International Journal of Advanced Research in Computational Thinking and Data Science

Journal homepage:
https://karyailham.com.my/journals/index.php/ctds/index
ISSN: 3030-5225

Integrating IoT Sensors and Machine Learning Algorithms for Early Flash Flood Detection System

Khaled Mohamed Abdelmagid[1], Mohd Fitri Mohd Yakub[1,*], Masitah Ghazali[1]

[1] Dept. Malaysia-Japan International Institute of Technology Universiti Teknologi Malaysia Kuala Lumpur, Malaysia

ARTICLE INFO

Article history:
Received in 28 December 2024
Received in revised form 27 January 2025
Accepted 1 March 2025
Available online 5 March 2025

Keywords:
Flash flood detection; machine learning algorithms; IoT sensors; early warning systems; real-time data analysis; environmental monitoring; disaster management; agile development; climate change impact; urbanization effects

ABSTRACT

The rising frequency of flash floods due to climate change demands efficient detection systems to reduce their impact. This study presents the "Early Flash Flood Detection System Using Machine Learning Algorithms," which integrates IoT sensors and machine learning for accurate, real-time flood prediction. Developed with Agile methodology, the project utilized key technologies like Flutter and TensorFlow to enhance functionality and user engagement. Testing showed 72% prediction accuracy, demonstrating the system's potential as a scalable solution for disaster management, advancing public safety, and fostering resilient communities.

1. Introduction

In the days where flash floods are becoming more common, particularly in locations prone to such catastrophes marked by quick onset and strong rainfall, sophisticated early warning systems are critical. This requirement prompted the creation of the "Integrating IoT Sensors and Machine Learning Algorithms for Early Flash Flood Detection System" project. The project's primary goal is to shift the paradigm of flash flood detection and response. It strives to improve early warning capabilities by implementing advanced, software-centric technologies. The project strategically integrates the real-time data collection capabilities of Internet of Things (IoT) sensors with the analytical complexity of machine learning algorithms. The goal is to create a more proactive and efficient framework for reducing the effects of flash floods. This effort aims to provide communities with timely and precise notifications, allowing for proactive steps in the face of impending flood hazards.

* *Corresponding author.*
E-mail address: mfitri.kl@utm.my

Second It combines cutting-edge technology innovation with comprehensive data analysis, establishing a new standard in catastrophe planning and response. The importance of early flash flood detection cannot be emphasized in a world where climate change is increasing the frequency and intensity of extreme weather occurrences [17]. In the face of this unrelenting natural force, the "Integrating IoT Sensors and Machine Learning Algorithms for Early Flash Flood Detection System" project stands as a beacon of hope, poised to improve our preparation, protect sensitive regions, and save lives.

1.1 Importance of the Project

The significance of this effort cannot be emphasized in today's world, when the effects of climate change are becoming increasingly apparent. Flash floods, with their unpredictable and destructive character, pose a serious threat to populations and infrastructure. This project's revolutionary approach, which combines technology and data science, has the possibility of producing a game-changing solution that goes beyond the constraints of traditional flood detection systems.

This project aims to create a cost-effective and highly efficient technique of early flash flood detection by using sophisticated technology and embracing cutting-edge approaches. The potential consequences of its success are far-reaching, since it can revolutionize how vulnerable areas prepare for and respond to flash flood catastrophes. Because of the innovative efforts of this endeavor, lives can be saved, property damage may be avoided, and resources can be deployed more effectively.

Furthermore, the initiative has substantial instructional and research implications at UTM. Using this system, students and researchers may obtain hands-on experience with modern IoT and machine learning technologies, leading to a better understanding of their applicability in real-world circumstances. This will not only improve the academic curriculum, but also establish UTM as a leader in environmental technology and disaster management research.

1.2 Analysis on Current System and Comparison

While digital marketing has its advantages, it is important to emphasize specific issues in the field of flash flood detection systems. The problems and complications of flood detection are dissimilar from those of marketing strategies. Nonetheless, there are some commonalities. These technologies necessitate a greater emphasis on accuracy, extensive interaction options, and a simple way of measuring their efficacy, similar to the insights acquired from ad campaign analytics. There is no space for complacency in the realm of flood detection, since the repercussions of mistakes may be catastrophic. The constant vigilance required in flood detection systems contrasts sharply with the 24-hour availability of digital marketing. In summary, the flood detection arena is unforgiving, and sustaining data monitoring and publishing processes is critical, even in the face of unforgiving competition.

Many present flood detection systems rely on significant physical infrastructure, such as water level monitors and rain gauges, which are frequently costly to construct and maintain. These systems often require extensive personal intervention to assure data quality and system performance. This reliance on physical infrastructure and manual processes can cause delays in data collecting and processing, reducing the timeliness and efficacy of flood warnings. Furthermore, the high expenses of implementing and maintaining these systems might be prohibitive for many communities, particularly those in underdeveloped countries.

