CMPE 492 / SENG 492

Senior Design Project II

Final Report

Burak Güçlü

Gizem Yüksel

İrem Özyurt

Zeynep Sude Bal

Advisor

Emin Kuğu

26.05.2025

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1. Project Purpose, Scope and Problem Definition

Natural disasters such as earthquakes, floods and fires are events that threaten the safety of society, require rapid response and can have long-lasting effects. One of the most important needs in the aftermath of such disasters is to establish effective communication with disaster victims, to evaluate calls for help and to deliver accurate information to the relevant units.

This project develops a communication and monitoring system that aims to determine the security status of individuals after a disaster, to collect emergency aid requests and to centrally manage data on the disaster area. The system consists of three main components: sensors, mobile application and web application.

Sensors detect events such as earthquakes, floods and fires, triggering the system.

The mobile application sends push notifications to users in the areas where the event occurs, asking whether they are safe or not. Users provide feedback with the options “I am safe” or “I am not safe”. With the “I am not safe” option, notifications are sent both to authorized units.

The chat module in the application allows users to communicate with authorized teams in real time. This provides an interaction environment that is not limited to automatic notifications.

Thanks to the map module, users can access and receive directions to the 5 nearest safe points such as hospitals, pharmacies and parks.

The web application, another important component of the project, serves as a control panel for authorized institutions and disaster management teams. Through this panel:

Users' “safe/not safe” responses can be monitored instantly,

Sensor activity can be monitored, and which disaster is occurring in which region,

Incoming requests for assistance can be prioritized and evaluated.

This integrated system is designed based on scenarios where GSM and internet infrastructure is operational and aims for uninterrupted information flow over the existing communication infrastructure. It aims both to protect the lives of individual users and to enable disaster response teams to act in a more organized and conscious manner.

1.2. Organization of the Report

This report comprehensively describes the technical infrastructure of the developed post-disaster communication system, its objectives, the implementation process, the results obtained and future recommendations. The report consists of the following chapters:

Chapter 2: System Design and Development

In this chapter, the general architecture of the system, the sensors used, the functioning of mobile and web applications, the software/hardware technologies and tools used are detailed. In addition, data management strategies and the security approach of the system are discussed.

Chapter 3: Impacts of the Project and Contemporary Issues

The social, environmental, economic and global impacts of the project are discussed; current problems in the field of post-disaster communication and the solutions brought to these problems within the scope of the project are evaluated.

Chapter 4: Testing Process and Results

The test plans, test types, evaluation of the results, detected errors and performance analysis are presented in this chapter to ensure the accuracy and reliability of the developed system.

Chapter 5: Conclusion and Future Work

A general evaluation of the project is made, the contributions of the system for users and authorities are summarized and suggestions for future improvements are presented.

2. System Design and Development

2.1. System Architecture and Basic Components

The post-disaster communication system's final architecture is organized using a layered, modular design that guarantees extensibility, scalability, and maintainability. The five main components that make up the system are Core, MobileApp, WebApp, SensorData, and NotificationSystem. Each package contains submodules and encapsulates particular duties that support the smooth operation of the entire system.

The system is supported by the Core Package. The three essential modules are Configurations, Utils, and Entities. User, Sensor, Notification, and EmergencyContact are among the main data models defined in the Entities module that are utilized across the system. These models stand in for the fundamental components of the system for communicating disasters. For example, whereas the Sensor entity encapsulates the metadata and measurements from Internet of Things devices like water level indicators, temperature monitors, and seismic sensors, the User entity holds contact and personal information along with login credentials. While the EmergencyContact entity allows users to predefine trusted contacts to be contacted in crucial situations, the Notification entity is in charge of encapsulating alert messages sent during emergencies. The Utils module offers crucial support functions for validity checks, error logging, and data formatting (such as JSON serialization). Lastly, the Configurations module makes it simple to modify the system to fit various operating scenarios by including global constants and parameters like sensor thresholds, API keys, and system timeouts.

The features that end users can access through the mobile interface are the main focus of the MobileApp Package. It has modules for UserManagement, NotificationHandling, and Authentication. The Authentication module controls session tokens to stop unwanted access and uses Firebase Authentication to enable safe user registration and login. The UserManagement module enables users to maintain and amend their emergency contacts and personal data. In times of emergency, this module makes sure that responders have access to current and correct contact information. When a disaster is discovered, alert messages are intended to be received and displayed by the NotificationHandling module. When asked, "Are you safe?" users have the option to answer, "I am safe," or "I am not safe." The system automatically notifies designated contacts and passes information to emergency agencies if no answer is received within a predetermined time frame or if a user reports feeling unsafe.

