Figure 5.4.9: POI Screen



Figure 5.4.10: Add POI Screen

The monitoring module consists of various functionalities, allowing users to remain informed about monitoring station status, weather forecasts, weather warnings, and flood conditions. This module integrates with external data sources and APIs to improve the precision and timeliness of information delivery. Firstly, the monitoring stations screen displays real-time data for monitoring stations located across the nation. The users can reach the screen from the main dashboard on the home screen (Figure 5.4.5). The system will provide a list of monitoring stations together with information on the current water level, water level indicator, water level trend, current rainfall, and rainfall indicator (Figure 5.4.6). Users can also filter stations by using the search box to search for the station's state (Figure 5.4.7). Besides that, the users can click on the cards to get more specific details about the station, such as the main basin and the latest update timestamp (Figure 5.4.8). Next, the users can instantly view their POIs by navigating to the POI screen (Figure 5.4.9). The system also enables users to add a new POI or remove an existing POI. Users can add a POI by selecting the state, district, type of station, and particular station, and then clicking the add button (Figure 5.4.10). After adding or removing POI, users can update the list by clicking the reload button in the upper right area of the application.

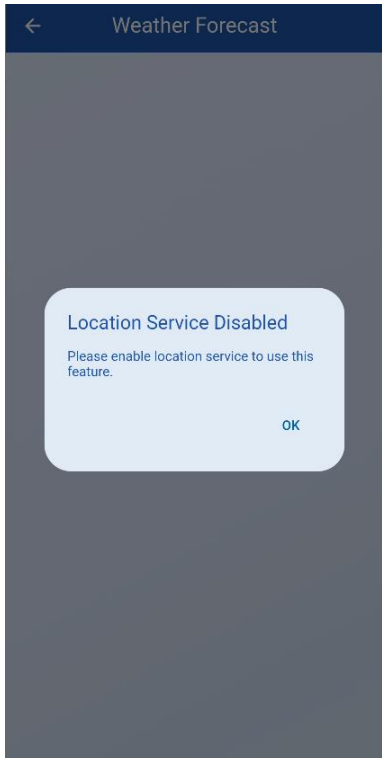


Figure 5.4.11: Weather Forecast Screen (1/3)



Figure 5.4.12: Weather Forecast Screen (2/3)



Figure 5.4.13: Weather Forecast Screen (3/3)

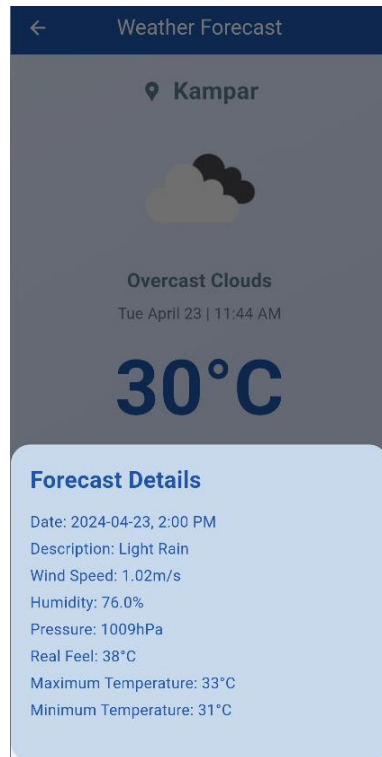


Figure 5.4.14: Forecast Details Screen



Figure 5.4.15: Weather Warning Screen

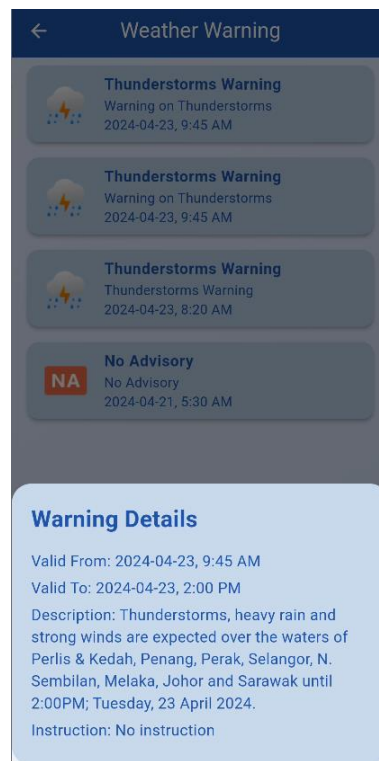


Figure 5.4.16: Warning Details Screen

In the weather forecasting screen, users are prompted to grant permission for location access, ensuring that the system can accurately locate their position (Figure 5.4.11). Weather forecasts are provided on the weather forecasting screen to help with preparedness and flood prediction. Based on the user's location, the system displays the current weather as well as 5-day weather forecasts (Figure 5.4.12). The users also have access to forecasts for several parameters, including weather description, wind speed, humidity, atmospheric pressure, and real-feel temperature (Figure 5.4.14). In addition, the weather warning screen shows warnings about extreme weather that could result in flooding (Figure 5.4.15). To ensure that users receive weather warnings from the Malaysian Meteorological Department (MET) in a timely manner, the system makes use of real-time data from the Weather API. Lastly, the users can view warning details such as the warning's interval, description, and instructions (Figure 5.4.16).

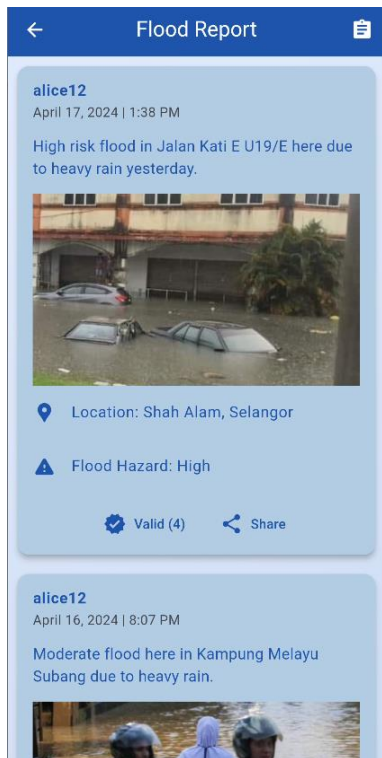


Figure 5.4.17: Flood Report Screen

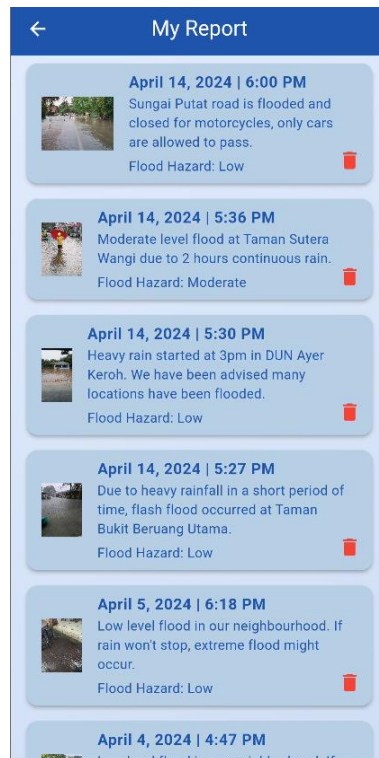


Figure 5.4.18: My Report Screen

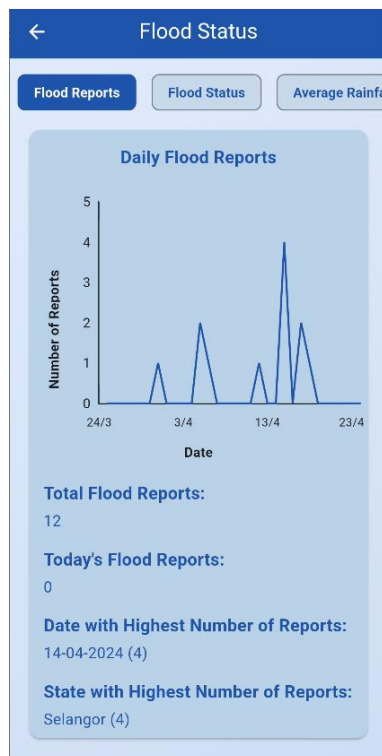


Figure 5.4.19: Flood Status Screen (1/6)

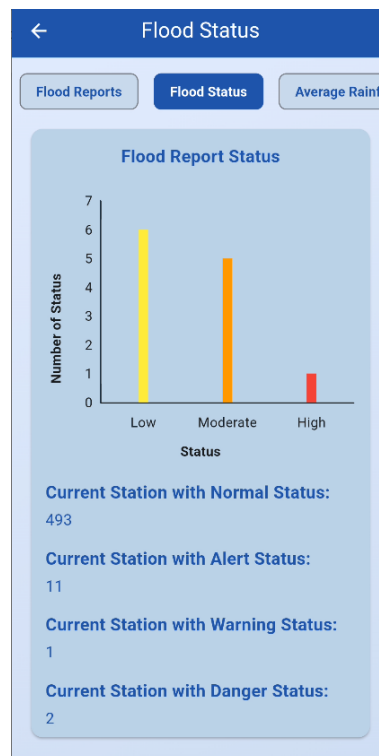


Figure 5.4.20: Flood Status Screen (2/6)

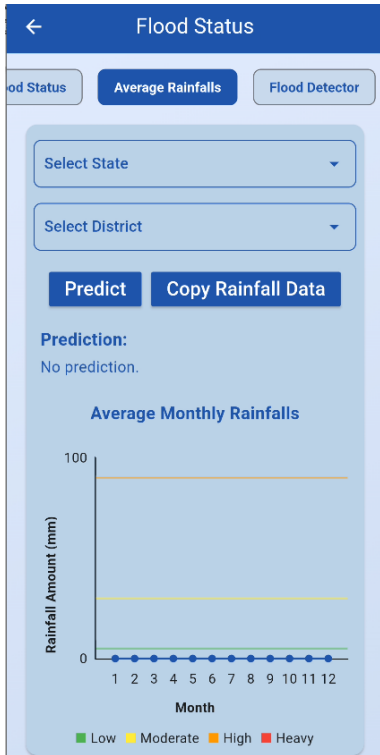


Figure 5.4.21: Flood Status Screen (3/6)

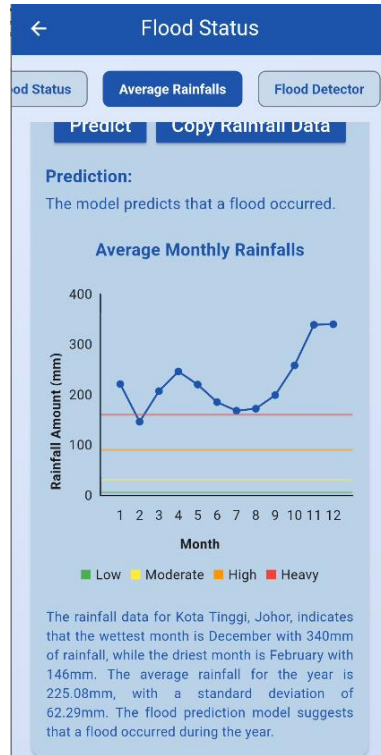


Figure 5.4.22: Flood Status Screen (4/6)

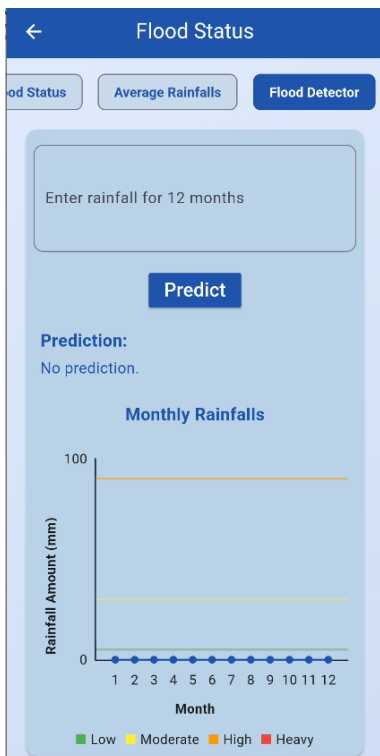


Figure 5.4.23: Flood Status Screen (5/6)

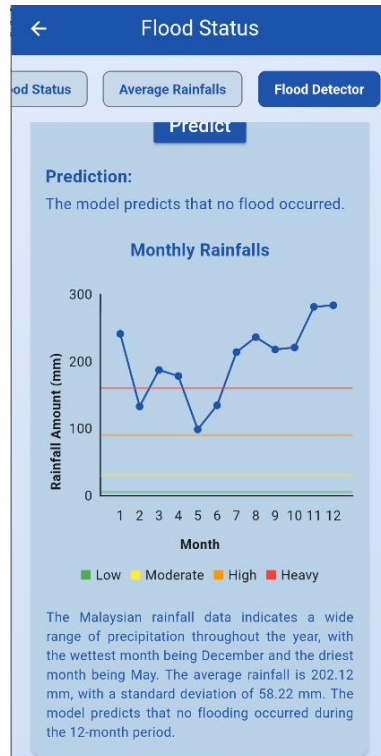


Figure 5.4.24: Flood Status Screen (6/6)

Aside from that, the flood reports screen is another component in the monitoring module that gives users access to both historical and current flood reports. As shown in Figure 5.4.17, the system displays a list of flood events, together with details about each one such as timestamp, description, image, location, and their severity. By clicking the “valid” button, users can validate reports, and the system will update the report’s validation count within the database. Moreover, users can directly share a particular report across other platforms by pressing the “share” button. The system also enables users to manage their reports by tapping the button located in the upper right corner of the system. Subsequently, users can review their reports and delete any reports if they wish to (Figure 5.4.18).