Technological advancements have resulted in increasingly automated and advanced techniques to flood detection, which include Internet of Things (IoT) devices and machine learning algorithms. However, these contemporary systems confront several obstacles, such as the necessity for constant data communication and the difficulty of integrating numerous data sources. Furthermore, there are worries regarding the accuracy and dependability of data acquired by IoT devices, especially in severe environments. These concerns underscore the need for continued research and development in improving the robustness and reliability of automated flood detection systems.

To provide insight on existing practices, a localized study of flood detection apps is conducted, like the method used in navigation apps such as Waze. The WeatherNow app and Waze are two of the apps that have been scrutinized [12. While these apps are like travel apps in that they give listings of active flood detection programs, none of them provide detailed restrictions and amounts related with these campaigns. In the context of public safety, it is critical to resolve these disparities and ensure that the greatest number of participants are well-informed.

Aside from the lack of standardized data gathering criteria, the advantages of digital marketing echo inside flood detection systems. These advantages include not just the transmission of essential information, but also the collecting of data from concerned individuals. The simplicity with which data may be collected from consumers has the potential to improve flood prediction systems. User demographics such as gender, age, hobbies, geography, and preferences can be extremely useful in developing flood detection algorithms.

1.3 Comparison between existing system

The process of evaluating and contrasting existing systems plays a pivotal role in enhancing the efficacy of our flash flood detection system. System analysis, in this context, involves meticulous data collection, problem identification, and the dissection of system components to elucidate their objectives.

This comparative research shows commonalities between two popular apps: WeatherNow and Waze. Despite their diverse primary tasks, both programs exhibit basic ideas and characteristics that are relevant to the flash flood detection system now under development. These shared characteristics include campaign listings, incentive structures, and user profiles, all of which are critical components of the proposed flash flood detection architecture. Table I compares the systems, including descriptions and limits, to help readers understand their functionality and potential areas for development.

Table 1
Comparison of the features between 2 existing system

Application	Description	Limitations
WeatherNow	• WeatherNow offers an easy-to-use platform for obtaining flood detection features. • User registration has been simplified, allowing for simple sign-up without the need for vehicle details. • The app's vast coverage across many locations increases brand exposure and familiarity.	• Limited sign-in choices (such as Facebook or Apple ID) may limit user access. • There is no guarantee that regional users will be interested in promoted brands.
Waze	• Waze takes an all-encompassing approach, allowing users to report and access specific flood-related information. • Users can report flood-related occurrences independently, going beyond typical cars to encompass multiple means of transportation.	• Some flood-related situations may not be widely mentioned, necessitating previous awareness on the part of users. • Inadequate documentation for calculating distances in km.

2. Methodology

The Agile Software Development Methodology was carefully chosen as the foundation for the "Integrating IoT Sensors and Machine Learning Algorithms for Early Flash Flood Detection System" project. This purposeful choice is more than just a personal preference; it is a strategic alignment with the project's inherent demands and the dynamic character of its aims. Agile, with its basic concepts of flexibility, stakeholder engagement, and value incrementation, is particularly competent in dealing with the unexpected patterns and pressing needs that flash flood detection requires [1,17].

Unlike the classical waterfall model's linear evolution, Agile's flexible framework accommodates the fluidity of project requirements, making it well-suited for a system that requires real-time flexibility to natural occurrences.

The project's ethos is based on iterative improvement, which ensures that each development cycle, or sprint, produces a more enhanced version of the system. This implies that stakeholders aren't just passive recipients waiting for the finished result; rather, they participate in functioning versions of the system from the beginning. This regular connection allows for real-time input, promoting a responsive and evolutionary growth process. It enables the project to pivot and adapt, ensuring that the product is not only a technical success but also a practical solution precisely tailored to the demands of the community it is designed to serve.

2.1 Requirement Analysis and Elicitation

This section goes into the flash flood detection system's functional needs. The services that the system must provide to its users are referred to as functional requirements. These requirements are critical in defining the system's functions and are generated using the Agile approach through a collaborative process including stakeholders. Figure 1 depicts the system's functions, while Table II has a thorough explanation of the functional requirements.