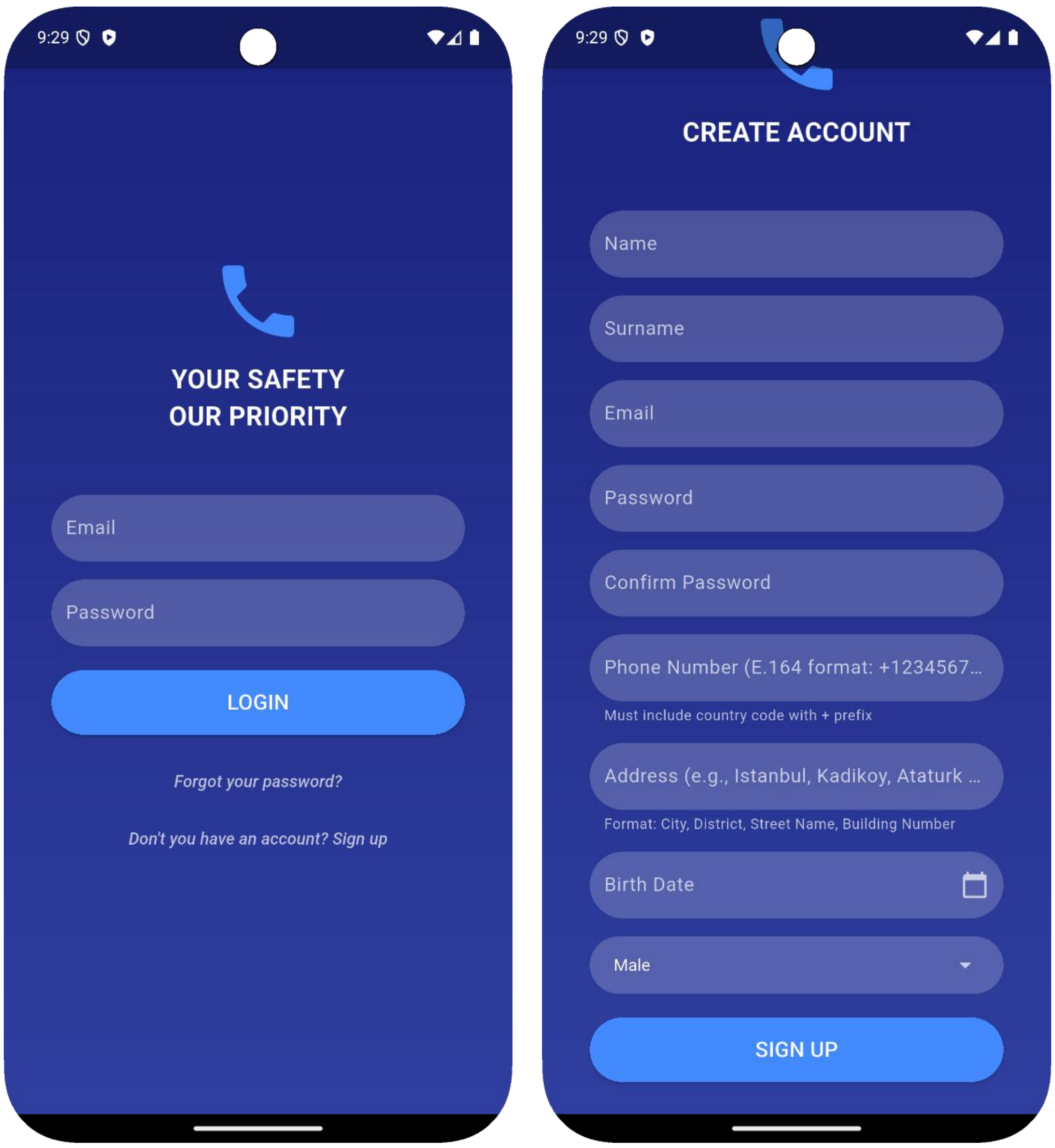
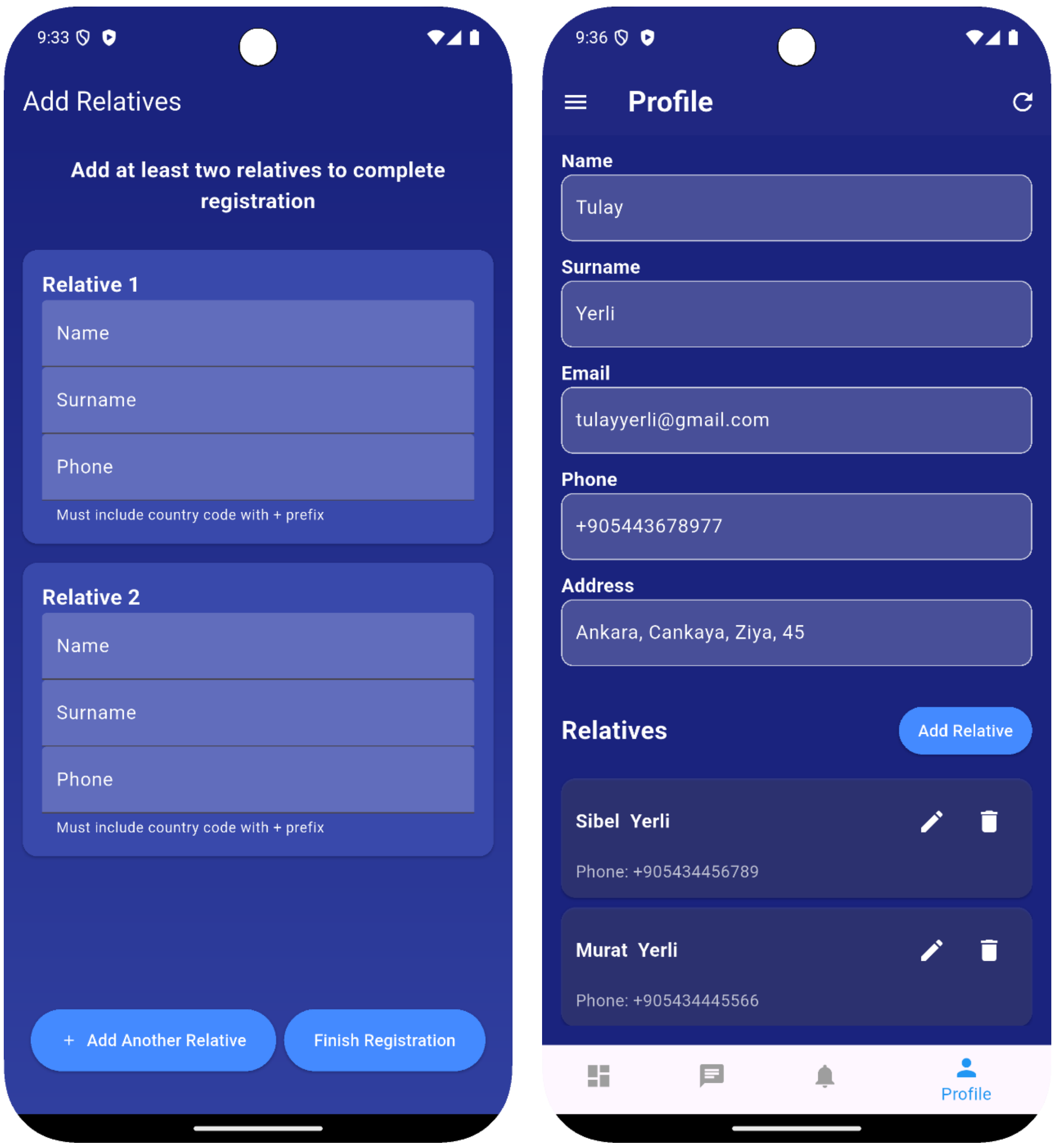

9:29
YOUR SAFETY
OUR PRIORITY
Email
Password
LOGIN
Forgot your password?
Don't you have an account? Sign up
9:29
CREATE ACCOUNT
Name
Surname
Email
Password
Confirm Password
Phone Number (E.164 format: +1234567…
Must include country code with + prefix
Address (e.g., Istanbul, Kadikoy, Ataturk …
Format: City, District, Street Name, Building Number
Birth Date
Male
SIGN UP

