CHAPTER 1

INTRODUCTION

INTRODUCTION

1.1 Introduction

Floods are among the most dangerous natural disasters, leading to loss of life, destruction of property, and significant disruptions in communities. With climate change resulting in more extreme weather patterns, the frequency and intensity of floods are on the rise, making it crucial for individuals and communities to be adequately prepared to respond swiftly and effectively when floods occur.

Traditional flood warning systems, such as television and radio alerts, often struggle to provide real-time information and may not reach everyone, particularly those in remote or isolated areas. Furthermore, during flooding events, power outages and communication failures can hinder the timely dissemination of vital updates.

Flood Guard is designed to address these challenges by leveraging modern technologyto deliver real-time flood alerts and updates via mobile applications, SMS notifications, and interactive maps. The platform also enables users to report local flood conditions, enhancing the accuracy of alerts and contributing to better flood mapping in the future.

By integrating automated alerts, community input, and effective management, Flood Guard aims to enhance disaster preparedness and mitigate the impacts of flooding on affected communities.

1.2 Motivation

The motivation for the Flood Guard project comes from the growing number of floodshappening around the world, which are becoming more frequent and severe due to climate change. Floods can cause significant damage to homes, businesses, and lives, especially in communities that are not well-prepared for such emergencies. The tragiceffects of recent floods have shown that there is a strong need for better tools to help people manage flood risks and respond quickly when disasters strike.

Flood Guard is designed to help individuals and communities by providing important information and alerts about potential floods. Our goal is to help people prepare in advance, so they can take action before a flood occurs, rather than just reacting afterward.

We also want to encourage teamwork within communities by creating a space where volunteers can coordinate their efforts to assist those in need during floods. The app includes feature interactive maps to help families affected by floods find the support they need.

Overall, the motivation behind FloodGuard is to make communities safer and more ready to handle floods. We hope to inspire a sense of preparedness and teamwork, changing how communities respond to these natural disasters.

1.3 Application

The Flood Guard project aims to provide a comprehensive solution for flood management by utilizing modern technology and community engagement. The applications of this project include:

Flood Alerts: The application delivers immediate alerts to users based on real-time data, ensuring that individuals receive timely warnings about potential flood risks in their areas.

Community Reporting: Users can report local flood conditions, which enhances the accuracy of the information disseminated and helps authorities make informed decisions.This feature fosters community involvement and situational awareness.

Multi-Channel Communication: Flood Guard leverages various communication channels, including mobile applications, SMS notifications, and interactive maps, to reach users effectively, regardless of their access to technology or internet connectivity.

Emergency Preparedness: By providing critical information and resources, the application helps communities prepare for floods more effectively. This includes educational materials on flood safety and emergency response protocols.

Data Analysis and Mapping: The collected data can be analyzed to improve flood forecasting models and create detailed flood risk maps, aiding in long-term planning andmitigation strategies.

1.4 Organization of report

This report presents the Flood Guard Communication Platform, starting with an introduction that outlines the project's background, motivation, and practical applications. It emphasises the importance of an efficient system for flood monitoring and emergency response, detailing the benefits it offers to users and authorities through real-time alerts and assistance.

Following this, the report reviews existing flood management technologies, identifying their strengths and limitations, which sets the stage for addressing the research gap. It defines the specific problem the project tackles and presents clear objectives focused on delivering timely alerts, managing SOS request. The proposed system is described, detailing its structure, design, and methodology for processing data and facilitating communication. Additionally, it covers the technologies used, including database requirements, performance metrics, and necessary software and hardware for operation. The report concludes with references to the sources that informed the project's development.

CHAPTER 2

LITERATURE SURVEY

LITERATURE SURVEY

2.1 Literature Survey

Flood management systems are essential for mitigating the impacts of flooding and protecting communities, particularly in areas prone to natural disasters. Recent advancements in technology, data-driven approaches, and community engagement strategies have significantlyenhanced the effectiveness of these systems.

Flood Early Warning Systems (FEWS) play a critical role in disaster management, as they significantly reduce property damage and enhance community preparedness. For instance, Vassar Labs has developed its aqua FLOOD platform, which employs advanced technologies to provide accurate flood forecasts by integrating real-time data and hydrological modeling. This platform exemplifies how technology can enhance public safety, minimize economic losses, and foster community engagement in flood monitoring and response activities [2]. By offering detailed forecasts, FEWS enable local authorities and residents to prepare adequately for potential flooding, thus improving overall disaster resilience.

In addition to FEWS, decision-support systems (DSS) based on spatially distributed hydrological models have been evaluated for their efficacy in managing flood risk and supporting agricultural planning. By integrating the SWAT hydrological model with decision- making tools, these systems analyze the impacts of land use, climate change, and water resource management on flood risks. A case study conducted in Italy demonstrated the utilityof DSS in providing simulations and forecasts that help stakeholders make informed decisions regarding flood management and agricultural strategies [1]. The ability to simulate various scenarios allows policymakers to understand potential future risks and implement proactive measures to safeguard communities and their livelihoods.