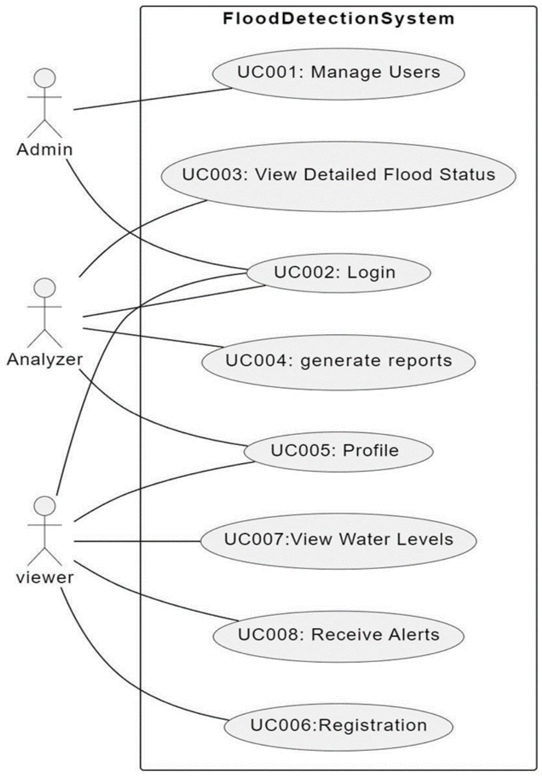


Fig. 1. Use Case Diagram of Early Flash Flood Detection System

Table 2
List of use cases in early flash flood detection system

No	Use Case	Description
1	UC001: Manage Users	Admin can manage user accounts, including creation, modification, and deletion.
2	UC002: Login	Users (Viewer and Admin) can log into their registered accounts in the web version of the application. Application will check the validity of email address and password. Analyzer and Admin can log in the Android version.
3	UC003: View Detailed Flood Status	Analyzer can access detailed information about flood status, including real-time data and historical records.
4	UC004: Generate Reports	Analyzer can generate PDF reports with the current flood information using a dedicated button.
5	UC005: Profile	Viewers and Analyzer can view and manage their profiles, including personal information and settings.
6	UC006: Registration	Viewers can register for an account, providing necessary information for the application.
7	UC007: View Water Levels	Viewers can view real-time water levels as part of the flood detection system.
8	UC008: Receive Alerts	Viewers receive alerts about flood situations based on system detection.

2.1 System Design

The Model-View-Controller (MVC) architecture in Figure 2 was chosen for its established concepts of modularity and separation of responsibilities for designing the Early Flash Flood Detection System (EFFDS). The Model performs data-related functions such as simulation, preprocessing, and user management in this method, establishing a clear separation from the user interface. The View contains pages for user interaction, whereas the Controller serves as a go-between for the Model and the View, handling user input and data flow. The modular nature of MVC improves system upkeep, upgrades, scalability, and extensibility.

This structure is especially beneficial for EFFDS, allowing for easy deployment, reducing dependencies, and enhancing real-time data processing speed. In summary, MVC assures that EFFDS is a well-organized, maintainable, and scalable system that meets the needs of effective flood detection and response.

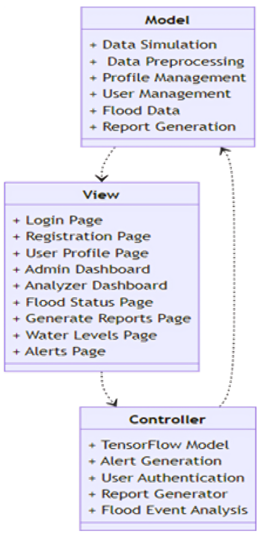


Fig. 2. Architectural model of early flash flood detection system

2.3 System Development

The technology stack utilized to create the Early Flash Flood Detection System (EFFDS) includes the Flutter framework and Firebase backend services. The Flutter framework was chosen for its ease of development and flexibility in designing user interfaces (UI). It allows the use of a single codebase to build natively compiled, multi-platform applications, which can run efficiently on both Android and iOS devices. Flutter's core language is Dart, which provides robust tools for formatting, analyzing, and testing code. The framework's fast performance ensures a smooth user experience across various devices.

Firebase serves as the backend service for EFFDS. Firebase is known for its straightforward setup and powerful capabilities, making it an ideal choice for real-time applications. The primary components of Firebase used in EFFDS are Cloud Firestore, Firebase Authentication, and Firebase Storage. Cloud Firestore is a flexible, scalable database for mobile, web, and server development. It provides real-time updates, powerful queries, and automatic scaling, making it a highly supportive client-accessible database solution. Firebase Authentication offers several sign-in methods, including email and password authentication, ensuring secure access to the system. Firebase Storage is used for storing user-uploaded files, such as images and documents, in a secure and scalable manner.

The architecture design pattern for EFFDS is the Model-View-Controller (MVC) pattern, facilitated with additional service components. This design ensures that the system is organized, maintainable, and scalable. The key components are the Model, which manages the data and business logic of the application; the View, which handles the presentation layer and displays the data provided by the Model in a user-friendly manner; and the Controller, which acts as an intermediary between the Model and the View, handling user inputs and updating the View accordingly. Service components include functions from additional services and packages imported, such as Firebase for database operations and other services. Shared components encompass widgets, dialogs, and constants used throughout the system to ensure consistency, maintainability, and reusability.