Figure 1 - Login Page and Sign-Up Page of the Mobile Application



9:33

Add Relatives

Add at least two relatives to complete registration

Relative 1

Name

Surname

Phone

Must include country code with + prefix

Relative 2

Name

Surname

Phone

Must include country code with + prefix

+ Add Another Relative

Finish Registration

9:36

Profile

Name

Tulay

Surname

Yerli

Email

tulayyerli@gmail.com

Phone

+905443678977

Address

Ankara, Cankaya, Ziya, 45

Relatives

Add Relative

Sibel Yerli

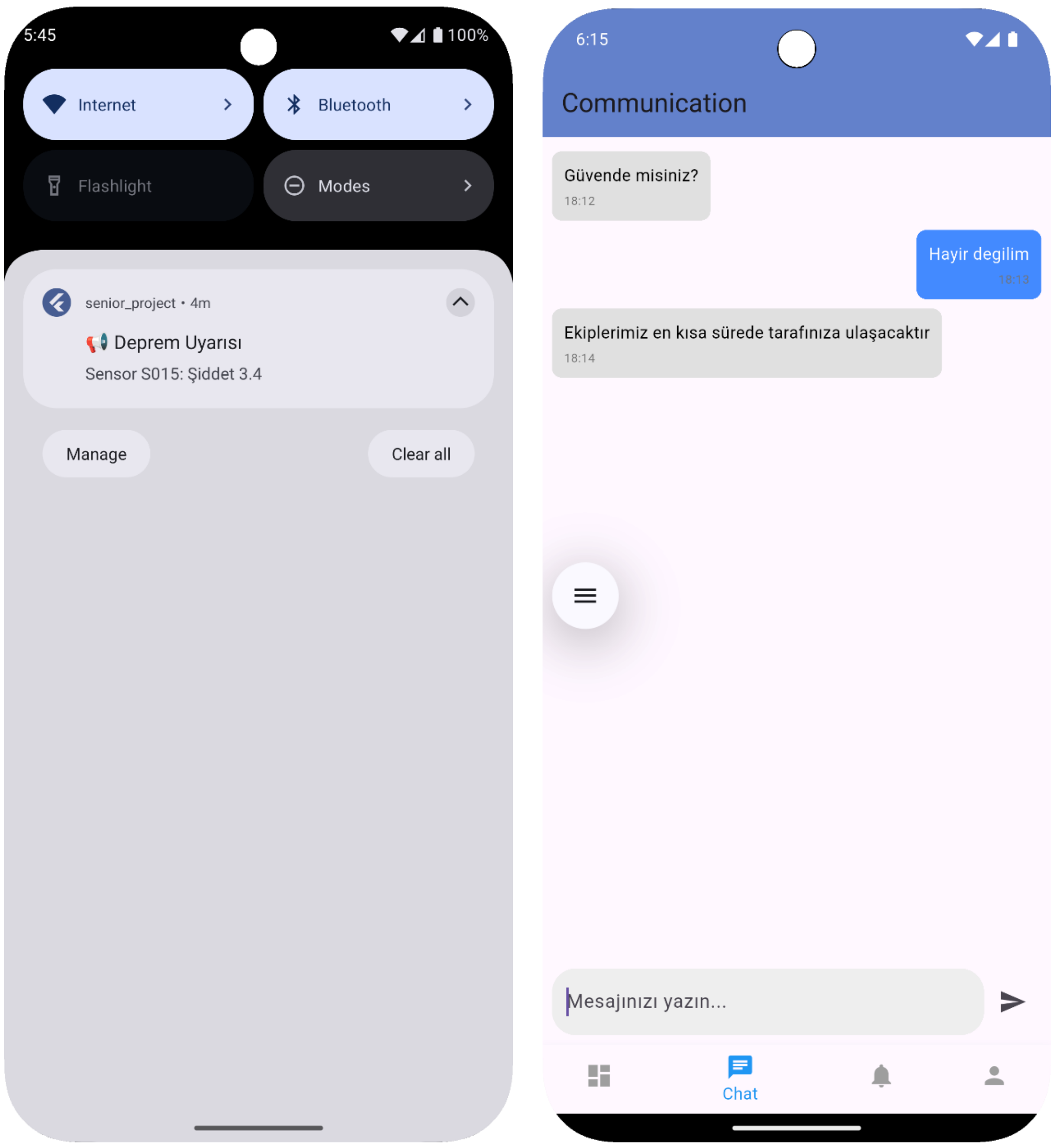
Phone: +905434456789

Murat Yerli

Phone: +905434445566

Profile

Figure 2 - Add Relative Page and Profile Page of the Mobile Application



senior_project • 4m
📢 Deprem Uyarısı
Sensor S015: Şiddet 3.4

Communication

Güvende misiniz?
18:12

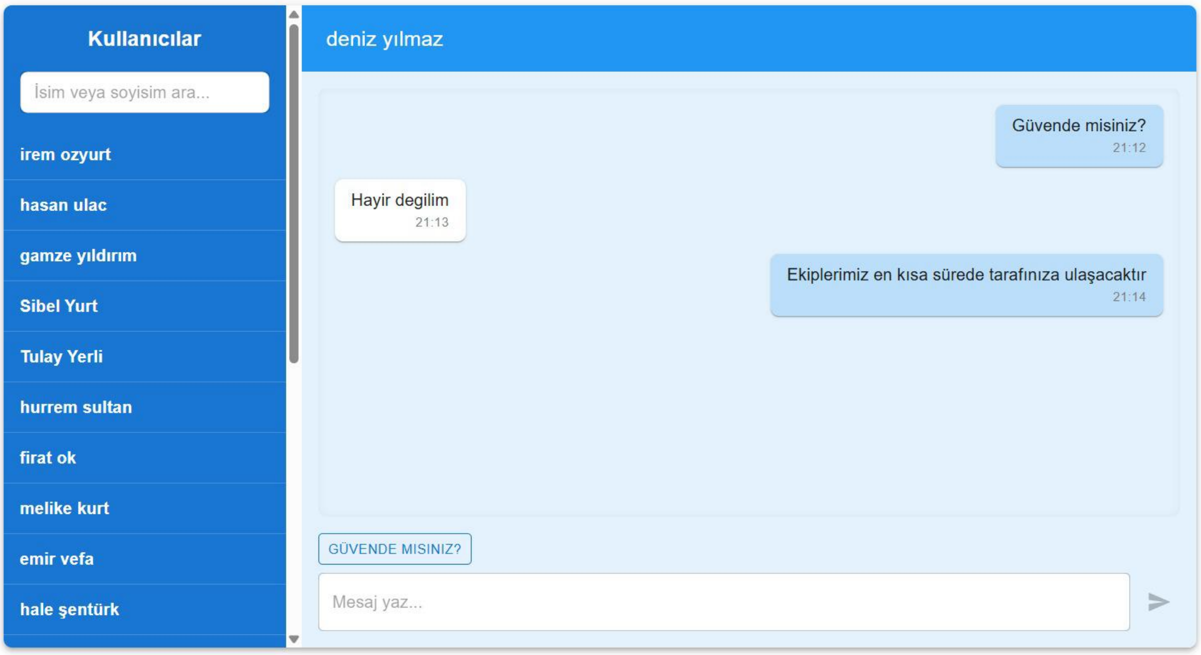
Hayir degilim
18:13

Ekiplerimiz en kısa sürede tarafınıza ulaşacaktır
18:14

Mesajınızı yazın...

Figure 3 - Notification and Communication Page of the Mobile Application

The WebApp Package acts as the monitoring and administrative interface for authorized individuals, including system administrators and emergency responders. Modules for User Responses, Emergency Dispatch, AdminManagement, IncidentMonitoring, and Authentication are all included in this package. Role-based access controls are assigned by the authentication module, which also makes sure that only verified personnel can use the platform. Interactive dashboards and maps are among the graphical representations of real-time data from environmental sensors that IncidentMonitoring provides. Responders are able to identify who is safe, who is not, and who has not responded at all as the UserResponses module gives them insight into user reactions to crisis notifications. EmergencyDispatch automatically notifies the relevant emergency agencies when users report or detect danger. By enabling back-end control, AdminManagement gives administrators the ability to manage user roles and permissions, examine logs, and make configuration changes.