Moreover, innovative flood management systems that leverage cloud computing and Internet-of-Things (IoT) technologies address the limitations of traditional human-centric disaster management programs. These systems aim to minimize human intervention and automate theflood prevention and mitigation processes. For instance, research on the

Mumbai floods of 2005 illustrates how IoT-based approaches can create a self-organizing network capable of forecasting flood events and managing wastewater disposal systems [3]. By harnessing real- time data from a variety of sensors, these systems can continuously monitor environmental conditions and provide timely alerts to relevant stakeholders, enhancing responsiveness and resource allocation during flood events.

A particularly noteworthy implementation is the Flood Level Monitoring System (FLMS), designed specifically for Kurla, a suburban area of Mumbai. This system utilizes machine learning techniques to analyze historical data on rainfall, water levels, and other relevant factors to predict potential flood events with a high degree of accuracy. As a web-based application, FLMS enables users to access real-time flood alerts, which keeps them informed and allows them to take necessary precautions. Moreover, the system facilitates communication with relevant personnel, enabling effective coordination and response during flood emergencies [4]. The FLMS also emphasizes community engagement by raising awareness about the stages of a disaster and providing guidance on mitigating losses, thereby enhancing overall community resilience.

In recent years, the growing use of machine learning algorithms in flood forecasting has gained significant attention. These algorithms are trained on historical climatic data, enabling them to predict potential flood events effectively. A variety of models have been applied successfully to flood forecasting tasks, highlighting the potential for advanced analytics to improve predictive capabilities [5] . Researchers aim to provide a comprehensive overview of these algorithms, empowering other practitioners to implement similar approaches in their flood forecasting projects. This shift towards data-driven decision-making marks a significant advancement in the field, as it enables more precise and timely responses to flooding events.

Additionally, an innovative flood early warning system utilizing IoT technology has been proposed, which employs a network of sensors to collect real-time data on rainfall intensity, water levels, and weather conditions. This data is processed on a cloud platform, generating accurate flood forecasts and alerts. The IoT-based approach offers several advantages over traditional flood early warning systems, including comprehensive monitoring of flood-prone regions and multi-channel alert delivery through mobile applications and notifications. [6] .

Overall, the integration of advanced technologies into flood management systems represents a transformative shift in how communities prepare for and respond to flooding. These innovations not only enhance predictive capabilities but also improve community engagement and resilience, making them invaluable tools for mitigating the adverse effects of floods. As climate change continues to exacerbate the frequency and intensity of flooding events globally, investing in these advanced systems will be crucial for safeguarding lives, property, and livelihoods. By fostering collaboration among researchers, policymakers, and communities, we can develop more effective flood management strategies that prioritize sustainability and resilience in the face of increasing environmental challenges.

2.2 Existing Systems

1. National Flood Early Warning System (NFEWS)

The National Flood Early Warning System (NFEWS) is designed to predict and monitor floods across India by integrating hydrological models, meteorological data, and remote sensing technologies. This comprehensive system enables real-time assessment of rainfall, river conditions, and potential flood scenarios. By providing proactive alerts to stakeholders, including government agencies and local communities, NFEWS aims to enhance preparedness and minimize the impact of flooding events on lives and property.

2. Flood Forecasting System (FFS) in Kerala

Kerala's Flood Forecasting System (FFS) is a regional initiative focused on utilizing real-time data from hydrological sensors to issue localized flood alerts. Given the state's susceptibility to heavy monsoon rains, FFS plays a crucial role in helping communities prepare effectively forpotential flooding. By providing timely and specific warnings, the system enhances local preparedness and response capabilities, ultimately aiming to reduce the risks associated with flood events.

3. Flood Inundation Mapping System (FIMS)

The Flood Inundation Mapping System (FIMS) is a vital tool for generating detailed maps that identify flood-prone areas based on hydrological data and land use patterns. This system aidsin long-term urban planning and disaster management by providing essential information for policymakers and urban planners. By visualizing flood risks, FIMS supports strategic decision- making and helps communities better prepare for potential flood events.

4. Central Water Commission (CWC) Flood Forecasting

The Central Water Commission (CWC) operates a robust flood forecasting system that monitors river basins across India. By collecting continuous data on river levels and flow rates, the CWC provides timely flood forecasts to mitigate potential damage. This system plays a crucial role in alerting communities and local authorities in advance, allowing for effective preparedness and response to flood situations, ultimately reducing the impact on lives and infrastructure.

CHAPTER 3

LIMITATIONS OF EXISTING SYSTEMS OR RESEARCH GAP

LIMITATIONS OF EXISTING SYSTEMS
OR RESEARCH GAP

3.1 Limitation Of Existing Systems

1. National Flood Early Warning System (NFEWS)

- Data Integration Challenges: NFEWS relies on a wide variety of data sources (hydrological models, meteorological data, and remote sensing). Integrating these diverse datasets can be complex and may lead to inconsistencies or inaccuracies in flood predictions.
- Limited Localized Alerts: While it provides nationwide alerts, the granularity of alerts may not be sufficient for specific local areas, potentially missing critical localized flood events.

2. Flood Forecasting System (FFS) in Kerala

- Dependence on Sensor Infrastructure: The effectiveness of this system is heavily reliant on the presence and accuracy of hydrological sensors. Areas with inadequate sensor coverage may not receive timely alerts.
- Short Forecast Lead Time: The system may have limited lead time for warnings, making it challenging for communities to prepare adequately before a flood event occurs.