To facilitate the implementation of the MVC architecture in the Flutter framework, the GetX package is used. GetX provides state management, route management, and dependency injection, enabling efficient interaction between the Controller and the View layers. Firebase integration involves setting up the service configuration and adding the Firebase Software Development Kit (SDK) to the project. This setup allows the application to communicate seamlessly with Firebase services. The Firebase Authentication service is used to manage user accounts securely, storing user credentials (email and password) for login purposes. Cloud Firestore is utilized as the main database, providing real-time data synchronization across connected devices. Firebase Storage is employed to store files uploaded by users, such as profile pictures and other documents.

3. Early Flash Flood Detection System

The Early Flash Flood Detection System (EFFDS) includes main functions such as User Authentication Module, Flood Status Module, Alerts Module, and Report Generation Module. For the User Authentication Module interface, users can log in, register, and edit their profile. The login page is shown in Figure 3. Users can enter their email and password, and then press 'Login' to access the system. Users can also be redirected to the register page by tapping the 'Register now' button and be redirected to the forgot password page by tapping the 'Forgot Password' button. depicts the Alerts interface.

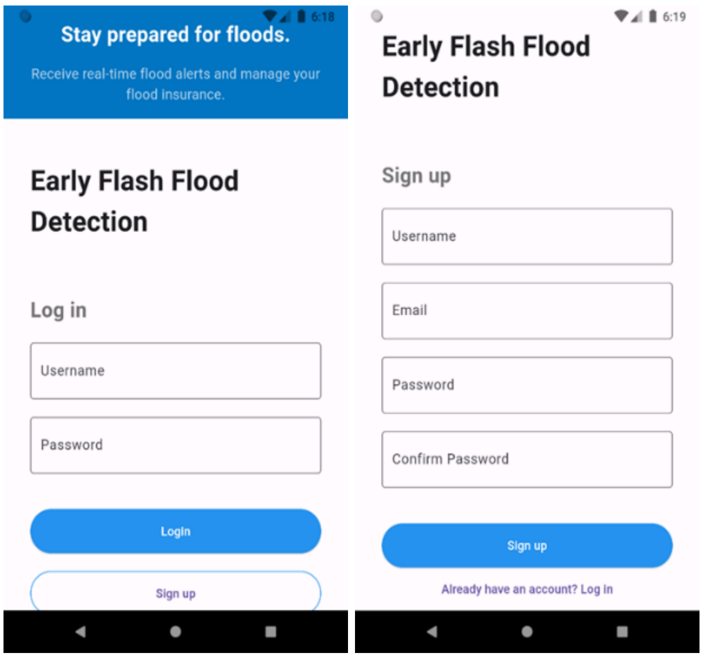
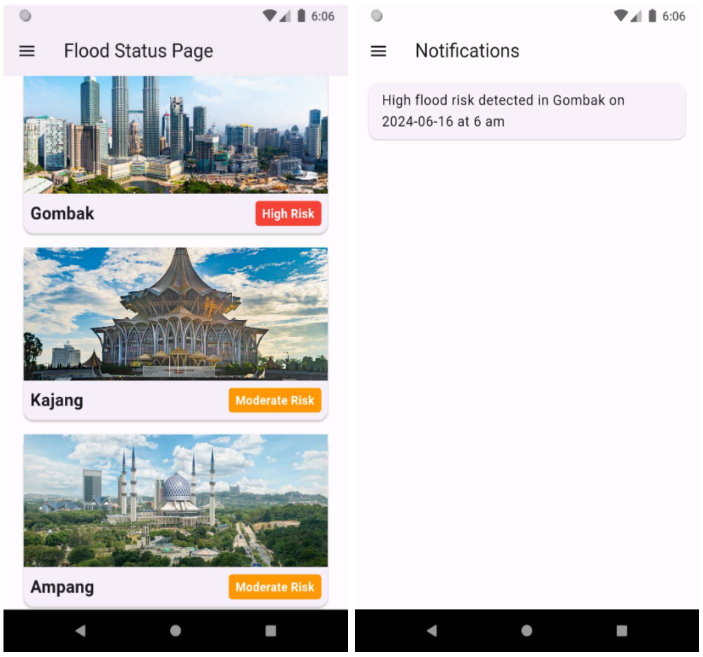

Stay prepared for floods.
Receive real-time flood alerts and manage your flood insurance.
Early Flash Flood Detection
Log in
Username
Password
Login
Sign up
Early Flash Flood Detection
Sign up
Username
Email
Password
Confirm Password
Sign up
Already have an account? Log in