On the flood status screen, it provides a summary of the flood reports and flood situations using various visualisation charts. There are four options available for users to choose from: flood reports, flood status, average rainfalls, and flood detector. If the users select flood reports, the system will show the daily flood reports chart and some highlights regarding the reports (Figure 5.4.19). If the users choose flood status, the system displays the flood report status graph and the current water level status of the monitoring stations (Figure 5.4.20).

Next, the average rainfalls option allows users to have access to the average rainfalls for a specific district over the duration of a year (Figure 5.4.21). Additionally, the users can forecast if a flood will occur during the year using the average rainfall. On the other hand, the flood detector works similarly to the previous feature, enabling users to predict the likelihood of a flood by utilising rainfall data over a 12-month period. However, the difference between them is that the flood detector lets users enter and customise the amounts of rainfall (Figure 5.4.23). By providing the rainfall values, the system will then feed the input into the detector to generate the output score. Lastly, the system displays a monthly rainfall chart and insights produced by the generative AI known as Gemini (Figure 5.4.24).

5.4.3 Map Module

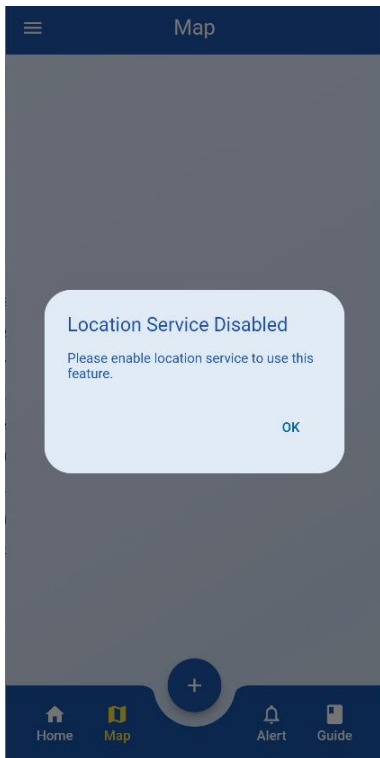


Figure 5.4.25: Location Service Disabled

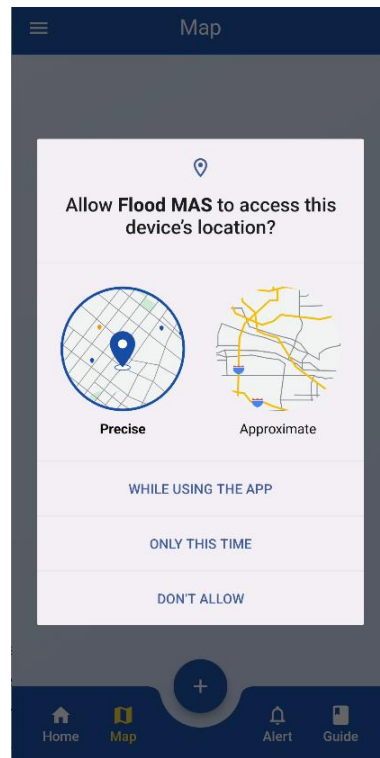


Figure 5.4.26: Access Device's Location

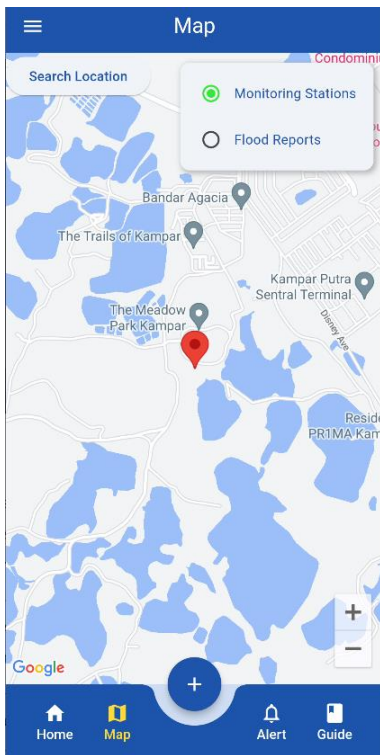


Figure 5.4.27: User's Current Location



Figure 5.4.28: Marker Clustering



Figure 5.4.29: Monitoring Station Location

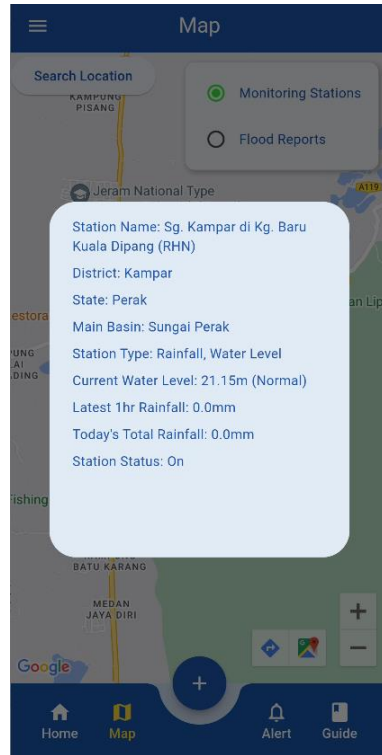


Figure 5.4.30: Station Details



Figure 5.4.31: Flood Report Location

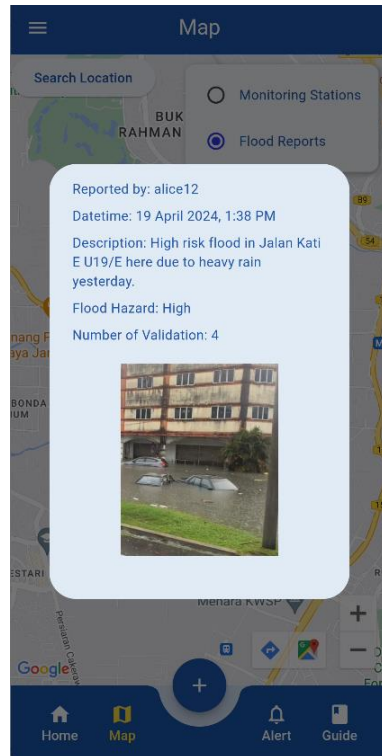


Figure 5.4.32: Report Details

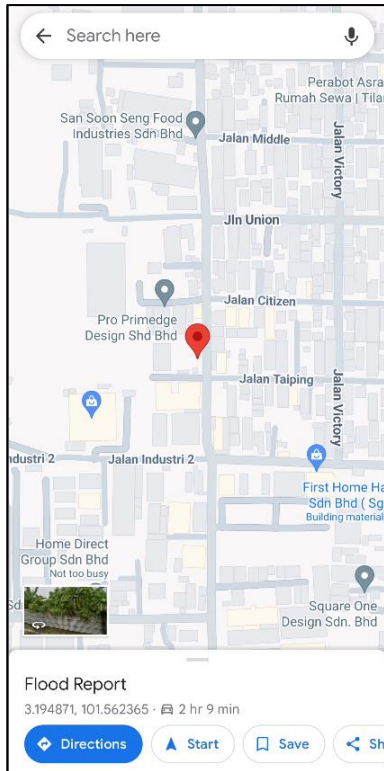


Figure 5.4.33: Navigation to Google Maps

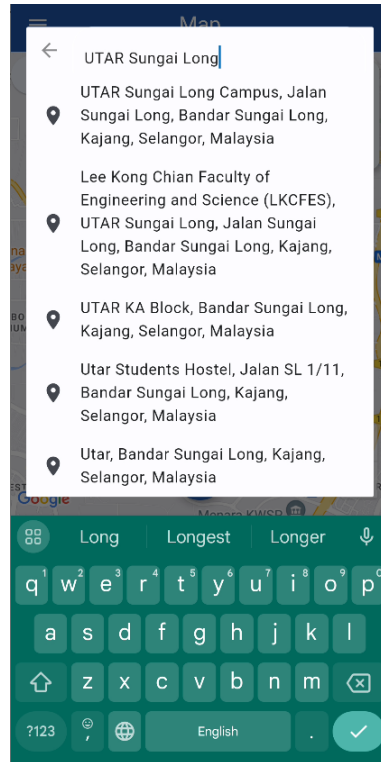


Figure 5.4.34: Search Location Function

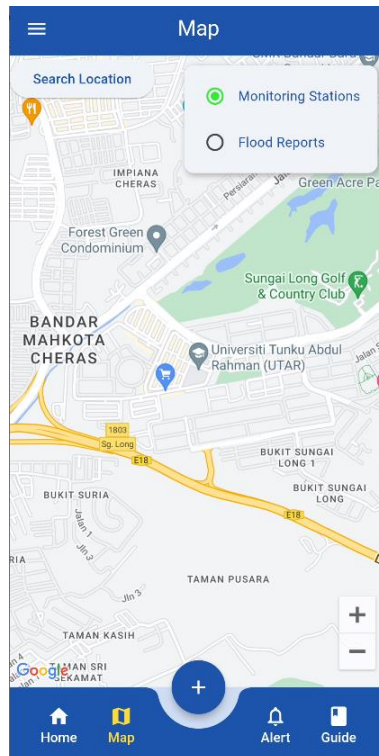


Figure 5.4.35: Search Location Results

In the map module, users are prompted to grant permission for location access, ensuring that the system can precisely locate and display their current geographical coordinates (Figure 5.4.26). Besides that, users can directly see their current location, providing them with a visual representation of their position relative to potential flood-prone areas and monitoring station location (Figure 5.4.27). As illustrated in Figure 5.4.28, marker clustering has been implemented to improve the visualisation of a huge number of markers in a limited geographic region. Marker clustering addresses this issue by grouping nearby markers into a single cluster marker. This cluster marker will display a number indicating the count of individual markers it represents.

Other than that, users can choose to see monitoring stations or flood reports by selecting the radio buttons located in the upper right corner of the screen. In accordance with the user's selection, the system will update the markers. To access more details, users can pan across the map and click on the marker that piques their interest. The green colour markers will display details about the monitoring station, whereas the blue colour markers will show information about the flood report. This makes it possible for users to quickly find out information about floods in their area. By clicking the navigation button on the lower right corner, users can seamlessly navigate to Google Maps based on the position of the selected marker (Figure 5.4.33). Lastly, users can also search for specific locations on the map to explore or locate specific points of interest (Figure 5.4.35).