3. Flood Inundation Mapping System (FIMS)

- Static Mapping: FIMS relies on historical data and may not adequately account for changes in land use, climate patterns, or infrastructure developments, which can affectflood risk.
- Limited Real-Time Data Utilization: The system may not effectively integrate real-time data, potentially delaying responses to emerging flood threats.

4. Central Water Commission (CWC) Flood Forecasting

- Limitation: Broad Focus: While CWC monitors multiple river basins, its broad focus may overlook local nuances, leading to less effective localized flood management strategies.
- Limitation: Resource Constraints: Limited funding and resources can hinder the implementation of advanced technologies and infrastructure improvements necessaryfor more accurate forecasting.

3.2 RESEARCH GAP

Existing flood management systems provide crucial support during emergencies, but several limitations remain unaddressed. Below are key research gaps identified in these systems, which the Flood Guard platform aims to fill.

1. Fragmented Communication Channels:

Many existing platforms offer centralized alerts through government services, websites, or SMS. However, they lack two-way communication, preventing real-time interaction betweenusers and responders. Flood Guard bridges this gap by enabling real-time SOS alerts and two- way interaction, enhancing coordination during emergencies.

2. Lack of Personalized and Predictive Alerts:

Most systems send generalized alerts, often without accurate, location-specific data. Additionally, they primarily rely on forecasts and river sensors, limiting predictive capabilities. Flood Guard addresses this by using machine learning models to generate hyper-local alerts based on real-time weather data and user locations.

3. Delayed Emergency Response and Coordination:

Current systems often rely on manual coordination, causing delays in rescue operations. Many notify emergency services only after floods occur, missing the chance for timely evacuation. Flood Guard automates SOS handling and provides live location tracking of users, ensuring quick response and better coordination with rescue teams.

4. Insufficient Community Engagement:

Most systems are top-down, with minimal involvement from local communities or volunteers. They do not encourage citizens to report incidents dynamically or allow volunteers toparticipate in rescue efforts. Flood Guard fills this gap by enabling citizen reporting and volunteer coordination to strengthen local disaster management.

5. Limited Integration of Data Sources:

Many platforms rely on single data sources (e.g., weather forecasts or sensor data), which limits the accuracy of their predictions. Flood Guard integrates multiple data streams—such as rainfall data, user reports, and sensor inputs—to ensure, real-time predictions.

CHAPTER 4
PROBLEM STATEMENT AND OBJECTIVE

PROBLEM STATEMENT AND OBJECTIVE

4.1 Problem Statement

Floods are dangerous natural disasters that can cause severe damage, disrupt communities, and lead to loss of life. With climate change making extreme weather more common, floods are becoming harder to predict and control. Although existing systems like the National Flood Early Warning System (NFEWS), the Central Water Commission's (CWC) forecasting, and state-level solutions provide alerts, they often face problems such as delayed updates, lack of local information, and poor communication during emergencies, especially in remote areas.

These systems rely heavily on radio or TV, which may not be accessible during power outagesor network disruptions. Additionally, most existing solutions do not allow people to provide real-time information from affected areas, limiting the accuracy of alerts. They are also focused on specific regions, making them less effective in dealing with cross-regional floods.

Flood Guard aims to solve these issues by providing real-time alerts, community reporting features, and offline access through SMS and mobile apps. It empowers both authorities and citizens to report flooding conditions, improving forecasts and response efforts. With multi- channel communication and interactive maps, Flood Guard ensures people receive timely andaccurate updates to stay safe, reducing the impact of floods on lives and property.

4.2 Objectives

Early Flood Detection and Prediction:

- Implement advanced algorithms to analyze real-time and historical data for early detection and prediction of flood events.

Localized Alert System:

- Develop a system that provides timely and localized alerts to users based on their geographic location, ensuring relevant and actionable information during emergencies.

User-Friendly Interface:

- Create an intuitive and accessible mobile application that allows all community members, including vulnerable populations, to easily receive alerts and access flood- related resources.

Community Engagement and Education:

- Enhance community awareness of flood risks and safety measures through educational programs, outreach initiatives, and in-app resources, fostering a culture of preparedness.

Efficient Resource Allocation:

- Establish a framework for local authorities and volunteers to coordinate and allocateemergency resources effectively during flood events.

Collaboration with NGOs and Local Organizations:

- Partner with non-governmental organizations and local groups to provide additional support and resources for communities in high-risk areas, enhancing the overall floodmanagement approach.

CHAPTER 5
PROPOSED SYSTEM

5.1 Analysis / Algorithm

Algorithms for Flood Guard System

1. Random Forest Algorithm for Flood Prediction:

The Random Forest algorithm is an ensemble learning technique that creates multiple decision trees using different data samples. Each tree votes on the flood likelihood, and the majority vote becomes the prediction. This approach ensures robustness by reducing overfitting, handling missing data, and improving prediction accuracy.

This algorithm ensures that even if some data sources are noisy or incomplete, reliable predictions are still made, helping communities receive alerts on time.