Fig. 3. Login and registration interfaces

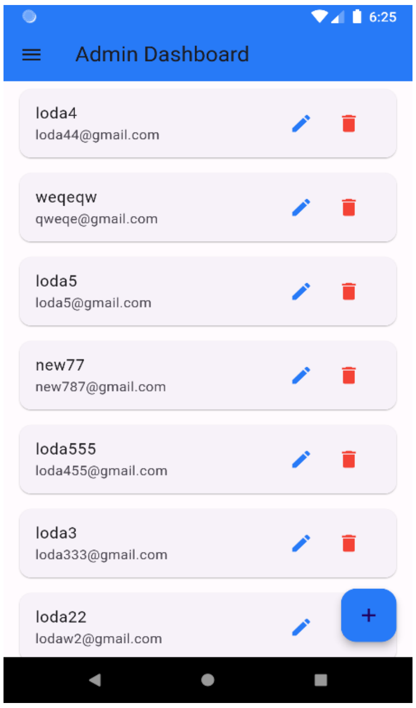
For the Flood Status Module, users can view current flood conditions, including real-time data and historical records. This interface is crucial for monitoring ongoing flood situations and making necessary preparations. Figure 4 shows the Flood Status interface, providing users with clear information about flood conditions. For the Alerts Module, users can receive notifications about potential flood risks.

This interface ensures that users are promptly informed about flood situations, enabling them to take timely actions to protect themselves and their properties. For the Admin Dashboard Interface, administrators are provided with a comprehensive view of user management and system analytics. This interface allows admins to perform essential tasks such as adding, editing, and deleting user accounts, as well as monitoring system performance and user activities. Figure 5 shows the Admin Dashboard interface



High flood risk detected in Gombak on 2024-06-16 at 6 am

Fig. 4. Flood status and alerts interfaces



loda4
loda44@gmail.com

weqeqw
qweqe@gmail.com

loda5
loda5@gmail.com

new77
new787@gmail.com

loda555
loda455@gmail.com

loda3
loda333@gmail.com

loda22
lodaw2@gmail.com

Fig. 5. Admin dashboard interface

For the Report Generation Module, analysts can create PDF reports based on current flood data. This function is essential for documenting and sharing flood analysis, helping stakeholders understand the flood situation and response actions. Figure 6 highlights the Generate Report interface. For the Analyzer Dashboard Interface, it offers tools for analyzing flood data and generating reports. This interface provides a detailed view of flood-related data and analytics, enabling analysts to make informed decisions based on real-time and historical data. Figure 8 shows the Analyzer Dashboard interface.

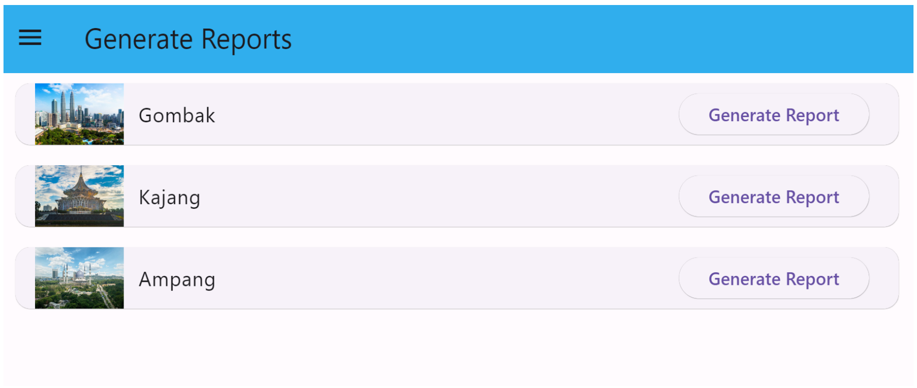
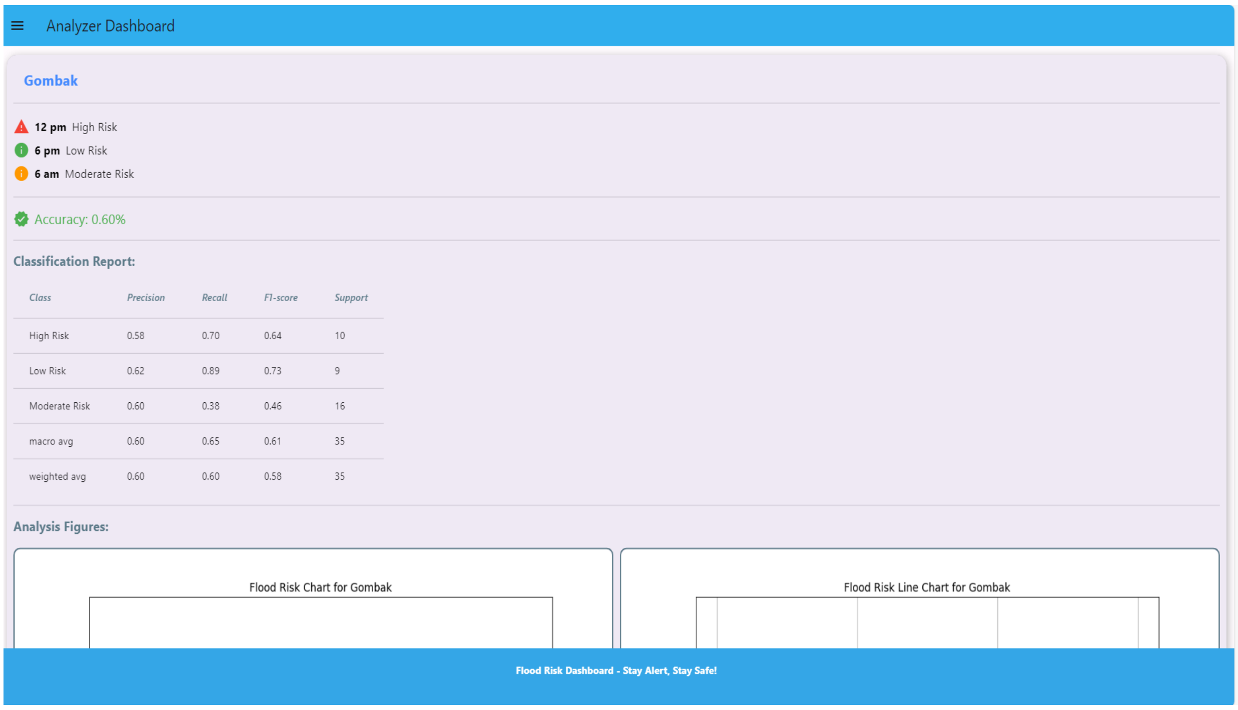