5.4.4 Alerting Module

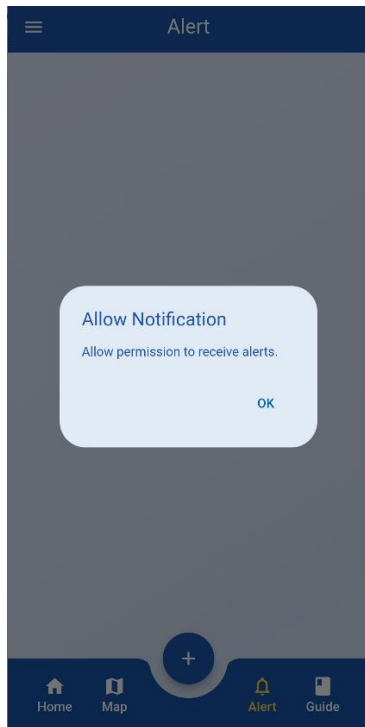


Figure 5.4.36: Notification Disabled

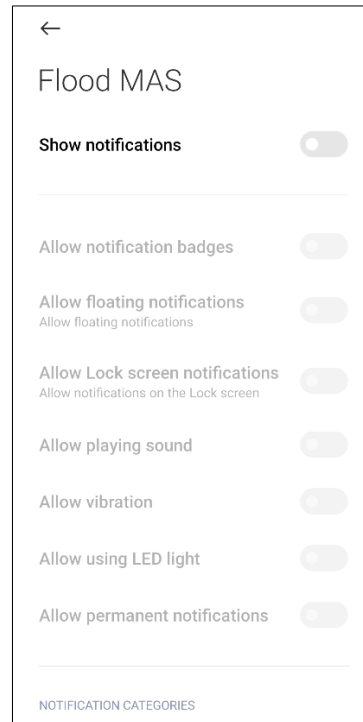


Figure 5.4.37: Allow Notification

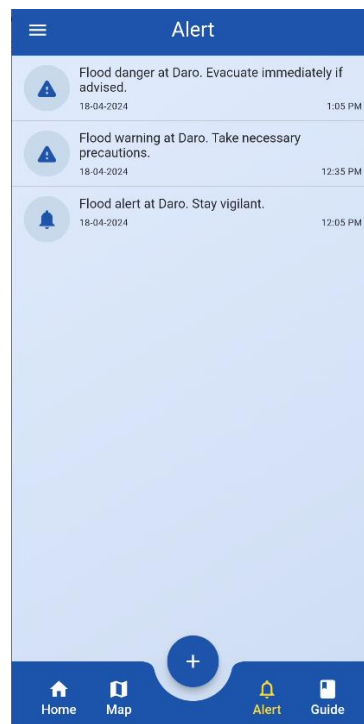


Figure 5.4.38: Alert Screen

In the alerting module, the users are prompted to grant permission for sending notifications if the permission is not allowed. The system displays a pop-up message to inform users to configure their notification settings (Figure 5.4.36). When the users click “OK”, the system directs them to the notification settings screen, where they can customise their notification preferences (Figure 5.4.37). If the permission is granted, the system stores the user’s device token in the database for the purpose of delivering messages. After that, the alert screen provides users with a list of historical alerts and warnings with information like the title, description, and timestamp (Figure 5.4.38). The users can review the past alerts to track the progression of events and response actions. As a result, this module facilitates the timely dissemination of critical information to users and enables an effective response to flood events.

For the backend part, the alerting module will deliver real-time alerts to users depending on their points of interest. A Cloud Function will be triggered to send messages to the user’s device if the water levels at their points of interest rise above a predefined threshold. The system will then handle the messages sent by the Cloud Function. Push notifications are used by the system to alert users when it is terminated or runs in the background. In contrast, while the system is in the foreground, Flutter Toast is used to notify the user with an alert that is sent within the app.

5.4.5 Flood Management Module

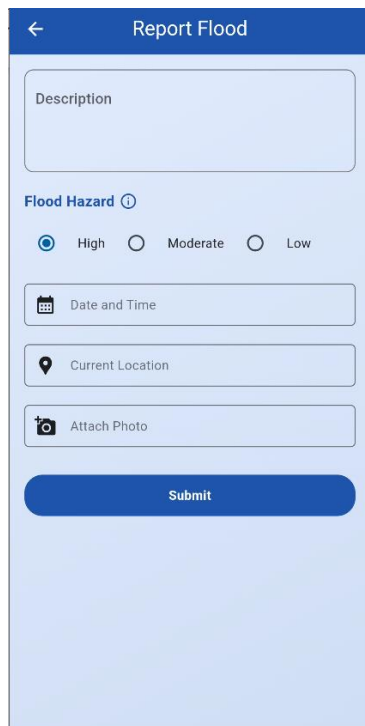


Figure 5.4.39: Report Flood Screen



Figure 5.4.40: Guide Screen

The flood management module is one of the most important parts of the system, which permits users to contribute to flood response operations by reporting incidents and accessing valuable information to improve their preparedness. The report flood screen allows users to report flood events in their region, which helps with real-time situational awareness and response coordination. As shown in Figure 5.4.39, users are prompted by the system to enter details about the flooding occurrence, including the description, severity level, timestamp, and location. The users may also upload images from their camera or gallery to supplement their reports. Once the report is submitted, the system verifies the reports and saves them in the database. Furthermore, the guide screen offers users guidance and useful advice on flood preparedness, response, and recovery. Users can access the flood tips screen from the main menu. Then, the system presents the flood safety information in the page views, enabling users to browse through and follow the safety guidelines (Figure 5.4.40). The emergency hotlines and tips for various flood management stages are covered on the screen.

5.5 Implementation Issues and Challenges

In this project, there are various concerns and challenges faced throughout the implementation phase. One of the implementation challenges encountered is the performance issues that arise from using Android Studio. The application experiences performance issues that slow down the entire system and frequently cause crashes when running the system, hindering efficient development and testing. These problems are caused by the specifications of the computer and the resource-intensive nature of Android Studio. To mitigate this, VS Code is used as an alternative coding environment, which is a more lightweight but powerful source code editor. However, VS Code's long emulator startup time presents another challenge. To address this issue, USB debugging is adopted as a solution to avoid emulator delays and test the application on a physical Android device. Therefore, this approach enhances the development experience by providing a more responsive and efficient testing environment.

Furthermore, another implementation issue faced is related to the occurrence of SSL certificate verification errors. These errors occur when attempting to fetch data from the Flood API, which interrupts the data retrieval processes. The SSL Certificate Verification Error is mainly caused by the failure of the system to verify the authenticity of the SSL certificate presented by the Flood API server. As a result, the inability to establish a secure connection with the server will significantly hamper the app's functionality to monitor flood conditions. One of the techniques to temporarily resolve this issue is to bypass or disable the SSL verification, but it is not recommended due to security risks such as data interception and man-in-the-middle attacks. On the other hand, one additional option is to manually obtain and configure trusted SSL certificates from the URL. However, this solution is only applicable to specific environments, and the difficulty still presents when deploying system-related programs on Google Cloud services. This is due to the fact that security and compliance are frequently given priority in the default configurations of cloud services, such as Cloud Functions and Compute Engines. Hence, there are restrictions on bypassing or configuring SSL certificates, which has made it more difficult to integrate the services into the project.

5.6 Concluding Remark

In summary, various modules are involved in the implementation of the flood monitoring and alert system, and each one is essential in giving users thorough flood-related information and enabling efficient response measures. First of all, the authentication module ensures secure access to the system's functionalities by using Firebase Authentication, allowing users to safely register, log in, and reset their passwords. Besides that, the monitoring module provides accurate and timely information on water levels, weather forecasts, and flood conditions by integrating real-time data from monitoring stations and weather APIs. Users can also personalise their monitoring experience by including POIs and gaining access to diverse visualisations for interpreting flood patterns and trends.

Furthermore, the map module improves users' spatial awareness by visualising monitoring stations, flood reports, and user positions. This enables users to instantly determine how close they are to possible flood-prone regions. Marker clustering is also used to enhance visibility when managing huge amounts of markers. By using Cloud Functions for backend processing, the alerting module makes sure that users receive critical information on time through real-time notifications that are triggered based on predefined thresholds. In addition, the flood management module allows users to contribute to flood response operations by reporting flood events with detailed information and photos. Users can also obtain helpful information, such as advice on flood preparation, response, and recovery, to enhance their preparedness.

Strategic solutions, such as using VS Code as an alternative development environment and manually implementing trusted SSL certificates, have been applied to overcome challenges like Android Studio performance issues and SSL certificate verification errors. Overall, the successful implementation and operation of the system show dedication to giving users a reliable and user-friendly platform for flood preparedness and response.

Chapter 6

System Evaluation and Discussion

This chapter offers a detailed assessment of the system’s performance, covering system testing and performance metrics, testing setup and results, and challenges encountered. It also emphasises the importance of the project’s objective and points out the system’s strengths, weaknesses, and possible areas for improvement.

6.1 System Testing and Performance Metrics

For the Flutter application, test cases are employed to verify the whole system’s operation and behaviour, ensuring that all components work together to achieve the intended results. For system testing, text cases are the user’s interactions with the application [13]. Every module contains unique test cases, along with the necessary test data and test steps to carry out the test case. Subsequently, the expected outcomes will be compared to the actual outcomes to determine the pass or fail status of the test case.

For flood detection, various performance metrics, including confusion matrix, precision, recall, F1-score, and accuracy score, are utilised to evaluate the model’s performance. Firstly, one of the metrics is the confusion matrix, which is a 2 x 2 matrix with four distinct combinations of predicted and actual values, such as True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). As shown in Table 6.1.1, the TP is the number of flood samples that are correctly predicted as flood, whereas the FP is the number of no flood samples that are falsely predicted as flood. On the other hand, the TN represents the number of no flood samples that are correctly predicted as no flood, while the FN represents the number of flood samples that are falsely predicted as no flood.

Confusion Matrix		Actual values	
		No Flood	Flood
Predicted values	No Flood	True Negatives (TN)	False Negatives (FN)
	Flood	False Positives (FP)	True Positives (TP)

Table 6.1.1: Confusion Matrix Table

Based on the confusion matrix, a few important metrics can be derived using the values of TP, TN, FP, and FN. The precision of a model is determined by the ratio of true positive predictions to the total number of positive predictions (Figure 6.1.1). It measures how accurate positive predictions are.

$$\textit{precision} = \frac{\textit{true positives}}{\textit{true positives} + \textit{false positives}}$$

Figure 6.1.1: Precision Score Formula

Next, the ratio of true positive predictions to the total number of actual positive cases in the data is known as recall (Figure 6.1.2). It assesses how well the model can recognise positive instances.

$$\textit{recall} = \frac{\textit{true positives}}{\textit{true positives} + \textit{false negatives}}$$

Figure 6.1.2: Recall Score Formula

The F1-score is the harmonic mean of precision and recall (Figure 6.1.3). When there is an imbalance between the classes, it might be useful for evaluating models because it offers a balance between precision and recall.

$$F_1 \textit{ score} = 2 \cdot \frac{\textit{precision} \cdot \textit{recall}}{\textit{precision} + \textit{recall}}$$

Figure 6.1.3: F1-Score Formula

To calculate the accuracy score, divide the number of accurately predicted instances by the total number of instances in the dataset (Figure 6.1.4). It gives a general indicator of the model's correctness.

$$\textit{accuracy} = \frac{\textit{correct predictions}}{\textit{all predictions}}$$

Figure 6.1.4: Accuracy Score Formula

6.2 Testing Setup and Result

6.2.1 Flutter Application

Authentication Module

Test Case ID	AUTHENTICATION-01
Test Case Scenario	Check Authentication Module Functionality
Test Case Description	Log In Check response on entering valid email and password. Register Check response on entering valid username, email, and password. Reset Password Check response on entering valid email address. Log Out Check response on clicking logout button.
Test Steps	Log In <ol style="list-style-type: none">1. Enter valid email and password.2. Click the “Login” button. Register <ol style="list-style-type: none">1. Enter valid username, email, and password.2. Click the “Register” button. Reset Password <ol style="list-style-type: none">1. Enter valid email address.2. Click the “Reset Password” button.3. Check the email inbox and click the password reset link.4. Enter new password and click the “Save” button. Log Out <ol style="list-style-type: none">1. Click the menu icon on the upper left corner of the app.2. Press the “Logout” button.
Test Data	A valid username, email address, and password
Expected Results	Log In Login should be successful and a confirmation message is displayed.