2. Dijkstra's Algorithm for Evacuation Route Optimization:

This algorithm finds the shortest path between nodes in a weighted graph, making it ideal for real-time evacuation route planning. In flood situations, Dijkstra's algorithm dynamically adjusts paths based on updated inputs (like closed roads or waterlogged areas). It ensures thatusers are guided to the nearest safe zones efficiently.

3. Naive Bayes Algorithm for Early Alerts:

The Naive Bayes classifier is a probabilistic algorithm that could complement Random Forest by providing early alerts with limited or incomplete data. It can predict flood probabilities byanalyzing isolated features like sudden changes in rainfall or water levels.

4. K-Means Clustering for Risk Zone Classification:

K-Means can classify areas based on flood vulnerability, historical data, and environmental conditions. This clustering helps identify high-risk zones for proactive alerts and preventive actions.

5.Support Vector Machines (SVM):

Train an SVM model on historical flood data to classify whether certain conditions will lead to flooding. This can help improve the accuracy of flood predictions.

5.2 Design Details

1. System Design for Flood Guard Communication Platform

The Flood Guard Communication Platform ensures proactive disaster management by integrating real-time data collection, automated alerts, and rescue coordination. Below is a detailed prediction of how the system flow and design operate across different modules.

i. User Interaction Module

The system begins with user interaction via a mobile application, where individuals can report flood-related incidents or water level observations. Users also receive personalised alerts andemergency instructions directly on their app, ensuring timely access to critical information. The app offers an SOS feature that allows users to send distress signals along with their live location to the server for immediate help.

ii. Data Collection and API Gateway Module

When a user submits a report, the API Gateway securely transmits the data to the backend server. Additionally, the system integrates data from external sources such as weather forecasts, water sensors, and satellite imagery. This continuous inflow of information helps the platform maintain an up-to-date understanding of environmental conditions.

iii. Server and Database Module

The server processes incoming data and stores it in a centralised database. For flood predictions, the server queries machine learning models, trained with historical weather and flood data. These models provide real-time predictions to detect high-risk flood situations. The server also manages the state of SOS requests, keeping track of pending and resolved incidents to prevent response delays.

iv. Alert and Notification Module

When the ML model identifies a potential flood, the server sends personalised alerts to users in affected areas. If an SOS request is received, the system automatically notifies emergency services and volunteer responders. Two-way communication between the server and responders ensures smooth coordination throughout the rescue process.

v. Emergency Response and Rescue Coordination Module

For users in critical situations, such as being trapped in floodwaters, the system sends their live location to the rescue team. This ensures responders can locate victims efficiently and send appropriate help. Volunteers also receive notifications, allowing community-led rescue efforts to complement formal emergency services.

vi. Post-Emergency Feedback and Continuous Monitoring Module

After resolving an SOS, the system requests user feedback on the rescue experience. This feedback is stored and analysed to improve future disaster responses. Additionally, the ML model adapts to new data, enabling it to make better predictions over time. The platform also continues to monitor environmental conditions, ensuring it is prepared for future events.