Fig. 6. Generate Report Interface

For the Analyzer Dashboard Interface, it offers tools for analyzing flood data and generating reports. This interface provides a detailed view of flood-related data and analytics, enabling analysts to make informed decisions based on real-time and historical data. Figure 8 shows the Analyzer Dashboard interface.


Analyzer Dashboard
Analysis Figures:
Flood Risk Chart for Gombak
Flood Risk Line Chart for Gombak
Flood Risk Dashboard - Stay Alert, Stay Safe!

Gombak

12 pm High Risk
6 pm Low Risk
6 am Moderate Risk

Accuracy: 0.60%

Classification Report:

Class	Precision	Recall	F1-score	Support
High Risk	0.58	0.70	0.64	10
Low Risk	0.62	0.89	0.73	9
Moderate Risk	0.60	0.38	0.46	16
macro avg	0.60	0.65	0.61	35
weighted avg	0.60	0.60	0.58	35

Fig. 7. Analyzer dashboard interface

4. System Testing and Results

The testing phase ensures the developed system fulfills user requirements and maintains high-quality standards. The primary focus is on identifying and resolving errors while also ensuring user satisfaction. Two types of testing were implemented: Black Box Testing and Usability Testing.

Black Box Testing aims to verify that the system behaves correctly based on specified requirements without considering the internal code, algorithms, or system architecture. Test cases were designed with valid, invalid, and empty inputs to verify if the results met expectations derived from user requirements. Prerequisites and test data were used to ensure testers performed the correct operations. All test results for the 16 test cases were 100% passed. Below are examples of Black Box Testing for the login process.

Table 3
UC001_01: User creation

Test Case ID	Input Data	Expected Result	Actual Result	Pass/Fail
TC001_01_01	User: John Doe, Email:john@example.com, Role: Viewer	Display message "User successfully created"	Display message "User successfully created"	pass
TC001_01_02	User: Jane Smith, Email:jane@example.com, Role: Analyzer	Display message "User successfully created"	Display message "User successfully created"	pass
TC001_01_03	User: Existing User, Email:existing@example.com, Role: Viewer	Display message "Invalid email or password"	Display message "Invalid email or password"	pass
TC001_01_04	User: Invalid Email, Email: invalid email, Role: Analyzer	Display message "Error: Invalid email format"	Display message "Error: Invalid email format"	pass

Table 4
UC001_02: User deletion

Test Case ID	Input Data	Expected Result	Actual Result	Pass/Fail
TC001_02_01	Admin clicks delete button for User ID: 1	Display message "User successfully deleted"	Display message "User successfully deleted"	pass
TC001_02_02	Admin clicks delete button for User ID: 999	Display message "Error: User not found"	Display message "Error: User not found"	pass
TC001_02_02	Admin clicks delete button for User ID: 999	Display message "Error: User not found"	Display message "Error: User not found"	pass

Table 5
Uc002_01: User login (Username)

Test Case ID	Input Data	Expected Result	Actual Result	Pass/Fail
TC001_01_01	Username: john	Prompt for password	Display message "User successfully created	pass
TC001_01_02	Username: jane	Display message "Error: Username not found"	Display message "Error: Username not found"	pass
TC001_01_03	Username: non_existing	Display message "Error: Invalid username format"	Display message "Error: Invalid username format"	pass
TC001_01_04	Username: inv@lid	Display message "Error: Invalid username format"	Display message "Error: Invalid username format"	pass

Out of a total of 34 test cases designed for the Early Flash Flood Detection System, all 34 test cases were executed, with every single one passing successfully, indicating a 100% test coverage and pass rate. This comprehensive testing ensured that all critical functionalities, including user authentication, flood status viewing, alerts, and report generation, performed as expected under various input conditions. The detailed testing tables for login functionalities (email, password, and login button) showed that all expected results matched the actual results, confirming the system's robustness and reliability.