	<p>Register Register should be successful and a confirmation message is displayed.</p> <p>Reset Password Password reset link should be successfully sent and a confirmation message is displayed.</p> <p>Log Out Logout should be successful and a confirmation message is displayed.</p>
Actual Results	<p>Log In Login was successful and a confirmation message is displayed.</p> <p>Register Register was successful and a confirmation message is displayed.</p> <p>Reset Password Password reset link was successfully sent and a confirmation message is displayed.</p> <p>Log Out Logout was successful and a confirmation message is displayed.</p>
Pass / Fail	Pass

Table 6.2.1: Test Case for AUTHENTICATION-01

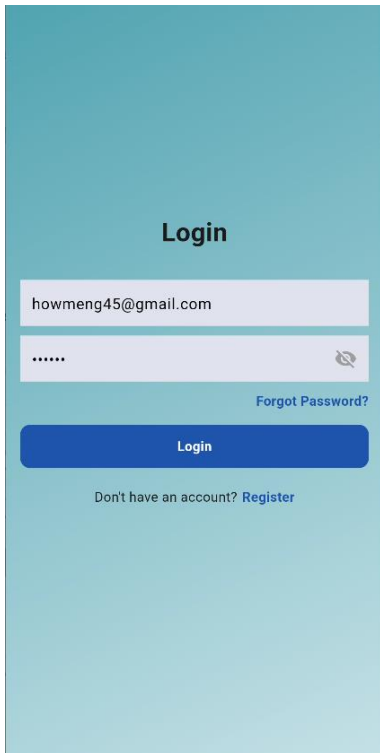


Figure 6.2.1: Valid Email and Password

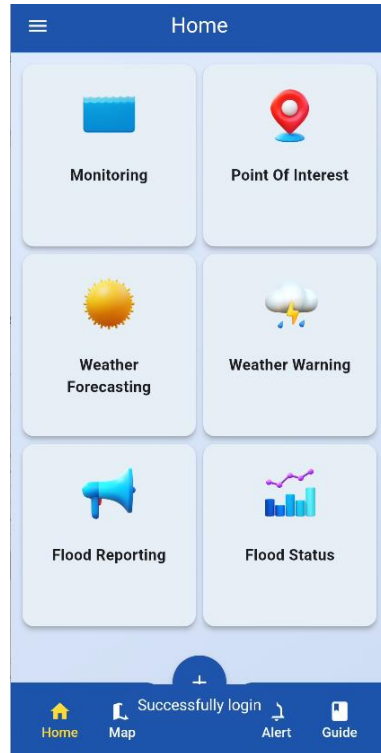


Figure 6.2.2: Successfully Login

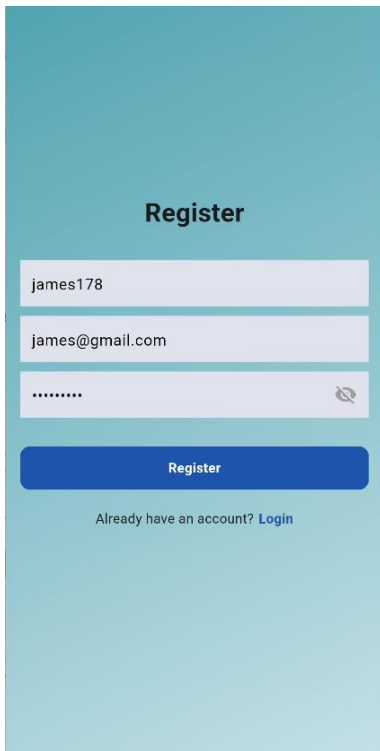


Figure 6.2.3: Valid User Information

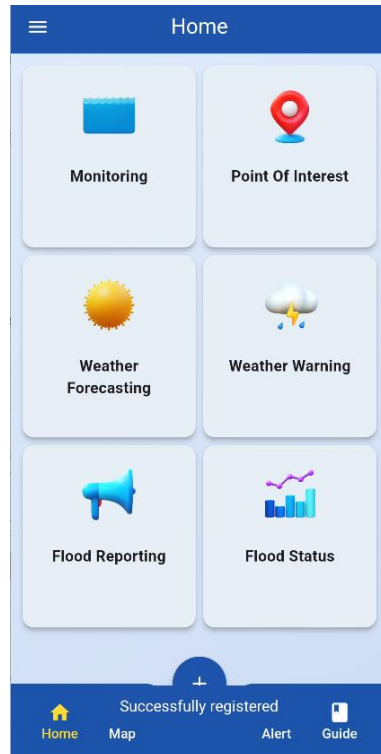


Figure 6.2.4: Successfully Registered

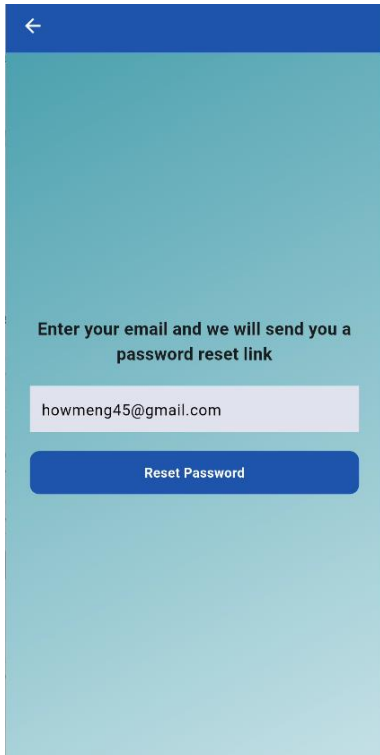


Figure 6.2.5: Valid Email Address

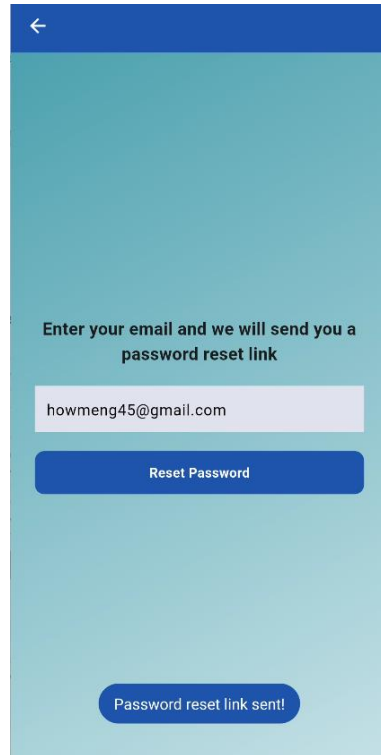


Figure 6.2.6: Successfully Sent

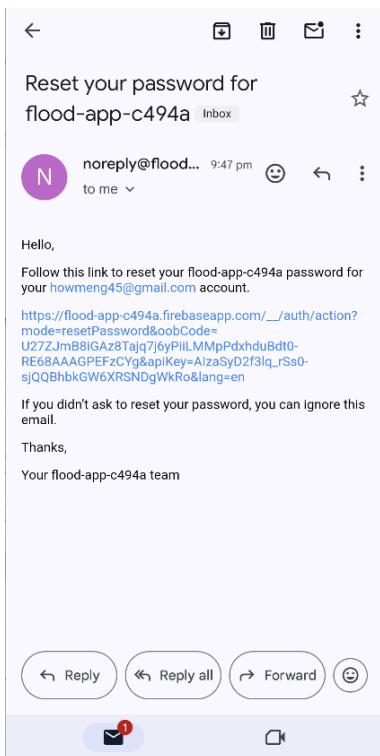


Figure 6.2.7: Password Reset Link

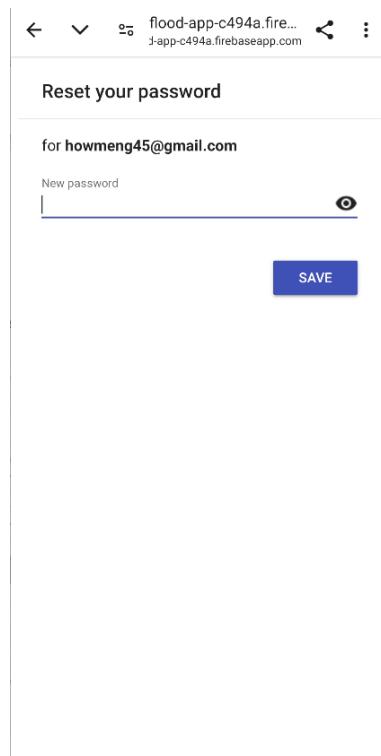


Figure 6.2.8: Enter New Password

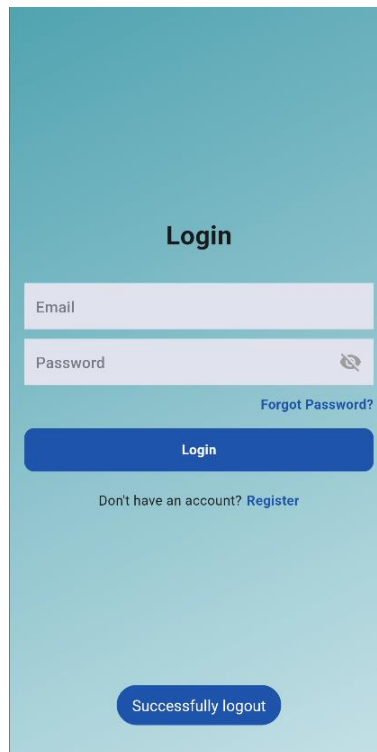


Figure 6.2.9: Successfully Logout

Test Case ID	AUTHENTICATION-02
Test Case Scenario	Check Authentication Module Functionality
Test Case Description	<p>Log In Check response on entering invalid email and password.</p> <p>Register Check response on entering invalid username, email, and password.</p> <p>Reset Password Check response on entering invalid email address.</p>
Test Steps	<p>Log In</p> <ol style="list-style-type: none"> 1. Enter invalid email and password. 2. Click the “Login” button. <p>Register</p> <ol style="list-style-type: none"> 1. Enter invalid username, email, and password. 2. Click the “Register” button. <p>Reset Password</p> <ol style="list-style-type: none"> 1. Enter invalid email address.

	2. Click the “Reset Password” button.
Test Data	An invalid username, email address, and password
Expected Results	<p>Log In Login should be unsuccessful and an error message is shown.</p> <p>Register Register should be unsuccessful and an error message is shown.</p> <p>Reset Password Password reset link should not be sent and an error message is shown.</p>
Actual Results	<p>Log In Login was unsuccessful and an error message is shown.</p> <p>Register Register was unsuccessful and an error message is shown.</p> <p>Reset Password Password reset link was not sent and an error message is shown.</p>
Pass / Fail	Pass

Table 6.2.2: Test Case for AUTHENTICATION-02

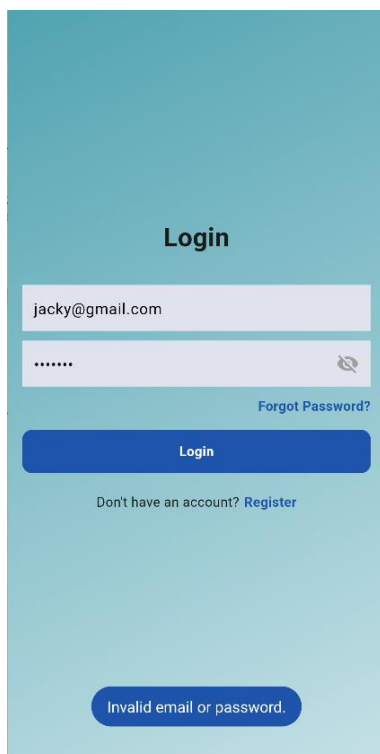


Figure 6.2.10: Invalid Email or Password

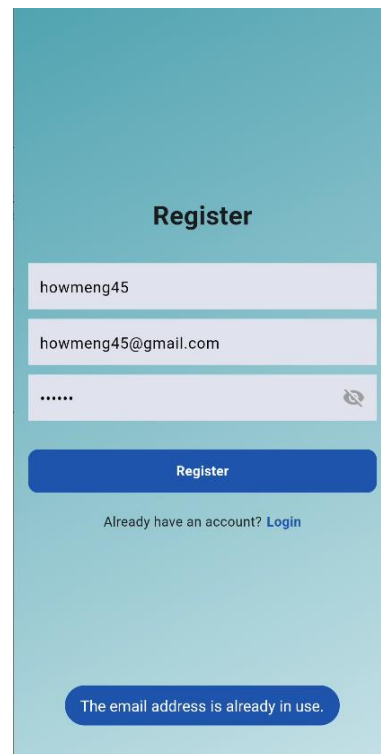


Figure 6.2.11: Invalid User Information

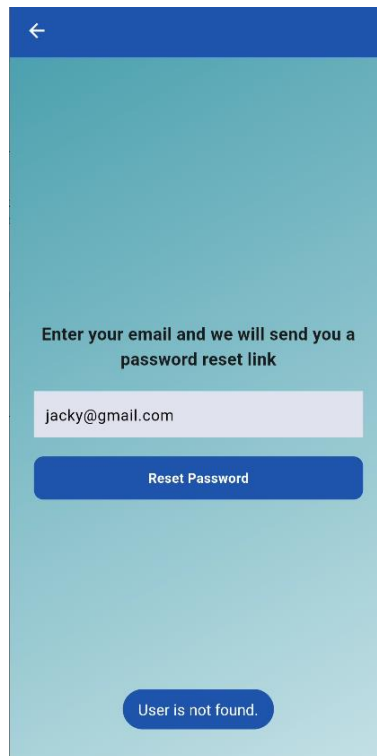


Figure 6.2.12: Invalid Email Address

Test Case ID	AUTHENTICATION-03
Test Case Scenario	Check Authentication Module Functionality
Test Case Description	<p>Log In Check response when email or password is empty and login button is pressed.</p> <p>Register Check response when username, email, or password is empty and register button is pressed.</p> <p>Reset Password Check response when email address is empty and reset password button is pressed.</p>
Test Steps	<p>Log In Click the “Login” button.</p> <p>Register Click the “Register” button.</p> <p>Reset Password Click the “Reset Password” button.</p>