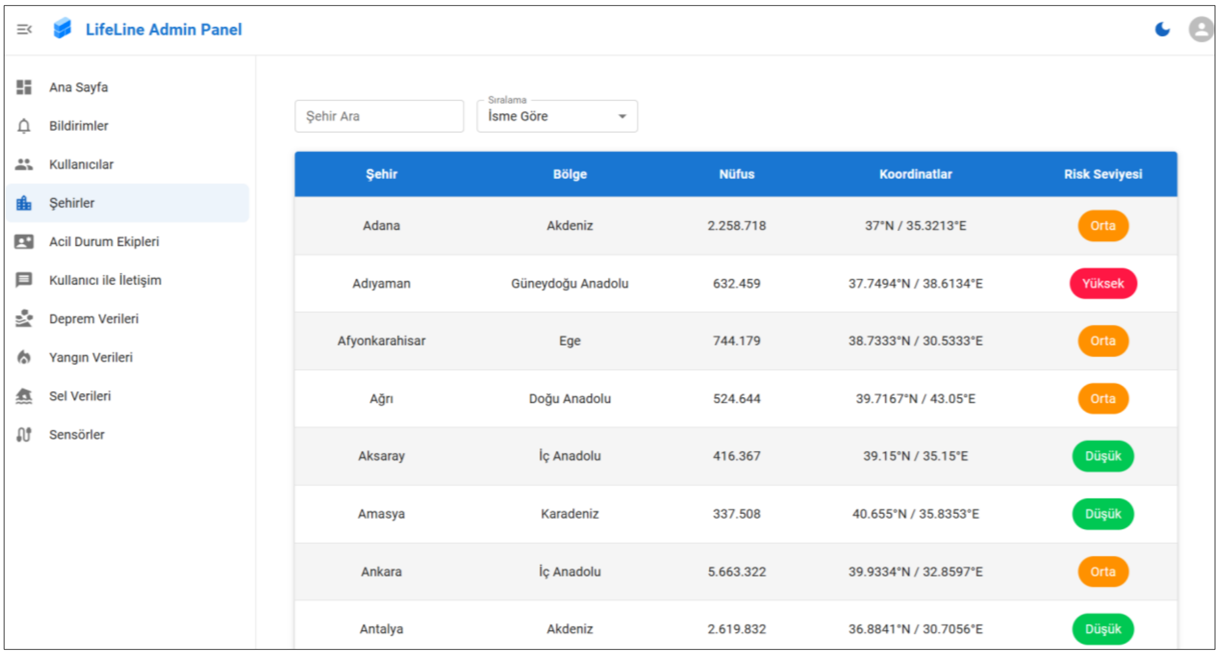


Güvende misiniz? 21:12

Hayir degilim 21:13

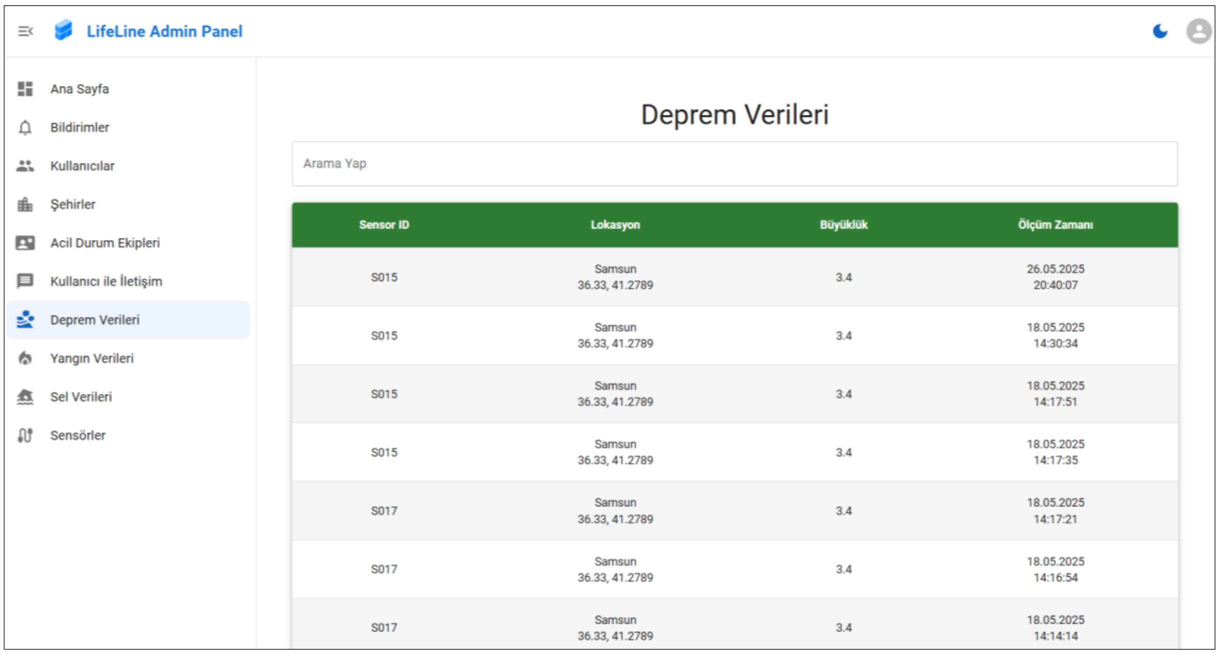
Ekiplerimiz en kısa sürede tarafınıza ulaşacaktır 21:14

Figure 4 - Communication Page of the Web Application



Şehir	Bölge	Nüfus	Koordinatlar	Risk Seviyesi
Adana	Akdeniz	2.258.718	37°N / 35.3213°E	Orta
Adıyaman	Güneydoğu Anadolu	632.459	37.7494°N / 38.6134°E	Yüksek
Afyonkarahisar	Ege	744.179	38.7333°N / 30.5333°E	Orta
Ağrı	Doğu Anadolu	524.644	39.7167°N / 43.05°E	Orta
Aksaray	İç Anadolu	416.367	39.15°N / 35.15°E	Düşük
Amasya	Karadeniz	337.508	40.655°N / 35.8353°E	Düşük
Ankara	İç Anadolu	5.663.322	39.9334°N / 32.8597°E	Orta
Antalya	Akdeniz	2.619.832	36.8841°N / 30.7056°E	Düşük

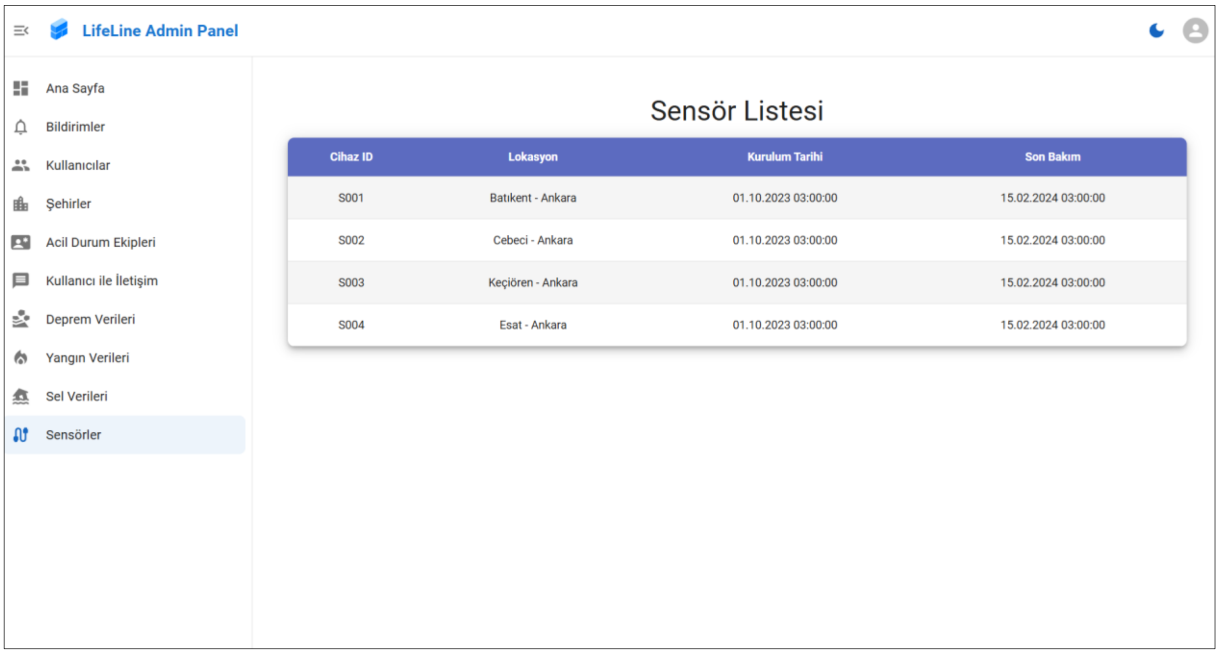
Figure 5 - Cities Page of the Web Application