2. System flow diagram

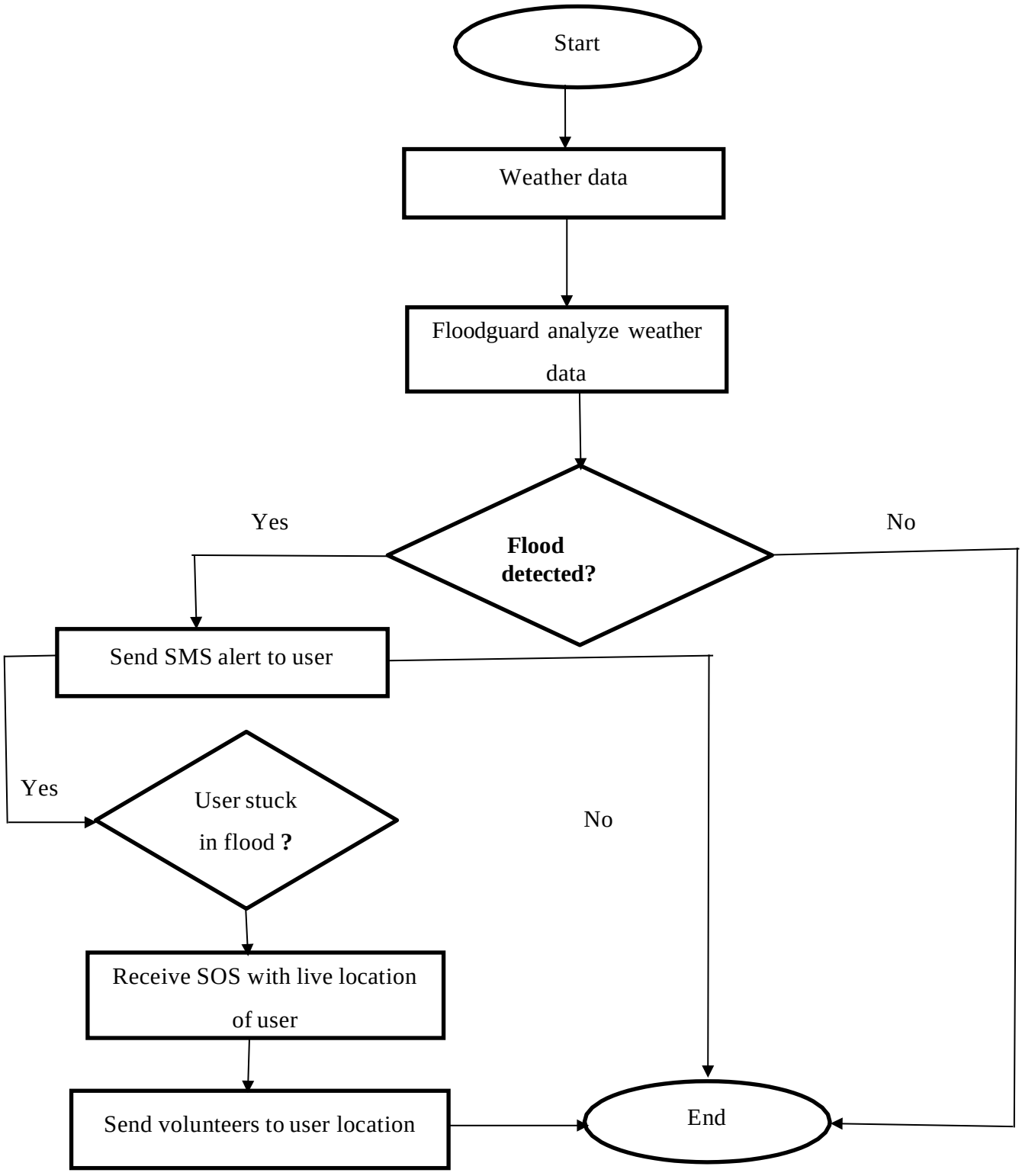

Fig 1. System flow diagaram

3. System Flow

Here is the explanation flowchart :

i. Start

- The system begins its operation from this point. It initiates by gathering weather-related information.

ii. Weather data

- The system collects weather data, which could be sourced from APIs or other weathermonitoring tools. This data is necessary for determining flood risks.

iii. FloodGuard analyzes weather data

- The collected weather data is analyzed by the FloodGuard system. The analysis likely checks factors such as rainfall intensity, water levels, or any other flood indicators.

iv. Flood condition detected?

- The system now evaluates whether a flood condition exists based on the analyzedweather data:
    - **Yes**: If the conditions indicate a potential flood, the system proceeds to alertusers.
    - **No**: If no flood risk is detected, the system concludes (leads to **ix. End**).

v. Send SMS alert to user

- If a flood condition is detected, the system sends SMS alerts to users in the affectedareas to notify them of the potential danger.

vi. User stuck in flood?

- After receiving the alert, users have the option to send an SOS if they are in trouble dueto the flood. The system checks if any user is stuck:
    - **Yes**: If the user is stuck and sends an SOS, the system moves to the next step tomanage the rescue operation.
    - **No**: If the user is not stuck or does not send an SOS, the system process ends

vii. Receive SOS with live location of user

- If a user is stuck in a flood and sends an SOS, the system receives the SOS message along with the user's live location. This information is crucial for organizing rescue .

viii. Send volunteers to user location

- Based on the SOS and live location data, the system dispatches volunteers to help theuser in need.