Test Data	None
Expected Results	<p>Log In Login should be unsuccessful and an error message is shown.</p> <p>Register Register should be unsuccessful and an error message is shown.</p> <p>Reset Password Reset link should not be sent and an error message is shown.</p>
Actual Results	<p>Log In Login was unsuccessful and an error message is shown.</p> <p>Register Register was unsuccessful and an error message is shown.</p> <p>Reset Password Reset link was not sent and an error message is shown.</p>
Pass / Fail	Pass

Table 6.2.3: Test Case for AUTHENTICATION-03

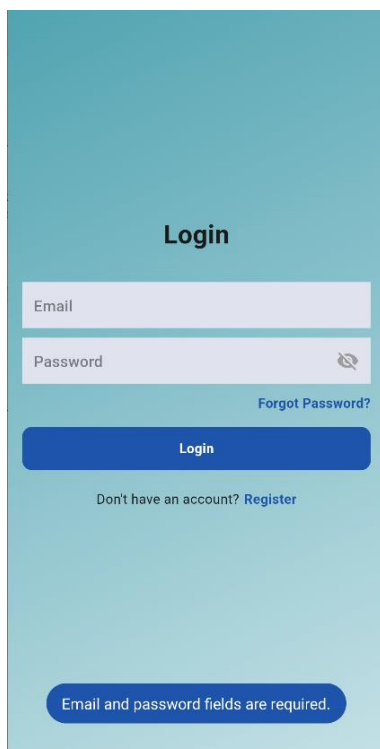


Figure 6.2.13: Empty Email and Password

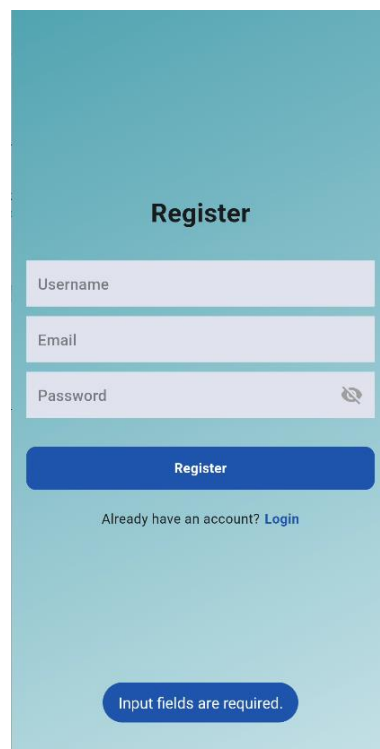


Figure 6.2.14: Empty Input Fields

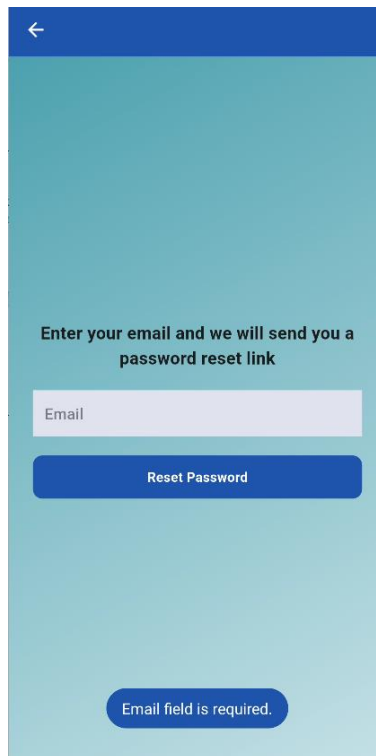


Figure 6.2.15: Empty Email Field

Monitoring Module

Test Case ID	MONITORING-01
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response when state name is empty and search icon button is pressed.
Test Steps	Click the search icon button.
Test Data	None
Expected Results	Search should be unsuccessful and an error message is shown.
Actual Results	Search was unsuccessful and an error message is shown.
Pass / Fail	Pass

Table 6.2.4: Test Case for MONITORING-01



Figure 6.2.16: Empty State Name

Test Case ID	MONITORING-02
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response on selecting a state, district, station type, and POI and add button is pressed.
Test Steps	<ol style="list-style-type: none"> 1. Select state from dropdown list. 2. Select district from dropdown list. 3. Select station type from dropdown list. 4. Select POI from dropdown list. 5. Click the “Add” button. 6. Click the reload icon on the upper right corner of the app.
Test Data	State, district, station type, and POI
Expected Results	POI should be successfully added and a confirmation message is displayed.
Actual Results	POI was successfully added and a confirmation message is displayed.
Pass / Fail	Pass

Table 6.2.5: Test Case for MONITORING-02



Figure 6.2.17: Valid Selection

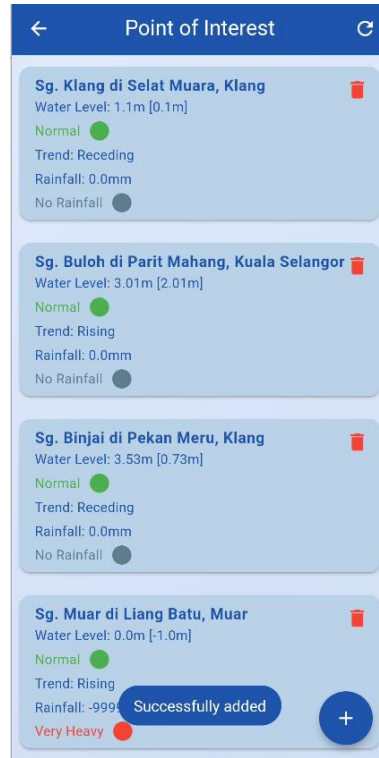


Figure 6.2.18: Successfully Added



Figure 6.2.19: Reload Page

Test Case ID	MONITORING-03
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response on selecting the same state, district, station type, and POI and add button is pressed.
Test Steps	<ol style="list-style-type: none"> 1. Select same state from dropdown list. 2. Select same district from dropdown list. 3. Select same station type from dropdown list. 4. Select same POI from dropdown list. 5. Click the “Add” button.
Test Data	State, district, station type, and POI
Expected Results	POI should not be added and an error message is shown.
Actual Results	POI was not added and an error message is shown.
Pass / Fail	Pass

Table 6.2.6: Test Case for MONITORING-03



Figure 6.2.20: POI is Already Added

Test Case ID	MONITORING-04
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response when no state, district, station type, and POI are selected and add button is pressed.
Test Steps	Click the “Add” button.
Test Data	None
Expected Results	POI should not be added and an error message is shown.
Actual Results	POI was not added and an error message is shown.
Pass / Fail	Pass

Table 6.2.7: Test Case for MONITORING-04

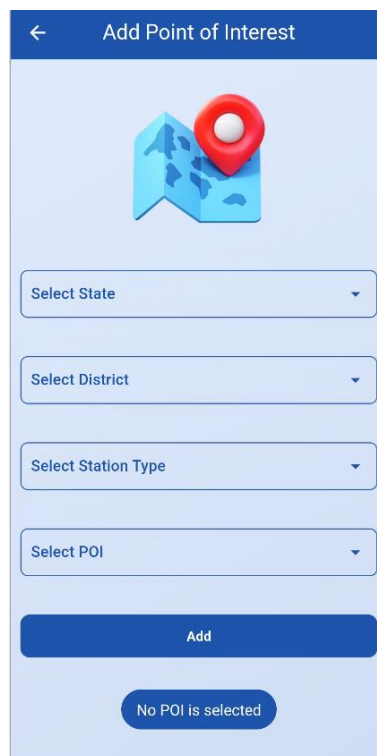


Figure 6.2.21: No POI is Selected

Test Case ID	MONITORING-05
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response on clicking the delete icon button.
Test Steps	<ol style="list-style-type: none"> 1. Click the delete icon button. 2. Click the “Delete” button. 3. Click the reload icon on the upper right corner of the app.
Test Data	None
Expected Results	POI should be successfully deleted and a confirmation message is displayed.
Actual Results	POI was successfully deleted and a confirmation message is displayed.
Pass / Fail	Pass

Table 6.2.8: Test Case for MONITORING-05

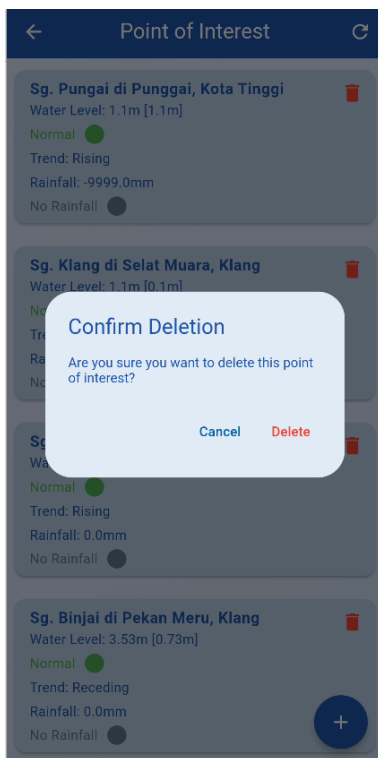


Figure 6.2.22: Confirm POI Deletion

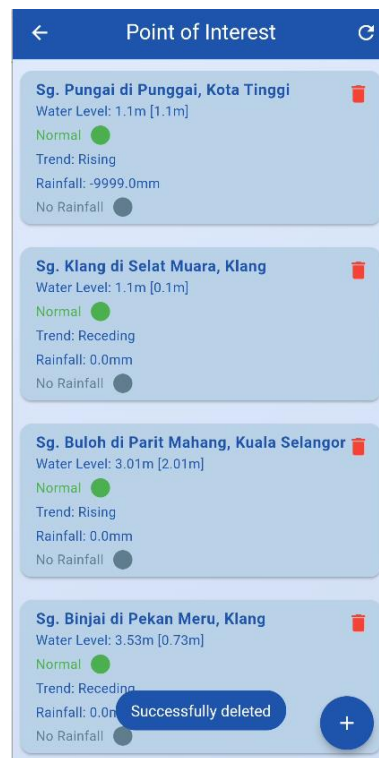


Figure 6.2.23: Successfully Deleted



Figure 6.2.24: Reload Page

Test Case ID	MONITORING-06
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response on clicking the valid button.
Test Steps	Click the “Valid” button.
Test Data	None
Expected Results	Validation count should be successfully updated.
Actual Results	Validation count was successfully updated.
Pass / Fail	Pass

Table 6.2.9: Test Case for MONITORING-06

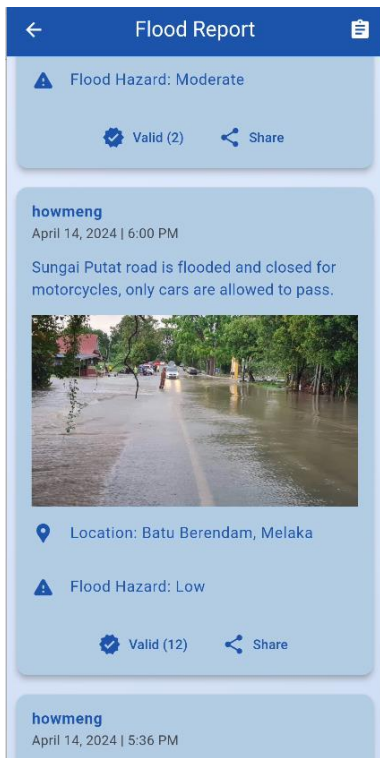


Figure 6.2.25: Before Clicking Valid

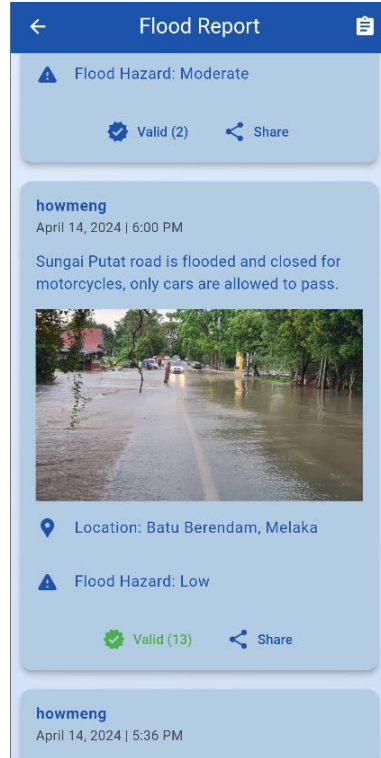


Figure 6.2.26: After Clicking Valid

Test Case ID	MONITORING-07
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response on clicking the share button.
Test Steps	<ol style="list-style-type: none"> 1. Click the “Share” button. 2. Select the platform to share.
Test Data	None
Expected Results	User should be successfully navigated to the platform with flood report details.
Actual Results	User was successfully navigated to the platform with flood report details.
Pass / Fail	Pass