Deprem Verileri

Sensor ID	Lokasyon	Büyüklük	Ölçüm Zamanı
S015	Samsun 36.33, 41.2789	3.4	26.05.2025 20:40:07
S015	Samsun 36.33, 41.2789	3.4	18.05.2025 14:30:34
S015	Samsun 36.33, 41.2789	3.4	18.05.2025 14:17:51
S015	Samsun 36.33, 41.2789	3.4	18.05.2025 14:17:35
S017	Samsun 36.33, 41.2789	3.4	18.05.2025 14:17:21
S017	Samsun 36.33, 41.2789	3.4	18.05.2025 14:16:54
S017	Samsun 36.33, 41.2789	3.4	18.05.2025 14:14:14

Figure 6 - Earthquake Data Page of the Web Application



Sensör Listesi

Cihaz ID	Lokasyon	Kurulum Tarihi	Son Bakım
S001	Batıkent - Ankara	01.10.2023 03:00:00	15.02.2024 03:00:00
S002	Cebeci - Ankara	01.10.2023 03:00:00	15.02.2024 03:00:00
S003	Keçiören - Ankara	01.10.2023 03:00:00	15.02.2024 03:00:00
S004	Esat - Ankara	01.10.2023 03:00:00	15.02.2024 03:00:00

Figure 7 - Sensors Page of the Web Application

The SensorData Package is in charge of controlling real-time data flow and interfacing with actual environmental sensors. It consists of modules for data collection, data processing, and storage. The DataCollection module creates an ongoing data stream from Internet of Things devices, recording variables like water levels, seismic activity, and temperature. Incoming data is assessed by the DataProcessing module in relation to predetermined safety thresholds. The system recognizes an abnormality as a risk condition and sends out the proper notifications if it is discovered, such as when the magnitude of an earthquake surpasses a certain value. In order to facilitate both retrospective analysis and future use in machine learning-based prediction systems, the Storage module saves past sensor data to a cloud database.

The NotificationSystem Package, which is the last package, controls the creation, distribution, and recording of disaster notifications. Using sensor inputs, the AlertGeneration module generates structured messages that include the kind of hazard, its location, and its level of severity. Through integration with email, SMS, and push notifications, the MessageDelivery module makes sure that users and their emergency contacts receive messages via dependable channels. Finally, the ResponseTracking module helps emergency services prioritize resources and make decisions by tracking user replies in real time and recording their activities.

The system design as a whole facilitates high cohesion and loose coupling, allowing for independent development, testing, and deployment of components. In order to ensure strong integration and extensibility for upcoming modules like wearable device integration or AI-based risk prediction, interactions between mobile, web, and sensor subsystems are carried out via thoroughly described RESTful APIs.

2.2. Technologies and Tools Used

The project employed a full-stack software development methodology with an emphasis on open-source, scalable, and cloud-based technologies in order to create a contemporary, real-time catastrophe management platform. Google's Flutter UI toolkit, which facilitates cross-platform development for both iOS and Android, was used to create the mobile application. Flutter was selected because of its robust community, expressive user interface, and excellent performance.

Using React.js, a popular JavaScript package that offers a component-based architecture for creating responsive and maintainable user interfaces, the web application was constructed. The ecosystem of React makes it possible to quickly create the dynamic dashboards and real-time data visualizations that emergency responders need.

Node.js with the Express framework powers the backend infrastructure, offering lightweight and effective HTTP server functionality. It manages routing, integrates middleware, and serves as a conduit between the cloud database and front-end elements.

The system uses Firebase (a NoSQL database), Firebase Auth, and Firebase Cloud Functions from Google Firebase for data storage, authentication, and real-time communication. With little operating overhead, Firebase provides high availability, robust security features, and smooth user data syncing. It has real-time update capabilities, which are essential for sending out emergency notifications and promptly getting user feedback.

Git and GitHub were used to manage version control and collaborative development, guaranteeing a clean codebase and simple integration of various development branches. Using Lucidchart and Draw.io, design models were produced in accordance with accepted UML standards. Postman was used for testing and validating the API, while Firebase Emulator Suite was utilized to test database interactions and cloud functionality in a secure local setting.

These technologies and tools were chosen due to their community support, maturity, and conformity to contemporary best practices. They gave the system a solid basis for quick development, simple upkeep, and future scalability.

2.3. Data Management and Security Approach

Secure and effective data management was emphasized because user data is sensitive, and the system is vital in emergency situations. The system's main data store is Firebase Firestore, which groups data into collections like users, sensorData, alerts, notifications, and messages. To guarantee traceability and consistency, every data item is timestamped and uniquely identified. Both current monitoring and historical analysis are made possible by the continuous streaming and real-time storing of sensor data.

All the layers of the system were integrated with security. Secure JWT tokens are used to manage session states, while Firebase Auth handles authentication through email and password protocols. This keeps user data safe from illegal access and guarantees that only authenticated users can communicate with the system. Role-based access control was applied to all API endpoints, limiting authorized people to do sensitive tasks like examining user responses or sending out emergency notifications.

In order to enforce database-level access permissions, Firebase Security Rules were established. As an example, users are limited to reading and writing their own messages, emergency contacts, and profiles. Malicious data injection is prevented since only verified and registered devices are able to upload sensor data.

There is a strong error-handling approach in the system. Every API response provides structured error codes along with insightful remarks, allowing the frontend to give consumers useful feedback. Backend logs are kept in order to track system malfunctions and spot irregularities.

Furthermore, efforts were made to guarantee privacy and data integrity, particularly with regard to user status and real-time location. Advanced cryptographic capabilities and compliance with data protection laws like the GDPR are planned for future upgrades, even if the current version does not offer encryption-at-rest for all records.

In overall, the data management approach guarantees that all user and system data is reliably processed, safely stored, and easily accessible when required even during high-load emergency situations. User trust is increased, data breaches are avoided, and the disaster response system's dependability is preserved thanks to the security measures in place.

3. Project Impacts and Contemporary Issues

3.1. Global, Economic, Environmental, and Social Impacts

The engineering solutions developed in this project provide significant and multifaceted contributions across global, economic, environmental, and social dimensions, especially in the context of post-disaster communication systems.

- **Global Impact**: Natural disasters such as earthquakes, floods, and wildfires affect millions of people worldwide. The proposed system is designed to be modular and scalable, making it adaptable to the unique needs of different countries and regions. Its cloud-based, cross-platform architecture allows for deployment across diverse socio-geographic environments. As such, it contributes to building resilient global communities by supporting early warning, rapid communication, and coordinated emergency response.
- **Economic Impact**: Traditional disaster response systems often require high infrastructure and personnel costs. This system replaces many manual tasks with automated alert generation, real-time data collection, and user feedback handling. It reduces resource consumption by using existing cloud services (Firebase), free frameworks (Flutter, React), and sensor APIs, thus lowering both development and operational costs. The efficient use of open-source and cloud-native technologies contributes to a more economically sustainable disaster management model.
- **Environmental Impact**: Environmental monitoring is an integral part of disaster mitigation. By utilizing IoT-based sensors (seismic, temperature, and water level), the system continuously evaluates risk indicators. This real-time data collection enables early intervention in cases such as flash floods or fire outbreaks, ultimately minimizing environmental damage. Moreover, the system promotes responsible energy and data usage by activating alerts only when sensor thresholds are breached, thereby avoiding unnecessary system loads.
- **Social Impact**: The solution enhances public safety and strengthens community resilience. Through mobile notifications like “Are you safe?”, individuals can easily report their safety status, allowing emergency responders to prioritize rescue efforts. It also notifies users' emergency contacts, providing psychological support during crises.