4. UML Sequence Diagram

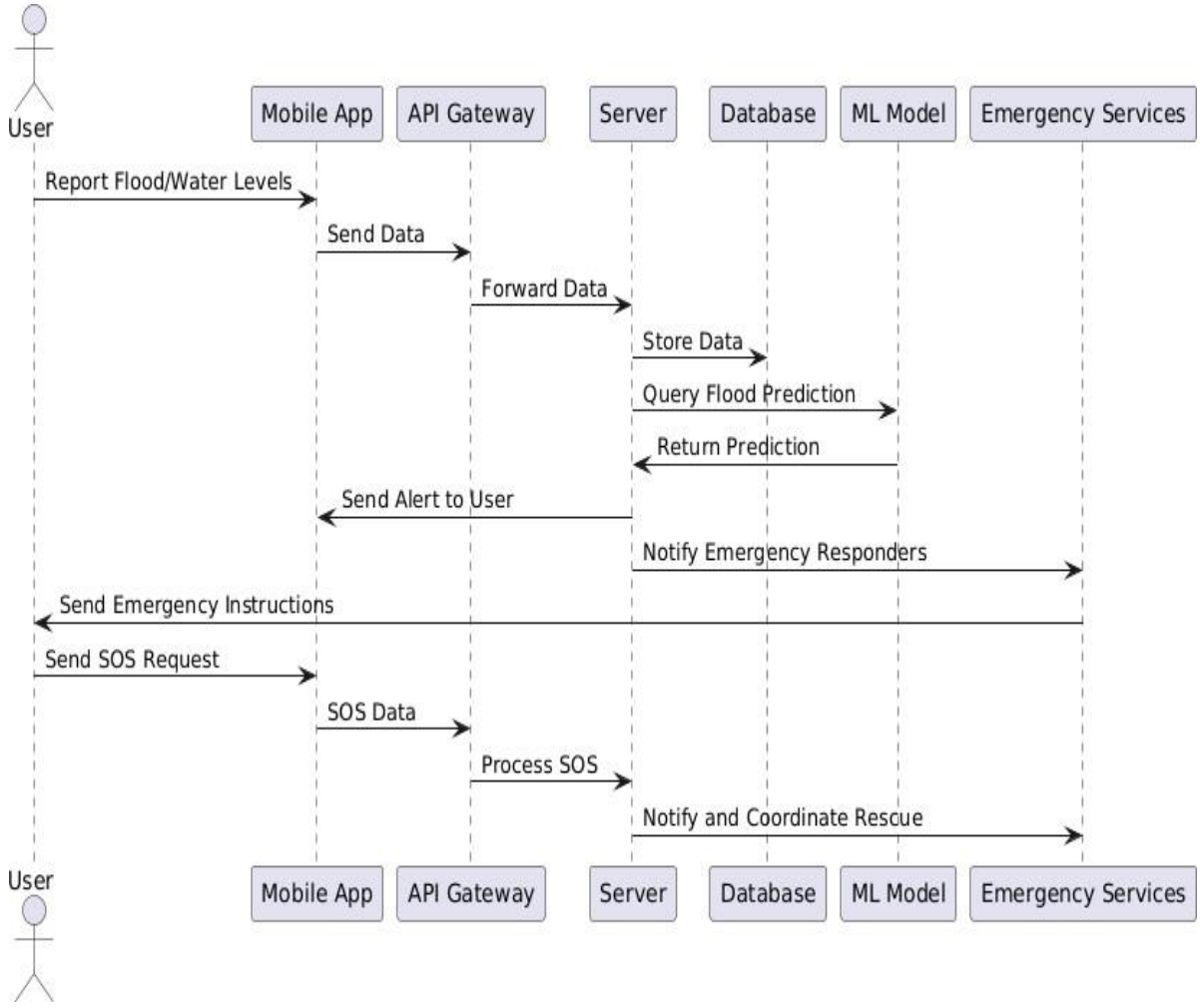

Fig 2. UML sequence diagram

This UML sequence diagram illustrates the flow of communication between different components of the Flood Guard system. It shows how users, the mobile app, API gateway, server, database, machine learning (ML) model, and emergency services interact in various scenarios like flood reporting, prediction, and SOS handling.

Actors and Components

1. **User**
    - The end-user of the mobile application who reports flood conditions or sendsan SOS.
    - Functions:
        - Reports flood levels and initiates SOS alerts if needed.
        - Receives alerts and emergency instructions.

2. **Mobile App**
    - The user interface to interact with the Flood Guard system.
    - Functions:
        - Collects user reports on flood conditions.
        - Sends SOS requests from users to the backend.
        - Receives alerts and instructions.
3. **API Gateway**
    - Acts as a middle layer to manage communication between the mobile app andbackend systems.
    - Functions:
        - Forwards data from the app to the server.
        - Ensures secure communication.
4. **Server**
    - The backend logic that coordinates data processing, predictions, and alerts.
    - Functions:
        - Stores data in the database.
        - Processes SOS requests and coordinates with emergency responders.
5. **Database**
    - Stores flood-related data and SOS request logs.
    - Functions:
        - Stores user-reported data.
        - Maintains records for predictions and SOS logs.
6. **ML Model**
    - A predictive model used to analyze flood-related data.
    - Functions:
        - Processes weather and water level data.
        - Returns predictions about potential floods.
7. **Emergency Services**
    - Responders notified in the case of an SOS.
    - Functions:
        - Receive notifications about critical emergencies.

5.3 Methodology

1. Requirement Analysis and Planning

The project begins by identifying the needs of stakeholders, including residents, disaster management teams, and meteorological agencies. Core features like real-time alerts, weather updates, and community engagement tools are defined. A project timeline is established, and technologies such as Python/Django, React.js, and MySQL are selected to ensure efficient development and deployment.

2. System Design

The platform is designed using a client-server architecture to manage communication between the backend, APIs, and the frontend interface. A structured database is developed to store dataon users, alerts, weather, and community posts. UI/UX mockups are created to ensure a user- friendly interface that allows easy access to flood alerts, weather forecasts, and interactive maps.

3. Development and Integration

The development process involves building the backend to handle data from APIs and send real-time notifications. Frontend development focuses on creating a responsive interface for displaying alerts, weather data, and community posts. External APIs like OpenWeatherMap and Google Maps are integrated to provide live updates, and an optional SMS-based alert system is set up to ensure notifications during poor internet connectivity.

4. Testing

Testing begins with unit testing to validate individual components such as notifications, alerts, and API calls. Integration testing ensures smooth interaction between modules, while User Acceptance Testing (UAT) involves gathering feedback from potential users. Identified bugs and usability issues are resolved to improve the platform's reliability and user experience.

5. Deployment:

Frontend components are either hosted on web servers or released as an Android app. Security measures such as data encryption and access controls are implemented, and backup systems are configured to ensure the platform remains operational during disasters.

6. Maintenance and Monitoring

After deployment, the platform is continuously monitored using analytics tools to track performance and detect issues. Logs are maintained to address errors promptly, and updates are performed to keep APIs and security protocols up to date. User feedback is collected through the community forum, ensuring the platform evolves and remains relevant over time.

CHAPTER 6

TECHNOLOGY USED

6.1Details of database or Details About Input to System or Selected Data

The database consists of several key tables, each designed to handle specific types of data.