Table 6.2.10: Test Case for MONITORING-07

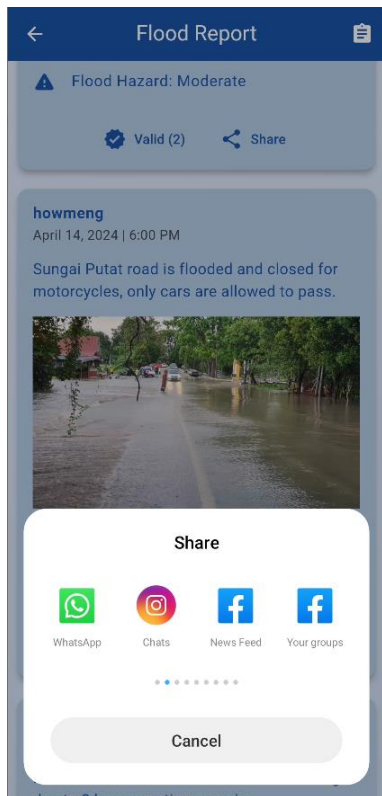


Figure 6.2.27: Select Platform

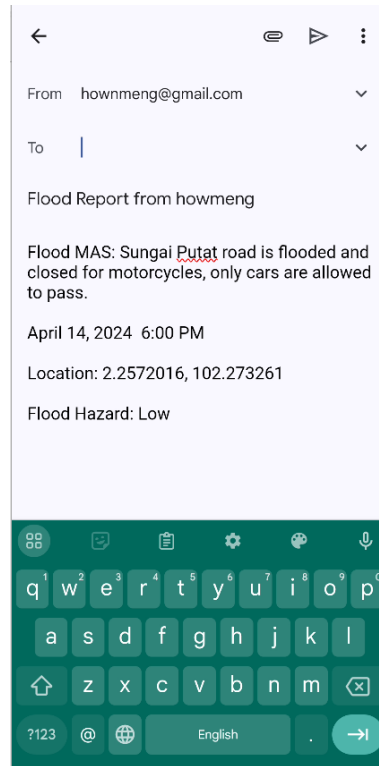


Figure 6.2.28: Navigate to Platform

Test Case ID	MONITORING-08
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response on clicking the delete icon button.
Test Steps	<ol style="list-style-type: none"> 1. Click the delete icon button. 2. Click the “Delete” button.
Test Data	None
Expected Results	Report should be successfully deleted and a confirmation message is displayed.
Actual Results	Report was successfully deleted and a confirmation message is displayed.
Pass / Fail	Pass

Table 6.2.11: Test Case for MONITORING-08

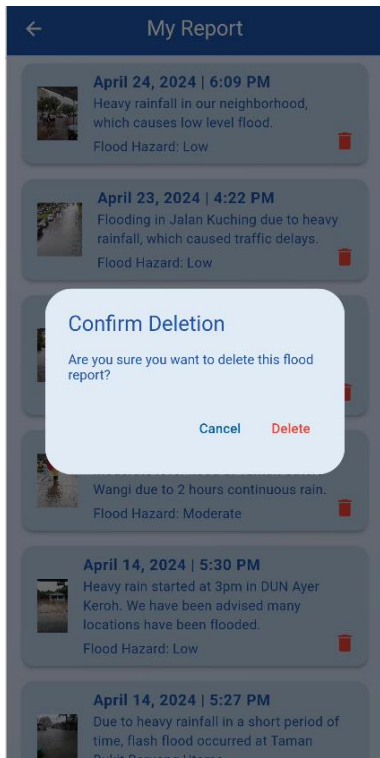


Figure 6.2.29: Confirm Report Deletion

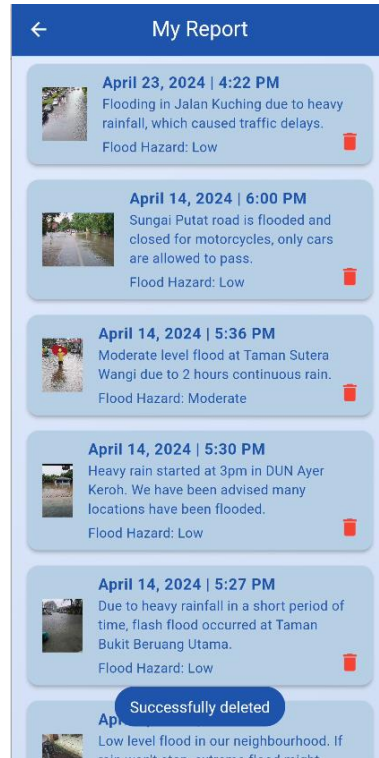


Figure 6.2.30: Successfully Deleted

Test Case ID	MONITORING-09
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response when no state and district are selected and two buttons are pressed.
Test Steps	<ol style="list-style-type: none"> 1. Click the “Predict” button. 2. Click the “Copy Rainfall Data” button.
Test Data	None
Expected Results	Prediction should be unsuccessful and an error message is shown. Rainfall data should not be copied and an error message is shown.
Actual Results	Prediction was unsuccessful and an error message is shown. Rainfall data was not copied and an error message is shown.
Pass / Fail	Pass

Table 6.2.12: Test Case for MONITORING-09

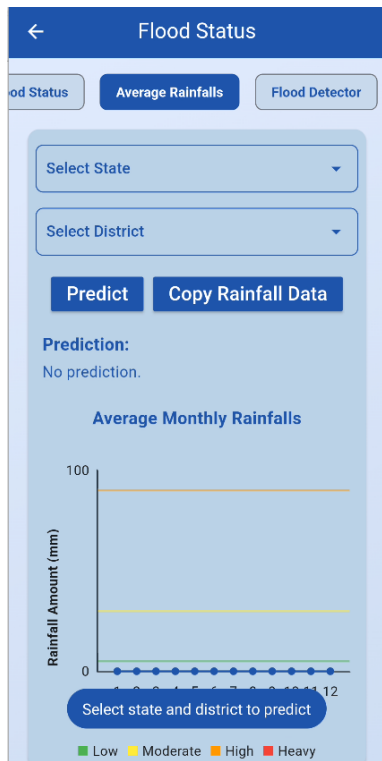


Figure 6.2.31: Failed to Predict

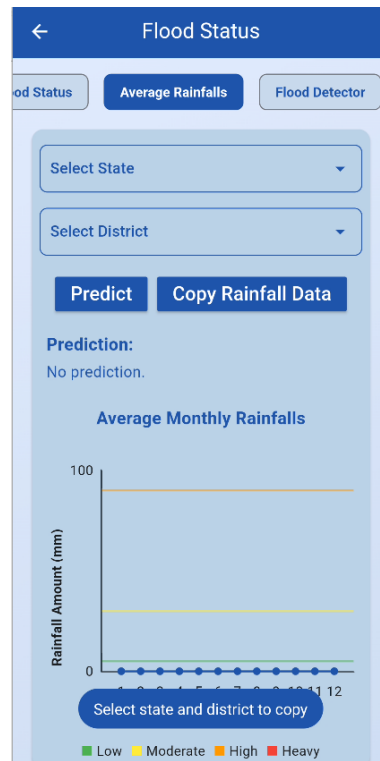


Figure 6.2.32: Failed to Copy

Test Case ID	MONITORING-10
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response on entering invalid rainfall data and predict button is pressed.
Test Steps	<ol style="list-style-type: none"> 1. Enter invalid rainfall data. 2. Click the “Predict” button.
Test Data	Invalid rainfall data
Expected Results	Prediction should be unsuccessful and an error message is shown.
Actual Results	Prediction was unsuccessful and an error message is shown.
Pass / Fail	Pass

Table 6.2.13: Test Case for MONITORING-10

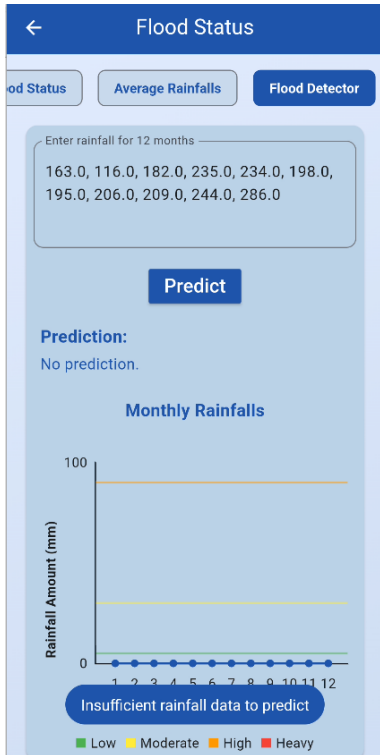


Figure 6.2.33: Insufficient Rainfall Data

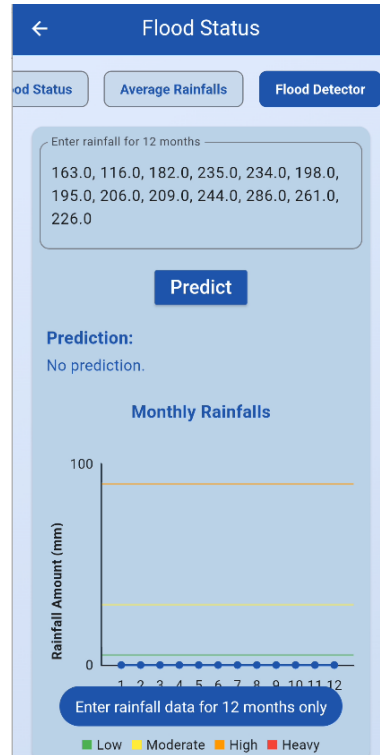


Figure 6.2.34: Invalid Rainfall Data

Test Case ID	MONITORING-11
Test Case Scenario	Check Monitoring Module Functionality
Test Case Description	Check response when the rainfall is empty and predict button is pressed.
Test Steps	Click the “Predict” button.
Test Data	None
Expected Results	Prediction should be unsuccessful and an error message is shown.
Actual Results	Prediction was unsuccessful and an error message is shown.
Pass / Fail	Pass

Table 6.2.14: Test Case for MONITORING-11

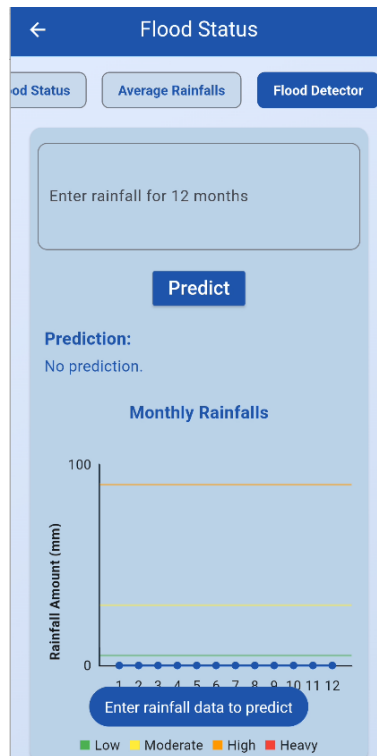


Figure 6.2.35: Empty Rainfall Data

Flood Management Module

Test Case ID	FLOODMANAGEMENT-01
Test Case Scenario	Check Flood Management Module Functionality
Test Case Description	Check response on entering valid description, flood hazard, date time, location, and photo.
Test Steps	<ol style="list-style-type: none"> 1. Enter valid description. 2. Select flood hazard. 3. Select date and time. 4. Click the location field. 5. Attach photo from camera or gallery. 6. Click the “Submit” button.
Test Data	A valid description, flood hazard, date time, location, and photo
Expected Results	Report should be successfully submitted and a confirmation message is displayed.
Actual Results	Report was successfully submitted and a confirmation message is displayed.