Moreover, the system fosters trust in emergency management infrastructure and promotes a more informed and prepared society.

3.2. Current Discussions and Issues Related to the Project

The project touches on several contemporary issues at the intersection of software engineering, public safety, and emerging technologies:

- **Data Privacy and Ethics**: Collecting and storing sensitive personal data—such as real-time location, user status, and contact lists—raises critical ethical and legal questions. To address this, the system integrates secure authentication and authorization mechanisms using Firebase Auth and JWT tokens. However, further enhancements, such as encryption-at-rest and GDPR-compliant data handling policies, are important for future versions.
- **Infrastructure Dependency and Offline Resilience**: A key limitation of many disaster systems is reliance on internet connectivity, which is often disrupted during major events. While the current system depends on online cloud infrastructure, one of the major areas of future development is **offline-first capability**. This includes supporting communication through Bluetooth mesh networks or peer-to-peer Wi-Fi, ensuring local message passing even in the absence of centralized servers or internet access.
- **User Accessibility and Inclusivity**: In emergency situations, users must be able to interact with the system quickly and without confusion. The mobile app is designed with a simple and accessible interface, but future iterations could introduce support for multiple languages, voice input, and assistive features to accommodate elderly, disabled, or low-literacy users.
- **Technology Stack**: The project utilizes state-of-the-art technologies including:
  - **Flutter** for cross-platform mobile application development.
  - **React.js** for responsive web interface.
  - **Node.js (Express)** for backend API and middleware.
  - **Firebase (Firestore, Auth, Cloud Functions)** for real-time data synchronization and user management.

These choices align with current best practices in modern full-stack software development.

- **Use of Academic and Technical Resources**: Design decisions were informed by international standards such as IEEE 830 (Requirements), IEEE 1016 (Software Design), and IEEE 1012 (Verification & Validation). Additional references included academic literature on IoT in disaster management, open-source documentation from Firebase and React ecosystems, and existing research on modular and scalable software architecture.

4. Testing Process and Results

In this section, the testing process applied to the Post-Disaster Communication System is presented in detail. The tests were conducted in accordance with the previously prepared test plan and the extent to which the system met both functional and non-functional requirements was evaluated. The testing process aimed to evaluate the accuracy, reliability, usability and performance of the system holistically.

Since the system undertakes vital responsibilities such as performing real-time risk detection in post-disaster situations, communicating with users and ensuring coordination of emergency response units; testing its accuracy, stability and durability is of great importance. In this direction, various testing methods such as unit testing, integration testing, system testing, performance testing, security testing, compatibility testing and user acceptance testing (UAT) were used. During the tests, various usage conditions such as realistic disaster scenarios created through IoT sensor simulators, weak network connections, high user load and different device types were also taken into consideration.

Throughout the testing process, 13 main scenarios and dozens of sub-test cases were progressed; functional and user experience-focused checks were performed on both the mobile application and the web-based management panel. “Are you safe?” Critical functions such as whether notifications are delivered on time, whether information messages are sent to emergency contacts and official institutions, and whether user location information is displayed correctly on the map were examined in detail in these tests. In addition, operational modules of the system such as user registration processes, profile and contact management, alarm mechanisms and incident management tools were also tested one by one.

The results obtained at the end of this test process provide important findings regarding the overall quality of the system and also reveal future improvement opportunities. It was documented whether each test case passed or not, the problems in the failed tests were analyzed and possible error sources and solution suggestions were developed. In the following subheadings, the tests applied, the results obtained, performance and reliability evaluations, and the suggestions offered for further development of the system are discussed in detail.

4.1. Test Plan and Applied Tests

A comprehensive test plan was created to evaluate the accuracy, reliability, security and user-friendliness of the Post-Disaster Communication System. This test plan aimed to test all modules of the system separately and together. The types of tests applied were carried out in a structure that started with unit tests, continued with integration and system tests, and completed with performance, security and user acceptance tests. Each test was conducted by simulating real disaster conditions and user behaviors.

During the tests, all system components, including Flutter-based mobile application, React.js-based web management panel and IoT sensor simulations, were evaluated. Thanks to the tests conducted according to different network conditions, high user load, multi-device compatibility and different scenarios, the system's resilience to both ordinary and extraordinary situations was measured.

Table 1 - Applied Test Types and Representative Test Scenarios

Test Type	Description	Representative Scenario
Unit Testing	It is carried out to check the accuracy of individual modules and functions.	Completion of user registration process in the mobile application with valid information.
Integration Testing	It is testing the data flow and compatibility between modules.	Correct transfer of data from IoT sensors to mobile and web applications.
System Testing	It is the test that evaluates whether the entire system works with end-to-end scenarios. it to the relevant units.	Triggering the “Are you safe?” notification, receiving the responses and reporting
Performance Testing	It is carried out to evaluate system durability, response time and scalability.	Measuring the response time of the system in high user and data traffic.
Usability Testing	It measures how easily and effectively the system interface can be used by users.	Real users performing location sharing and requesting help in the application.
Security Testing	Unauthorized access, data protection and system vulnerabilities were tested.	Automatic exit control at the end of the session period in the web panel.
Compatibility Testing	The system was tested under different devices, browsers and network conditions.	It was checked whether the application worked stably on Android and iOS devices.
Acceptance Testing (UAT)	Validation of the functionality and ease of use of the system by real users.	AFAD volunteers test the application with real scenarios.

Table 2 - Test Results for Test Cases

Test Case ID	Test Case Description	Test Steps	Preconditions	Test Data	Expected Result	Actual Result	Status
TC-IoT-001-001	Verification of data acquisition below earthquake threshold.	1. Start the IoT sensor simulator with a Richter 2.0 input 2. Monitor the sensor data flow on the web management panel 3. Observe the logs and real-time data visualization for 5 minutes	IoT simulator active; system is running; earthquake sensor functioning	Seismic activity at Richter level 2.0	Sensor data received regularly, matches expected values; no alarm is triggered	Data matched simulator output, no alarms triggered	Pass
TC-IoT-001-002	Alarm generation for high seismic data.	1. Input simulated seismic data of Richter 6.0 using the IoT sensor simulator 2. Observe the alarm status section on the web panel 3. Check mobile app for incoming alert	IoT simulator active; system is live; mobile app installed	Seismic activity at Richter level 6.0	Alarm is triggered on panel; mobile notification is delivered to user	Alarm shown on web; mobile alert successfully received	Pass
TC-IoT-001-003	Flood alarm with critical water level.	1. Start simulator to send critical water level data 2. Monitor the water level graph on web panel 3. Check for flood alarm and user notifications	Simulator running; water sensor enabled; system active	Critical water level input	Flood alarm is triggered; users are notified via mobile app	Flood alarm raised; notifications delivered as expected	Pass
TC-NOT-002-001	Notification delivery upon alarm trigger.	1. Trigger a simulated earthquake using the sensor system 2. Check user devices for 'Are you safe?' notifications 3. Confirm timely delivery and content accuracy	Mobile app installed and logged in; simulated earthquake triggered	Simulated hazard trigger	Users receive 'Are you safe?' prompt with timestamp	All test users received prompt with expected message	Pass
TC-NOT-002-002	Processing of 'I'm safe' response.	1. User responds to prompt by selecting 'I'm safe' 2. Admin logs into web panel	Notification received; user is active; system running	User selects 'I'm safe'	Status updated to 'Safe' on the admin panel with timestamp	Correctly reflected in admin panel	Pass