1. **Users Table**

Column Name	Data Type	Description
UserID	INT (Primary Key, Auto Increment)	Unique identifier for each user.
ContactNumber	VARCHAR(15)	User's phone number for alerts and communication.
Address	VARCHAR(255)	Residential address of the user.
EmergencyContacts	VARCHAR(255)	List of emergency contacts
Name	VARCHAR(100)	Full name of the user.

Table 1. User database

2. **Weather Data Table**

Column Name	Data Type	Description
WeatherID	INT (Primary Key, AutoIncrement)	Unique identifier for each weather data record.
Humidity	FLOAT	Current humidity level at the location.
Rainfall	FLOAT	Rainfall amount (in mm) at the location.
Timestamp	DATETIME	Date and time when the weather data was recorded.
Humidity	FLOAT	Current humidity level at the location.
Temperature	FLOAT	Current temperature at the given location.

Table 2. Weather Data database

3. Volunteer Information Table

Column Name	Data Type	Description
VolunteerID	INT (Primary Key, Auto Increment)	Unique identifier for each volunteer.
Name	VARCHAR(100)	Full name of the volunteer.
Address	VARCHAR(255)	Address of the volunteer.
AvailabilityStatus	VARCHAR(50)	Current availability status (e.g.,Available, Busy).

Table 3. Volunteer Information Database

4. Locations Table

Column Name	Data Type	Description
LocationID	INT (Primary Key, AutoIncrement)	Unique identifier for each location.
UserID	INT (Foreign Key)	Links to the user associatedwith this location.
Address	VARCHAR(255)	Geocoded address of the location.
LocationType	VARCHAR(50)	Type of location (e.g., Home, Office, etc.).

Table 4. Locations Database

5. **Alerts Table**

Column Name	Data Type	Description
AlertID	INT (Primary Key, AutoIncrement)	Unique identifier for each alert.
UserID	INT (Foreign Key)	Links to the user receivingthe alert.
AlertType	VARCHAR(50)	type of alert(e.g.,Warning, SOS, etc.).
AlertMessage	TEXT	The message content of the alert sent to users.
Timestamp	DATETIME	Date and time when the alert was generated.

Table 5. Alerts Database

6. **SOS Requests Table**

Column Name	Data Type	Description
SOSRequestID	INT (Primary Key, Auto Increment)	Unique identifier for each SOS request.
UserID	INT (Foreign Key)	Links to the user making the SOS request.
Timestamp	DATETIME	Date and time when the SOS request was made.
Status	VARCHAR(50)	Current status of the SOS request (e.g,Pending,Resolved)

Table 6. SOS Requests Database

Input to the System

The **Flood Guard communication platform** processes various types of input from users and external systems, which include:

1. User Registration Input

During the registration process, users provide the following information:

- **Name**: To identify users within the community.
- **Email**: For account verification and sending alerts.
- **Phone Number**: To receive SMS notifications, especially crucial during emergencies.
- **Location**: Geographical information to tailor alerts based on their risk of flooding.

2. Flood Alerts Input

Flood alerts can be generated through multiple channels:

- **Automated Input**: The platform integrates with external weather APIs (like OpenWeatherMap) to fetch real-time data regarding rainfall and flood conditions. Alerts are automatically generated based on predefined thresholds (e.g., rainfall exceeding a certain level).
-
- **Manual Input**: Disaster management officials or community leaders can manually enter alerts when automated systems are insufficient or during unexpected events. This ensures that critical information is disseminated promptly.

3. Weather Data Input

The system continuously pulls weather data from:

- **External APIs**: Real-time updates on weather conditions, rainfall, temperature, andforecasts are fetched from services like OpenWeatherMap. This data is vital for assessing flood risks and triggering alerts.
- **Historical Data**: The system may also store historical weather data for analysis andcomparison, helping improve future flood predictions.

4. Community Interaction Input

Users contribute to the platform through:

- **Posts**: Users can share updates, requests for help, or offers of assistance during floods.Each post is tagged as an emergency if it relates to an immediate crisis.
- **Comments**: Community members can comment on posts, fostering dialogue andsupport during emergencies.

5. Notification Preferences

Users can specify their preferred methods for receiving notifications. Options include:

- **Email Notifications**: For detailed updates and alerts.
- **SMS Alerts**: For immediate notifications, especially when internet access may belimited.
- **In-App Notifications**: For users actively engaging with the mobile or webapplication.

6.2 Performance Evaluation Parameters (For Validation)

following performance evaluation parameters can be used for validation to ensure that the system functions effectively and meets the desired objectives:

1. Accuracy

- Definition: The ratio of correctly predicted flood events to the total number of events(both floods and non-floods).
- Importance: High accuracy indicates that the system is effective in predicting floodevents and can be relied upon for accurate information.

2. Precision

- Definition: The proportion of true positive flood predictions to all positive predictionsmade by the system (true positives + false positives).
- Importance: High precision reduces the number of false alarms, ensuring that alerts aremeaningful and trustworthy.

3. Response Time

- Definition: The time taken by the system to process incoming data and send alerts tousers.
- Importance: Quick response times are critical in emergencies to provide timely information to users in flood-affected areas.