Pass / Fail	Pass
--------------------	------

Table 6.2.15: Test Case for FLOODMANAGEMENT-01

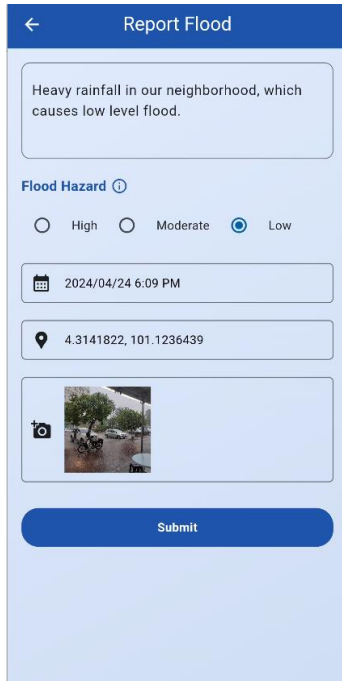


Figure 6.2.36: Valid Report Details

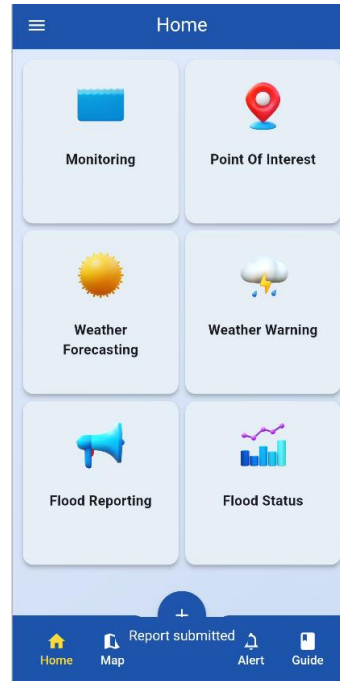


Figure 6.2.37: Successfully Submitted

Test Case ID	FLOODMANAGEMENT-02
Test Case Scenario	Check Flood Management Module Functionality
Test Case Description	Check response when the description, date time, location, and photo is empty and submit button is pressed.
Test Steps	Click the “Submit” button.
Test Data	None
Expected Results	Report should not be submitted and an error message is shown.
Actual Results	Report was not submitted and an error message is shown.
Pass / Fail	Pass

Table 6.2.16: Test Case for FLOODMANAGEMENT-02

Figure 6.2.38 : Empty Report Details

6.2.2 Flood Detection

Hyperparameter Tuning

Parameters	Model 1 (Base)	Model 2	Model 3	Model 4
batch_size	10	15	10	15
epochs	50	100	100	100
optimiser	adam	rmsprop	rmsprop	adam

Table 6.2.17: Hyperparameter Tuning

To evaluate the flood detection model, performance metrics are essential for identifying its strengths and weaknesses. A high precision shows that the model is good at correctly detecting flood instances, whereas a high recall suggests that the model effectively captures the majority of the actual flood events. These two measures are combined into a single value, the F1-score, considering both precision and recall simultaneously. Since model 4 has the best performance among the three tuning models, it is chosen and assessed based on the performance metrics. As shown in Tables 6.2.18 and 6.2.19, the confusion matrices of the base model and tuned model are constructed.

Confusion Matrix		Actual values		
		No Flood	Flood	Total
Predicted values	No Flood	34	15	49
	Flood	14	20	34
	Total	48	35	83

Table 6.2.18: Confusion Matrix of the Base Model

Confusion Matrix		Actual values		
		No Flood	Flood	Total
Predicted values	No Flood	37	11	48
	Flood	11	24	35
	Total	48	35	83

Table 6.2.19: Confusion Matrix of the Tuned Model

Evaluation Results

Model	Validation Accuracy	Precision	Recall	F1-Score
Model 1 (Before tuning)	0.6506	0.5882	0.5714	0.5797
Model 4 (After tuning)	0.7349	0.6857	0.6857	0.6857

Table 6.2.20: Evaluation Results of the Models

As shown in Table 6.2.20, there is an obvious improvement in validation accuracy after tuning the model, indicating that the fine-tuned model performs better overall in accurately detecting flood instances. Besides that, Model 4 demonstrated better precision, suggesting that a larger percentage of predicted flood instances are correct. Similarly, there is an increase in recall after fine-tuning, showing that the model is better at capturing actual flood instances. In addition, the F1-score significantly increases, indicating an overall improvement in the model's performance. To sum up, Model 4 performs better than Model 1 on every evaluation criteria, which demonstrated that the tuning process has successfully enhanced the model's ability to detect floods effectively.

Testing Results

Five sample test sets were utilised to test the tuned model in order to assess its performance on unseen data. According to the testing results, 4 out of 5 times the tuned model correctly predicted the outcomes (Table 6.2.21). This suggested that the model has a high success rate in accurately detecting floods when updated with new and unseen data. In this testing scenario, the model demonstrated that it could generalise well beyond the training data, with an accuracy rate of 80%.

	1 st Test	2 nd Test	3 rd Test	4 th Test	5 th Test
Actual	Flood	Flood	No Flood	Flood	No Flood
Predicted	Flood	Flood	Flood	Flood	No Flood

Table 6.2.21: Testing Results

```
[ ] example_data = X_test[0].reshape(1, -1)
prediction = FloodDetector.predict(example_data)[0][0]

if prediction >= 0.5:
    print("The model predicts that a flood occurred.")
else:
    print("The model predicts that no flood occurred.")

1/1 [=====] - 0s 19ms/step
The model predicts that a flood occurred.
```

Figure 6.2.39: First Test Result

```
[ ] example_data = X_test[1].reshape(1, -1)
prediction = FloodDetector.predict(example_data)[0][0]

if prediction >= 0.5:
    print("The model predicts that a flood occurred.")
else:
    print("The model predicts that no flood occurred.")

1/1 [=====] - 0s 17ms/step
The model predicts that a flood occurred.
```

Figure 6.2.40: Second Test Result

```
[ ] example_data = X_test[2].reshape(1, -1)
prediction = FloodDetector.predict(example_data)[0][0]

if prediction >= 0.5:
    print("The model predicts that a flood occurred.")
else:
    print("The model predicts that no flood occurred.")

1/1 [=====] - 0s 18ms/step
The model predicts that a flood occurred.
```

Figure 6.2.41: Third Test Result

```
[ ] example_data = X_test[3].reshape(1, -1)
prediction = FloodDetector.predict(example_data)[0][0]

if prediction >= 0.5:
    print("The model predicts that a flood occurred.")
else:
    print("The model predicts that no flood occurred.")

1/1 [=====] - 0s 19ms/step
The model predicts that a flood occurred.
```

Figure 6.2.42: Fourth Test Result

```
[ ] example_data = X_test[4].reshape(1, -1)
prediction = FloodDetector.predict(example_data)[0][0]

if prediction >= 0.5:
    print("The model predicts that a flood occurred.")
else:
    print("The model predicts that no flood occurred.")

1/1 [=====] - 0s 20ms/step
The model predicts that no flood occurred.
```

Figure 6.2.43: Fifth Test Result

6.3 Project Challenges

One of the critical challenges in this project is the availability of data, particularly concerning relief centre locations. Relief centres serve as crucial hubs for aiding and providing support to affected communities during flood occurrences. For individuals and families affected by floods, assistance may include shelter, food, medical assistance, and other necessary services. The inability to access the Relief Centre API has hindered the integration of relief centre locations into the map module, affecting the effectiveness of the system in providing timely support and assistance to affected communities. Besides that, the absence of relief centre locations makes it difficult for victims to organise their evacuation because they do not know where to go for safety and support.

Another project's challenge is the appropriateness of the dataset used for flood detection. The dataset's limitations, such as its temporal scope (from 2000 to 2010) and sample size (825 samples), cause significant obstacles to the development and evaluation of accurate flood detection models. Moreover, the dataset's exclusive dependence on rainfall data for flood prediction overlooks additional factors such as topography and infrastructure, which may have an effect on the model's reliability and effectiveness. Additionally, the use of 12-month rainfall data to predict the likelihood of flooding may not be appropriate in real-world situations, as stakeholders frequently need more urgent and actionable insights on flood hazards.

Lastly, a significant challenge in this project is determining precise rainfall thresholds that trigger flood alerts. Rainfall thresholds can differ significantly depending on various factors, including geographic location, terrain, infrastructure, and land use. Therefore, collaboration with local authorities or experts, such as government agencies, meteorological departments, or hydrology experts, is essential to addressing this situation. These stakeholders often possess valuable knowledge and guidelines regarding rainfall thresholds that are known to cause floods in specific regions. By working with these experts, they can provide relevant data and insights that will help improve and validate the rainfall thresholds for the flood monitoring and alert system.

6.4 Objectives Evaluation

The primary objective of this project is to develop a flood monitoring and alert system to monitor and notify people about possible floods, lessening the impacts of floods on communities and infrastructure. To achieve the main goal, various sub-objectives must be fulfilled. First of all, the system has successfully integrated flood warning and weather APIs, providing precise and up-to-date information on water levels, rainfall, weather forecasts, and weather warnings for monitoring flood risk levels and status across different regions of Malaysia. Secondly, the application can effectively facilitate the dissemination of flood alerts to the public through push notifications, which enhances public awareness and preparedness. Furthermore, this system successfully delivers flood safety tips and incorporates geospatial data to provide users with geospatial visualisation, which emphasises potential flood zones and helps with resource allocation and evacuation preparation. Last but not least, the project successfully applied deep learning techniques, particularly ANN, to detect floods based on rainfall data.

6.5 Concluding Remark

In summary, the evaluation of the flood monitoring and alert system highlights both its accomplishments and the difficulties faced throughout its development. The Flutter application successfully passed each test case through system testing. Furthermore, the performance of the flood detection model is considered acceptable for effectively detecting floods. However, the project also encounters critical challenges, especially the availability and appropriateness of data. The absence of complete relief centre data hinders the system's ability to provide support and evacuation guidance during flood incidents. Similarly, limitations in the dataset, particularly its temporal scope and reliance solely on rainfall data, affect the reliability of flood detection. Overall, the flood monitoring and alert system successfully accomplishes its objectives by providing accurate, real-time information, facilitating timely dissemination of alerts, integrating geospatial visualisation for decision support, and implementing advanced deep learning techniques for flood detection. These achievements help to enhance community resilience and mitigate the impacts of floods on both livelihoods and infrastructure.

Chapter 7

Conclusion and Recommendation

7.1 Conclusion

In conclusion, the mobile-based flood monitoring and alert system proposed in this project addresses the critical issues related to flash floods in Malaysia. A comprehensive solution was developed in response to the identified problems, such as lack of timely information, limited early warning systems, and challenges in managing flood events. The motivation behind the project was to emphasise the urgency of implementing a digital technology innovation to mitigate the devastating impact of floods on communities and infrastructure. The proposed architecture involves a user-friendly mobile application with modules for monitoring, mapping, alerting, and flood management. The use of Firebase Authentication simplifies user access, and the integration of FCM ensures real-time alert dissemination. The system's innovation lies in its seamless integration of diverse data sources using external APIs to enhance the accuracy and effectiveness of flood monitoring. Another novel aspect of the project is the integration of geospatial data to offer users geospatial visualisations. This enables users to visualise locations that are vulnerable to flooding and nearby flood reports to make informed decisions about evacuation. The focus on a user-centric approach to alert dissemination through a mobile application ensures that people and communities receive information that is relevant to their specific points of interest. The application of deep learning techniques to detect floods is also an innovative approach that allows users to examine the relationship between precipitation and flood occurrence. By providing tools for exploration and visualisation, such as graphs and statistical analysis, users can gain insights about how rainfall patterns influence the likelihood of flood disasters. Therefore, the project aims to contribute significantly to better flood management and disaster response in Malaysia, providing a practical and low-cost solution to reduce the effect of flash floods on communities.

7.2 Recommendation

In this project, a few crucial areas for the flood monitoring and alert system's future development and enhancement are highlighted. For future development, the monitoring module will be improved to include additional information such as flood coverage area, population affected, and evacuation plans. This information is critical for decision-making processes for authorities, emergency responders, and affected communities. Furthermore, exploring better-quality historical data for deep learning could enhance the performance and reliability of the predictive model for more precise flood forecasts. For the alert module, future work may involve refining the alerting mechanism to allow users to set personalised thresholds for receiving warnings based on specific flood severity levels. Moreover, integration of computer vision using image processing techniques can be implemented in the future to detect floods based on river images captured by IP cameras. Next, the flood management module could be expanded to include interactive features, such as educational videos and an integrated chatbot feature within the application. Last but not least, developing functionality for managing and distributing aid would enhance its allocation and coordination during emergency response operations. This feature will enable the public to distribute essential supplies more effectively to impacted populations.