		3. Admin checks updated user status on dashboard					
TC-NOT-002-003	Processing of 'I'm not safe' response and emergency alerts.	1. User selects 'I'm not safe' in app 2. Admin opens panel to verify status change 3. Check if emergency contact and emergency units receive alert	Emergency contact and unit info configured; system live	User selects 'I'm not safe'	Status updated to 'Not Safe'; alerts sent with location info	Status change visible; alerts received by recipients	Pass
TC-EMC-003-001	Notification to emergency contact after 'Not Safe' response.	1. Send safety prompt to user 2. User selects 'Not Safe' 3. Check if emergency contact receives notification	Mobile app active; emergency contact registered	'Not Safe' response by user	Emergency contact receives alert with user's location and timestamp	SMS received by contact as expected	Pass
TC-EMC-003-002	Notification to emergency contact after no response.	1. Send safety notification 2. User gives no response within timeout (e.g., 60 seconds) 3. Check contact for fallback alert	Emergency contact defined; no user response within defined time	No user interaction	Emergency contact receives alert with last known location	Expected fallback message delivered	Pass
TC-EMC-003-003	Notification sent to multiple emergency contacts.	1. Add multiple emergency contacts 2. User responds 'Not Safe' or no response 3. Monitor all registered contacts for notifications	Multiple contacts defined in app; simulated emergency	'Not Safe' or timeout from user	All contacts receive alert with same location and timestamp	Messages received by all listed contacts	Pass
TC-OFF-004-001	Notification sent to emergency unit after 'Not Safe' response.	1. User selects 'Not Safe' in app 2. System sends data to emergency unit 3. Monitor API log or endpoint receipt	System configured with emergency unit API; response captured by system	'Not Safe' user input	API call made with user location and risk type	Correct request sent to emergency endpoint	Pass
TC-OFF-004-002	Notification sent to emergency unit after no response.	1. Trigger notification 2. Wait without user input 3. Monitor emergency unit endpoint	Emergency contact and unit configured; app running	No user response	System sends alert to emergency unit after timeout	Notification delivered correctly after no response	Pass

TC-OFF-004-003	Correct emergency unit receives alert based on hazard type.	1. Trigger simulated earthquake 2. Verify alert goes to AFAD 3. Trigger flood scenario and verify fire brigade alert	Sensor data mapped to correct units; routing logic in place	Earthquake and flood data	Earthquake → AFAD, Flood → Fire Brigade	Unit-specific routing verified for each hazard	Pass
TC-MOB-005-001	Successful user registration with valid data.	1. Launch mobile app 2. Enter valid email and password 3. Submit registration form	App installed; network available	Valid user input	User registered and session initialized	User created successfully; session active	Pass
TC-MOB-005-002	Registration fails with invalid email.	1. Open app registration 2. Enter invalid email (e.g., 'user@') 3. Submit form	App available; internet on	Invalid email format	Error shown: 'Enter valid email'	Validation error correctly triggered	Pass
TC-MOB-005-003	Registration fails with already used email.	1. Enter existing registered email 2. Input valid password 3. Submit registration form	Account already exists with given email	Previously registered email	System shows: 'Email already in use'	Duplicate email check passed	Pass
TC-MOB-005-004	Successful login with valid credentials.	1. Open the mobile app 2. Enter valid registered email and password 3. Tap on 'Login' button 4. Wait for dashboard to load	User account already registered; internet connection active	Valid login credentials	User successfully logs in and redirected to main screen	Logged in successfully	Pass
TC-MOB-005-005	Login attempt with invalid password.	1. Open app 2. Enter registered email and wrong password 3. Tap on 'Login' 4. Observe feedback message	User account exists; incorrect password entered	Wrong password for existing email	Login fails with error: 'Incorrect email or password'	Error displayed, login blocked	Pass
TC-MOB-006-001	Profile information displayed correctly.	1. Login to mobile app 2. Navigate to 'My Profile' 3. Review displayed name, email, phone	User has profile data stored in system	User profile (name, email, phone)	Displayed info matches stored data	Correct info shown in profile	Pass
TC-MOB-006-002	User updates profile info successfully.	1. Open profile section in app 2. Click 'Edit'	User logged in; app running	Valid new phone number	Confirmation message shown; updated info reflected	New number saved correctly	Pass

		3. Change phone number 4. Save and confirm					
TC-MOB-006-003	Invalid profile update is rejected.	1. Attempt to update phone with invalid input (e.g., letters) 2. Submit change 3. Observe error feedback	App running; user logged in	Invalid phone format (e.g. 'abc123')	System blocks update with validation error	Error shown, no update made	Pass
TC-MOB-007-001	Adding emergency contact with valid input.	1. Login to app 2. Go to Emergency Contacts section 3. Add valid name and phone 4. Tap Save	App running; user authenticated	Valid contact name and number	Emergency contact added and saved	Contact saved as expected	Pass
TC-MOB-007-002	Adding emergency contact with invalid phone.	1. Go to Emergency Contacts 2. Enter invalid phone (e.g., too short) 3. Try to save	User logged in; emergency contact form open	Invalid phone number format	Error shown, contact not saved	Validation worked correctly	Pass
TC-MOB-007-003	Editing emergency contact.	1. Go to Emergency Contacts 2. Tap Edit on an existing contact 3. Change phone number 4. Save changes	At least one emergency contact exists	Updated contact number	New number appears in list; confirmation message shown	Changes applied successfully	Pass
TC-MOB-007-004	Deleting emergency contact.	1. Navigate to Emergency Contacts 2. Select a contact 3. Tap Delete and confirm 4. Verify it is removed	At least one contact exists to delete	Target contact	Contact removed from list and DB	Contact no longer visible	Pass
TC-WEB-008-001	Successful login to admin panel.	1. Open browser 2. Go to web panel URL 3. Enter valid username/password 4. Click Login	Admin credentials available; web server running	Correct login info	Access granted; redirected to dashboard	Login successful	Pass
TC-WEB-008-002	Login fails with invalid username.	1. Enter invalid username and any password 2. Submit login form 3. Observe error message	Access to login page; invalid input used	Invalid username	Error: 'Invalid username or password'	Correct rejection with error	Pass