The machine learning algorithm implemented in the system, specifically the Random Forest Classifier, was subjected to rigorous training and testing processes. The system utilized Stratified K-Fold cross-validation to ensure robust model training and prevent overfitting. The selected model demonstrated strong performance metrics with an accuracy of 72%, a weighted F1 score of 0.91, a precision of 0.92, and a recall of 0.91. Additionally, the ROC AUC score was 0.93, indicating a high level of discrimination capability in identifying flood risk levels.

The classification report provided further insights into the model's performance, showing that it effectively managed the different flood risk levels (low, moderate, high) with balanced precision and recall values. The successful implementation of these machine learning algorithms and the extensive testing carried out confirm the system's capability to provide early and accurate flood warnings. This ensures not only the fulfillment of project requirements but also contributes significantly to enhancing public safety and disaster preparedness. The testing procedures and results, along with the detailed analysis and performance metrics of the machine learning model, are documented in the appendices of this report for further reference.

5. Conclusions

Early Flash Flood Detection System (EFFDS) has been developed with a primary focus on integrating IoT sensors and machine learning algorithms to predict flash floods early. This project involved extensive planning, design, implementation, and testing phases to achieve its objectives. The major goal of the EFFDS project was to accurately predict flash floods using machine learning algorithms applied to generated data. This goal was realized through a thorough development process involving significant stakeholder engagement. Two key objectives were achieved in the first phase of the project: gathering and analysing stakeholder requirements and creating a system based on those requirements. The formation of the Software Requirement Specification (SRS) and Software Design Document (SDD) during the planning phase set a solid foundation for the project. The design process, utilizing UML, enabled a flexible and evolving system architecture. These designs were implemented with a focus on collaborative development and continuous integration.

The system's functionality and integrity were rigorously tested to meet high-quality standards necessary for reliable flood detection. The technology stack, including Visual Studio Code, Flutter, Dart, Firebase, and TensorFlow, played a crucial role in the project's success. These tools and frameworks facilitated rapid development, seamless integration, and robust machine learning capabilities essential for processing and analysing data effectively. As the project moves into its second phase, the focus will shift to enhancing system efficiency and expanding data generation capabilities. The Agile methodology will continue to guide the project, ensuring responsiveness to new requirements and maintaining system adaptability in the dynamic field of disaster management. More intensive testing strategies, including automated testing, will be employed to ensure ongoing quality and system integrity. User feedback will be crucial in guiding future development cycles.

The EFFDS project has successfully met its objectives and requirements, providing a reliable and efficient system for early flash flood detection. The system is expected to play a significant role in improving disaster preparedness and response, contributing to safer and more resilient communities. While the current system is well received by users, there is room for further enhancement. Future improvements could include additional features such as direct communication tools, more comprehensive navigation aids for users, and better data management capabilities for collaborating companies. These enhancements will help ensure the system remains effective and user-friendly, further aiding in disaster management efforts.

Acknowledgement
In preparing this thesis, I have had the privilege of interacting with numerous individuals whose contributions have been invaluable to my understanding and progress. I would like to express my deepest gratitude to my principal thesis supervisor, Professor Dr. Fitri Yakub, for his unwavering support, meticulous guidance, constructive criticism, and enduring friendship. His insightful direction has been instrumental in the completion of this work. I am equally indebted to my co-supervisor, Professor Dr. Masitah Ghazali, for her exceptional guidance, profound counsel, and steadfast encouragement. Dr. Masitah's dedication to my academic journey and her thoughtful feedback have significantly shaped this thesis. Her expertise and unwavering support have been a constant source of inspiration and motivation. Without her continuous attention and invaluable insights, this thesis would not have achieved its present form. My fellow students should also be commended for their assistance. My heartfelt gratitude goes to all my colleagues and those who have helped me on several occasions. Their opinions and suggestions are quite beneficial. Unfortunately, there isn't enough room to mention them all. I am grateful to all my family members. I would like to mention that this research was not funded by any grant.'

References

[1] Dubakov, M. 7 Steps of Agile System Analysis Process. (2008).

[2] Wong, Euphemia. "User interface design guidelines: 10 rules of thumb." *User Interface Design, September* (2020).

[3] CareerFoundry. "Explore UI Design | the CareerFoundry Blog," (2023).