REFERENCES

- [1] H. Kamel. “Govt Encourages Public To Download MyPublicInfoBanjir App For Flood Related Updates And Warnings.” Lowyat.net. <https://www.lowyat.net/2021/229356/govt-encourages-public-to-download-mypublicinfobanjir-app-for-flood-related-updates-and-warnings/> (accessed Nov. 4, 2023).
- [2] SERVIR Hindu Kush Himalaya. “Preparing for the 2019 floods in Bangladesh.” server.icimod.org. <https://servir.icimod.org/news/preparing-2019-floods-bangladesh/> (accessed Nov. 4, 2023).
- [3] SOBOS. “FloodAlert Water Levels and Warning.” earlyfloodalert.com. <https://earlyfloodalert.com/en/> (accessed Nov. 4, 2023).
- [4] HKV. “My Flood Risk Accra App launched in Ghana.” hkv.nl. <https://www.hkv.nl/en/news/my-flood-risk-accra-app-launched-in-ghana/> (accessed Nov. 4, 2023).
- [5] Adservio. “What is Flutter and Its Advantages?” adservio.fr. <https://www.adservio.fr/post/what-is-flutter-and-what-are-its-advantages> (accessed Nov. 11, 2023).
- [6] K. T. Hanna and L. Rosencrance. “What is Google Firebase?” techtarget.com. <https://www.techtargget.com/searchmobilecomputing/definition/Google-Firebase> (accessed Nov. 11, 2023).
- [7] A. Ming. “Everything about TensorFlow Lite and start deploying your machine learning model.” seedstudio.com. <https://www.seedstudio.com/blog/2022/05/08/everything-about-tensorflow-lite-and-start-deploying-your-machine-learning-model/> (accessed Nov. 11, 2023).
- [8] Y. Hutabarat. “Agile in a Nutshell.” Medium. <https://medium.com/@yafonia/agile-in-a-nutshell-7725674ee31e> (accessed Sep. 8, 2023).
- [9] National Centre for Environment Information, Jan. 2024, “Flood Dataset (Malaysia),” National Centre for Environment Information, National Oceanic and Atmospheric Administration. [Online]. Available: [https://datasets.omdena.com/dataset/flood-dataset-\(malaysia\)](https://datasets.omdena.com/dataset/flood-dataset-(malaysia))

- [10] TechTarget. “What is Android Studio?” techtarget.com. <https://www.techtarget.com/searchmobilecomputing/definition/Android-Studio> (accessed Dec. 3, 2023).
- [11] M. Heller. “What is Visual Studio Code? Microsoft’s extensible code editor.” infoworld.com. <https://www.infoworld.com/article/3666488/what-is-visual-studio-code-microsofts-extensible-code-editor.html> (accessed Dec. 3, 2023).
- [12] J. Delaney. “FCM Push Notifications For Flutter.” fireship.io. <https://fireship.io/lessons/flutter-push-notifications-fcm-guide/> (accessed Apr. 22, 2024).
- [13] T. Hamilton. “What is Use Case Testing?” guru99.com. <https://www.guru99.com/use-case-testing.html> (accessed Apr. 22, 2024).

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 3
Student Name & ID: Kok How Meng 20ACB02193	
Supervisor: Ts Tan Teik Boon	
Project Title: Flood Monitoring and Alert System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Modify and complete introduction.
- Modify and complete literature review.

2. WORK TO BE DONE

- Improving map module.
- Start developing the monitoring module.

3. PROBLEMS ENCOUNTERED

- Availability of the relief centre API and Flood Warning API.
- Requires some time to debug compatibility issues for coding environments.

4. SELF EVALUATION OF THE PROGRESS

- The problems encountered are manageable.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 5
Student Name & ID: Kok How Meng 20ACB02193	
Supervisor: Ts Tan Teik Boon	
Project Title: Flood Monitoring and Alert System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Utilised alternative solution to retrieve the monitoring stations' details.
- Improved the map module.

2. WORK TO BE DONE

- Perform water level and rainfall data collection.
- Continue developing the monitoring module.

3. PROBLEMS ENCOUNTERED

- Availability of historical data to perform data analysis.

4. SELF EVALUATION OF THE PROGRESS

- Monitoring module may take longer time than expected.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 7
Student Name & ID: Kok How Meng 20ACB02193	
Supervisor: Ts Tan Teik Boon	
Project Title: Flood Monitoring and Alert System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Finish developing monitoring page.
- Finish developing point of interest page.
- Finish developing weather forecast page.
- Collect water level and rainfall data for two consecutive weeks.

2. WORK TO BE DONE

- Start developing the alerting module.
- Start developing the flood management module.
- Continue developing the monitoring module.

3. PROBLEMS ENCOUNTERED

- No problem at this stage.

4. SELF EVALUATION OF THE PROGRESS

- Progress is on track.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 9
Student Name & ID: Kok How Meng 20ACB02193	
Supervisor: Ts Tan Teik Boon	
Project Title: Flood Monitoring and Alert System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Finish developing the alerting module.
- Finish developing weather warning page.

2. WORK TO BE DONE

- Continue developing the report flood page.
- Start developing flood reporting page.
- Start developing flood status page.

3. PROBLEMS ENCOUNTERED

- No problem at this stage.

4. SELF EVALUATION OF THE PROGRESS

- Progress is on track.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 11
Student Name & ID: Kok How Meng 20ACB02193	
Supervisor: Ts Tan Teik Boon	
Project Title: Flood Monitoring and Alert System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Finish developing the report flood page.
- Finish developing flood reporting page.
- Finish developing flood status page.

2. WORK TO BE DONE

- Start developing the flood tips page.
- Start performing system testing.
- Start completing system methodology and system design.

3. PROBLEMS ENCOUNTERED

- No problem at this stage.

4. SELF EVALUATION OF THE PROGRESS

- Progress in on track.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3T3	Study week no.: 13
Student Name & ID: Kok How Meng 20ACB02193	
Supervisor: Ts Tan Teik Boon	
Project Title: Flood Monitoring and Alert System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Finish developing the flood management module.
- System testing is performed.
- Complete system methodology and system design.
- Modify and improve the monitoring module.

2. WORK TO BE DONE

- Complete other chapters in the report.
- Design poster.
- Submit to Turnitin and ready for submission.

3. PROBLEMS ENCOUNTERED

- No problem at this stage.

4. SELF EVALUATION OF THE PROGRESS

- Progress is on track.



Supervisor's signature



Student's signature

POSTER



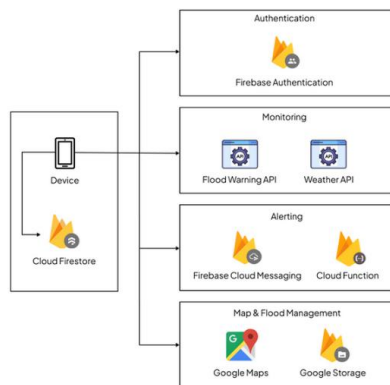
Faculty of Information and Communication Technology

FLOOD MONITORING AND ALERT SYSTEM

Project Developer: Kok How Meng
Project Supervisor: Ts Tan Teik Boon

INTRODUCTION

Floods have been a serious threat in Malaysia recently, especially in urban regions. The previous floods were horrible; many were stuck in flooded houses waiting for help to come, unsure of where to go, and lack of real-time information about floods. The need for an effective flood monitoring and alert system is critical to mitigate the impact of such events. This project aims to develop a robust system to monitor flood conditions and disseminate timely alerts to users.



METHODS

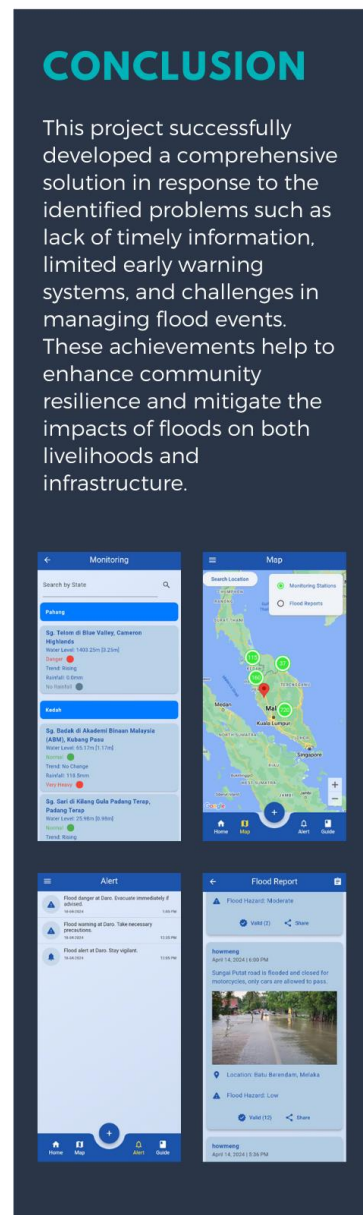
The software development approach implemented in this project is Agile methodology. The application is developed using Android Studio and Flutter. It utilises diverse data sources from external APIs, Google Firebase, and Google Cloud services for the modules.

DISCUSSION

- The monitoring module provides accurate and timely information on water levels, rainfalls, weather forecasts, weather warnings, and flood reports.
- The map module improves users' spatial awareness by visualising monitoring stations, flood reports, and user positions.
- The alerting module makes sure that users receive critical information on time through real-time notifications that are triggered based on predefined thresholds.
- The flood management module allows users to obtain flood tips and report flood-related events with detailed information and photos.

CONCLUSION

This project successfully developed a comprehensive solution in response to the identified problems such as lack of timely information, limited early warning systems, and challenges in managing flood events. These achievements help to enhance community resilience and mitigate the impacts of floods on both livelihoods and infrastructure.



PLAGIARISM CHECK RESULT

Flood Monitoring and Alert System

ORIGINALITY REPORT

4%	3%	1%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	eprints.bournemouth.ac.uk Internet Source	<1%
2	www.nwmo.ca Internet Source	<1%
3	www.ijraset.com Internet Source	<1%
4	eprints.utar.edu.my Internet Source	<1%
5	Submitted to Universiti Tunku Abdul Rahman Student Paper	<1%
6	Jim McBee. "Mastering Microsoft® Exchange Server 2007 SP1", Wiley, 2009 Publication	<1%
7	Submitted to Liverpool John Moores University Student Paper	<1%
8	scholarworks.aub.edu.lb Internet Source	<1%
9	www2.mdpi.com	

	Internet Source	<1 %
10	www.frontiersin.org Internet Source	<1 %
11	link.springer.com Internet Source	<1 %
12	img1.wsimg.com Internet Source	<1 %
13	Submitted to Washington University of Science and Technology Student Paper	<1 %
14	publisher.tbsnews.net Internet Source	<1 %
15	www.jma.go.jp Internet Source	<1 %
16	Submitted to Marquette University Student Paper	<1 %
17	Maged Elaasar, Nicolas Rouquette, David Wagner, Bentley James Oakes, Abdelwahab Hamou-Lhadj, Mohammad Hamdaqa. "openCAESAR: Balancing Agility and Rigor in Model-Based Systems Engineering", 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), 2023 Publication	<1 %

18	Submitted to The Hong Kong Polytechnic University Student Paper	<1 %
19	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1 %
20	www.arxiv-vanity.com Internet Source	<1 %
21	Submitted to Laureate Education Inc. Student Paper	<1 %
22	www.thefreemanonline.org Internet Source	<1 %
23	Thomas J. Huggins, Feiyu E, Kangming Chen, Wenwu Gong, Lili Yang. "Infrastructural Aspects of Rain-Related Cascading Disasters: A Systematic Literature Review", International Journal of Environmental Research and Public Health, 2020 Publication	<1 %
24	docplayer.net Internet Source	<1 %
25	icymi.in Internet Source	<1 %
26	unsworks.unsw.edu.au Internet Source	<1 %
27	www.haag-streit.com Internet Source	<1 %

		<1 %
28	www.metacalculus.com Internet Source	<1 %
29	Submitted to Fiji National University Student Paper	<1 %
30	www.coursehero.com Internet Source	<1 %
31	www.mdpi.com Internet Source	<1 %
32	www.scribd.com Internet Source	<1 %
33	Ali Mohamud Abdulle, Adam Saed Ali, Abdirahman Salad Ali, Mohamed Abdullahi Ali, Sharamke Ali Kahie, Nasra Abdulkadir Mohamed. "IoT-based river monitoring and alerting system to mitigate flood damage", Elsevier BV, 2024 Publication	<1 %
34	arxiv.org Internet Source	<1 %
35	www.ijritcc.org Internet Source	<1 %

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1 of 1



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Kok How Meng
ID Number(s)	2002193
Programme / Course	CS
Title of Final Year Project	Flood Monitoring and Alert System

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u> 4 </u> % Similarity by source Internet Sources: <u> 3 </u> % Publications: <u> 1 </u> % Student Papers: <u> 1 </u> %	
Number of individual sources listed of more than 3% similarity: <u> 0 </u>	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Signature of Co-Supervisor

Name: Tan Teik Boon

Name: _____

Date: 25/4/2024

Date: _____



UNIVERSITI TUNKU ABDUL RAHMAN

**FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)**

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	2002193
Student Name	Kok How Meng
Supervisor Name	Ts Tan Teik Boon

TICK (√)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

Handwritten signature

(Signature of Student)

Date: 25/4/2024