4. User Feedback

- Definition: Qualitative assessments collected from users regarding their experiencewith the alerts and the overall system.
- Importance: User insights can highlight strengths and weaknesses, guidingimprovements and enhancing user satisfaction.

5. Coverage Area

- Definition: The geographical area where the system can effectively send alerts to users.
- Importance: Ensures that all potential flood-affected areas are monitored and alerted,maximizing user safety.

6. Scalability

- Definition: The system's capability to handle increasing numbers of users and alertswithout degradation in performance.
- Importance: A scalable system can grow as the user base expands, ensuring continuouseffectiveness.

7. Volunteer Response Time

- Definition: The average time taken by volunteers to respond to SOS messages fromusers in distress.
- Importance: Quick volunteer response is critical in emergencies and can significantlyimpact user safety.

8. Integration with External APIs

- Definition: Evaluating how well the system integrates with weather and emergencyresponse APIs.
- Importance: Effective integration ensures timely access to relevant data, improving theaccuracy and reliability of alerts.

9. User Engagement Metrics

- Definition: Metrics that indicate how actively users are interacting with the system,such as the number of active users and interaction frequency.
- Importance: High user engagement often correlates with system effectiveness and usertrust, suggesting that users find the system valuable.

6.3 Software and Hardware Requirements

Hardware Requirements

1. **User Devices:**
    - Smartphones/Tablets:
        - Operating System: Android (minimum version 7.0) or iOS (minimum version 12.0).
        - Processor: Quad-core or higher.
        - RAM: Minimum of 2 GB for smooth operation.
        - Storage: At least 100 MB of free space for app installation and data storage.
2. **Server Requirements:**
    - Web Server:
        - Processor: Dual-core or higher.
        - RAM: Minimum of 8 GB.
        - Storage: SSD with a minimum of 500 GB to support database and application files.
        - Network Interface: 1 Gbps Ethernet for fast data transmission.
3. **Backup and Recovery System:**
    - External Hard Drive/Cloud Storage:
        - For data backup and recovery, ensure at least 1 TB of storage capacity.

Software Requirements

1. **Mobile Application Development:**
    - Programming Languages:
        - Android: Kotlin or Java.
        - iOS: Swift or Objective-C.
    - Development Frameworks:
        - Cross-platform: Flutter or React Native (for both Android and iOS).
2. **Backend Development:**
    - Server-side Language:

        - Node.js, Python (Django or Flask), or Java (Spring Boot).
    - Database Management System:
        - Relational Database: PostgreSQL or MySQL.
        - NoSQL Database: MongoDB (for handling unstructured data).
    - API Development:
        - RESTful API or GraphQL for communication between frontend and backend.

3. **Data Analytics and Machine Learning:**
    - Libraries/Frameworks:
        - Python: Pandas, NumPy, Scikit-learn, TensorFlow, or PyTorch for predictive modeling.
    - Data Visualization:
        - Libraries such as Matplotlib or Plotly for generating graphs and visual representations of data.
4. **Web Development:**
    - Frontend Frameworks:
        - React.js or Angular for creating a web interface for administrators and volunteers.
    - Backend Framework:
        - Node.js or Django for handling server-side operations.
5. **Development and Collaboration Tools:**
    - Version Control:
        - Git (with platforms like GitHub or GitLab).
    - Project Management:
        - Tools like Trello, Jira, or Asana for task tracking and collaboration.
6. **Testing and Deployment:**
    - Testing Frameworks:
        - Jest for JavaScript testing, JUnit for Java, or Pytest for Python.
7. **User Interface Design:**
    - Design Tools:
        - Figma, Adobe XD, or Sketch for creating app mockups and prototypes.

CHAPTER 7

REFERENCES

References:

[1]A. Bhattacharya, R. Sharma, and S. S. Shukla, "A comprehensive review of flood prediction models using machine learning and deep learning," *IEEE Access*, vol. 10, pp. 12345-12360, 2022. doi: 10.1109/ACCESS.2022.3181675.

[2]S. Asadi and M. K. Khosravi, "Development of an IoT-based flood monitoring and early warning system using wireless sensor networks," *IEEE Sensors Journal*, vol. 22, no. 1, pp. 876-884, Jan. 2022. doi: 10.1109/JSEN.2021.3078226.

[3] S. Jha, D. Gupta, and M. Gupta, "Flood management system using cloud computing and Internet-of-Things," *IEEE Access*, vol. 10, pp. 10146661, 2021. doi: 10.1109/ACCESS.2021.10146661.

[4]V. Jadhav and S. Ghosh, "Flood location, management, and solution (FLMS): A flood prediction and management system for Kurla," *IEEE Access*, vol. 10, pp. 10392346, 2021. doi: 10.1109/ACCESS.2021.10392346.

[5]V. Agarwal and A. Agrawal, "Flood forecasting using machine learning: A review," *IEEEAccess*, vol. 9, pp. 9528099, 2021. doi: 10.1109/ACCESS.2021.9528099.

[6] M. S. Khan, F. Jabeen, and H. A. Khattak, "IoT-based flood early warning system for effective disaster management," *IEEE Access*, vol. 10, pp. 10559355, 2021. doi: 10.1109/ACCESS.2021.10559355.