[4] GitHub. "Flutter/flutter: Flutter makes it easy and fast to build beautiful apps for mobile and beyond," (2020).

[5] Flutter definition & meaning. (n.d.)

[6] Ionicframework. "Android Development: Ionic Documentation." (2023).

[7] Khan, Rijwan, Mohammad Shabaz, Sarfaraj Hussain, Faraz Ahmad, and Pranav Mishra. "Early flood detection and rescue using bioinformatic devices, internet of things (IOT) and Android application." *World Journal of Engineering* 19, no. 2 (2022): 204-215. https://doi.org/10.1108/WJE-05-2021-0269

[8] Subeesh, A., Prashant Kumar, and Naveen Chauhan. "Flood early detection system using internet of things and artificial neural networks." In *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2018, Volume 1*, pp. 297-305. Springer Singapore, 2019. https://doi.org/10.1007/978-981-13-2324-9_30

[9] Anbarasan, M., BalaAnand Muthu, C. B. Sivaparthipan, Revathi Sundarasekar, Seifedine Kadry, Sujatha Krishnamoorthy, and A. Antony Dasel. "Detection of flood disaster system based on IoT, big data and convolutional deep neural network." *Computer Communications* 150 (2020): 150-157. https://doi.org/10.1016/j.comcom.2019.11.022

[10] Mosavi, Amir, Pinar Ozturk, and Kwok-wing Chau. "Flood prediction using machine learning models: Literature review." *Water* 10, no. 11 (2018): 1536. https://doi.org/10.3390/w10111536

[11] Costache, Romulus, Alireza Arabameri, Ismail Elkhrachy, Omid Ghorbanzadeh, and Quoc Bao Pham. "Detection of areas prone to flood risk using state-of-the-art machine learning models." *Geomatics, Natural Hazards and Risk* 12, no. 1 (2021): 1488-1507. https://doi.org/10.1080/19475705.2021.1920480

[12] Roy, Minakshi, Prakar Pradhan, Jesson George, and Nikhil Pradhan. "Flood detection and water monitoring system using IOT." *International Journal Of Engineering And Computer Science* 9, no. 07 (2020): 25113-25115. https://doi.org/10.18535/ijecs/v9i07.4499

[13] Zhang, Tianhong, Sheng Liu, Weidong Xiang, Limei Xu, Kaiyu Qin, and Xiao Yan. "A real-time channel prediction model based on neural networks for dedicated short-range communications." *Sensors* 19, no. 16 (2019): 3541. https://doi.org/10.3390/s19163541

[14] Lim, I. "Flash floods hit Kuala Lumpur again: What you need to know." (2022).

[15] Tan, C. "Malaysia floods: Flooding in Malaysia leaves 30,000 people displaced." (2021).

[16] Sani, R. "Floods disrupt life in Malaysian cities." (2021).Kadir, Muhammad, Radzi Ambar, Mohd Helmy, Abd Wahab, Isa Khalid, Herdawatie Kadir, Dedi Suryadi, et al. "FloWMS: Flood Rapid Warning and Monitoring System." *Journal of Advanced Research in Applied Sciences and Engineering Technology Journal Homepage* 56, no. 2 (2026): 108–26.

[17] Shipun Anuar Hamzah, Mohd Noh Dalimin, Mohamad Md Som, Mohd Shamian Zainal, Khairun Nidzam Ramli, Mohd Hamim Sanusi@Ikhsan, Azli Yusop, et al. "Flood Level Detection System Using Ultrasonic Sensor and ESP32 Camera: Preliminary Results." *Journal of Advanced Research in Applied Mechanics* 119, no. 1 (2024): 162–73. https://doi.org/10.37934/aram.119.1.162173

[18] Norah N. Alajlan, and Dina M. Ibrahim. "Deep Smart Cities: A Review of Smart Cities Applications-Based an Inference of Deep Learning upon IoT Devices." *Journal of Advanced Research in Applied Sciences and Engineering Technology* 47, no. 2 (2024): 94–120. https://doi.org/10.37934/araset.47.2.94120.

[19] Bakhit, A, None Nur, None Mohd, None Mohd, None N.H. Ramli, None Muhammad, A Alhaj, and None Ehtesham Ali. "IoT-Based Machine Learning Comparative Models of Stream Water Parameters Forecasting for Freshwater Lobster." *Journal of Advanced Research in Applied Mechanics* 117, no. 1 (June 2, 2024): 137–49. https://doi.org/10.37934/aram.117.1.137149.

[20] Norfatihah Bahari, Nor, Mohd Wafi Nasrudin, Rashidah Che Yob, Nur Hidayah Ramli, and Herwansyah Lago. “IoT Based Earthquake Detection System.” *Journal of Advanced Research in Applied Sciences and Engineering Technology* 51, no. 1 (September 4, 2024): 160–70. https://doi.org/10.37934/araset.51.1.160170.