TC-WEB-008-003	Login fails with wrong password.	1. Enter correct username 2. Enter wrong password 3. Submit and observe result	Correct username; incorrect password	Wrong password	Access denied; error message shown	Blocked as expected	Pass
TC-WEB-008-004	Session timeout auto-logout.	1. Login to web panel 2. Stay inactive for 15 minutes 3. Try to interact with panel	Admin user logged in; session timer enabled	Session timeout policy	User is logged out and redirected to login screen	Auto-logout worked	Pass
TC-WEB-009-001	Display users' locations on map.	1. Login to admin panel 2. Go to Map View 3. Check active users' markers	Users have location data; system online	User location data	User positions marked correctly on map	Locations correctly displayed	Pass
TC-WEB-009-002	User safety status shown with colors.	1. On Map View, observe status indicators 2. Compare color coding with users' safety states	Users have different status: Safe, Not Safe, No Response	Safety status for test users	Colors (e.g. Green, red, yellow) match definitions	Colors rendered correctly	Pass
TC-WEB-009-003	Display user info on map marker click.	1. Login to admin panel 2. Go to Map View 3. Click on a user marker 4. Review popup data	At least one user with data available	User marker with location and status	User's name, status, and location info displayed in popup	Details shown accurately	Pass
TC-WEB-010-001	Display of earthquake risk map.	1. Login to web panel 2. Go to Risk Maps section 3. Select Earthquake map 4. Review color-coded zones	Earthquake sensor data simulated	Seismic risk inputs	Risk zones shown on map with intensity coloring	Earthquake risks mapped correctly	Pass
TC-WEB-010-002	Display of water level risk map.	1. Trigger critical water sensor data 2. View flood risk map 3. Check legend and zone display	Sensor data active; admin logged in	Water level above threshold	High-risk areas shown in darker colors	Flood zones displayed properly	Pass
TC-WEB-010-003	Track incident response teams.	1. Admin logs in 2. Open Map View 3. Enable Incident Team layer 4. Check team icons	At least one response team active	Team GPS/location input	Team locations marked with distinct icons	All teams visible on map	Pass
TC-WEB-	Send instructions to response team.	1. Click on response team marker 2. Select 'Send Instruction'	Admin panel live; team simulated	Instruction text (e.g. 'Assist User X')	Message logged and visible on recipient side	Instruction sent and recorded	Pass

010-004		3. Enter message 4. Confirm sending					
TC-COM-011-001	Transmission under strong network.	1. Enable strong network on mobile 2. Send safety response 3. Check for quick sync on web	Strong Wi-Fi or 5G; system running	'I'm Safe' response	Response appears within 2 seconds on panel	Real-time sync successful	Pass
TC-COM-011-002	Transmission under weak network.	1. Enable low-signal simulation 2. Submit 'I'm Not Safe' 3. Measure response delay	Weak signal conditions simulated	'I'm Not Safe' response	Data syncs with minor acceptable delay	Response received after 4s delay	Pass
TC-COM-011-003	Performance under high traffic.	1. Simulate 1000 users responding 2. Send simultaneous data 3. Monitor load and system stability	Load test tool set; services live	1000 concurrent inputs	System stable; no crashes; response time within limits	System handled load with slight delay	Pass
TC-LOC-012-001	Accurate GPS location detection.	1. Open mobile app in open area 2. Allow GPS access 3. Compare shown map position	GPS enabled; location permission granted	Device GPS position	Map location matches actual user location	Correct location shown	Pass
TC-LOC-012-002	Location via Wi-Fi/Cellular when GPS off.	1. Turn off GPS 2. Enable Wi-Fi or mobile data 3. Open app and check location	No GPS; alternative positioning on	Wi-Fi based location	Location shown with slightly reduced accuracy	Near-correct location shown	Pass
TC-LOC-012-003	User location display on web map.	1. Enable location sharing in mobile app 2. Login to admin panel 3. Check map for user location	Mobile user location sharing active	Location update from app	User marked accurately on web panel map	Map reflects correct location	Pass
TC-LOC-012-004	User disables location sharing.	1. User turns off location sharing in app 2. Admin checks user on web map 3. Observe map behavior	User logged in and manually disables GPS	Location sharing disabled	User disappears or shows 'Unknown'	Location stopped updating	Pass
TC-ALM-	Triggering earthquake alarm	1. Simulate earthquake data above threshold	Sensor simulator active	Earthquake above threshold	Alarm triggered; mobile alert delivered	Alert received in seconds	Pass

013- 001	and mobile notification.	2. System detects danger 3. Mobile receives alert					
TC- ALM- 013- 002	Correct content in alarm message.	1. Trigger simulated danger 2. Check alert content 3. Ensure danger type and location included	Alert system operational	Danger data from simulator	Message includes type (e.g. quake) and coordinates	Message correct and informative	Pass
TC- ALM- 013- 003	Alarm delivery via multiple channels.	1. Trigger flood alarm 2. Check mobile notification 3. Confirm SMS/voice alert if configured	Channels active (mobile/SMS)	Flood alert from sensor	Notification sent via all enabled channels	All alerts delivered	Pass
TC- ALM- 013- 004	Alarm prioritization and false alarm handling.	1. Trigger multiple low/high-level alarms 2. Check priority in display and logs 3. Trigger false alert and observe system behavior	System running with alert rules	Low and high level events	System gives priority to critical alerts; false alarms marked	Priorities respected; false alarm logged	Pass

4.2. Evaluation of Test Results and Error Analysis

During the testing process, all basic modules of the Post-Disaster Communication System were evaluated in detail, and both functional and performance-oriented tests were successfully completed. All 45 test scenarios were concluded as “Successful” (Pass), and the stability and accuracy of the system were largely proven. In particular, it was observed that critical features such as user security status notification, emergency notification and map integration worked stably.

In the tests conducted on the mobile application side, it was determined that processes such as user registration, login, profile update and emergency contact management were both user-friendly and worked correctly. It was observed that correct warnings were given against incorrect login data and that the system's verification mechanisms were reliable. In addition, it was verified through tests that mobile notifications were triggered correctly and delivered on time according to user reactions.

In the tests conducted on the web-based management panel, functions such as displaying user locations correctly on the map, color coding according to security situations and monitoring emergency teams worked flawlessly. It was demonstrated through tests that login transactions were managed securely and that mechanisms such as session timeout were successfully implemented.

In the load tests conducted in terms of system durability, it was observed that the system could operate without crashing or experiencing serious performance loss against the data load from thousands of simultaneous users. It was documented that user responses could be processed in the system even under weak network conditions and synchronization was achieved within maximum acceptable delays.

Although no critical errors were detected during the tests performed, it was observed that in some cases the response time of the system approached the threshold value, and minimal delays were experienced, especially in heavy traffic scenarios. This situation shows that performance optimization should be continued in order for the system to be ready for larger-scale use in the future. In addition, it is recommended that user-friendly interfaces be made more intuitive in stressful situations.

When evaluated in general, the testing process showed that the system complied with the requirements and offered high accuracy, reliability and accessibility in times of disaster. The success rate obtained shows that the software development process was well planned and the test plan was implemented effectively. The system can be further strengthened with performance increases, user experience improvements and the integration of new features in future versions.

4.3. Performance and Reliability Assessment

he performance and reliability features of the system were evaluated in detail in various test environments simulating real disaster scenarios. In particular, system behaviors against user density, low connection quality, different device types and unexpected outages were measured and analyzed. In the tests performed, both the response times of backend services and the response times of frontend components were taken into account.

In the performance tests conducted under critical scenarios, it was observed that the system could successfully process notifications and responses from 1000 simultaneous users. In the measurements made at the API level, the average response time remained in the range of 0.8–1.2 seconds, which provided a performance that could be accepted within the industry standards. No error situations such as system crashes, unresponsiveness or disconnection were experienced during the test period.

The durability of the mobile application under low network quality was also tested. In the tests conducted at low bandwidths such as 2G, maximum 4–5 second delays were observed in the transmission of user responses such as 'I am safe' and 'I am not safe' to the system. This shows that the system maintains its functionality even under minimum connection conditions.

In the reliability tests conducted, it was observed that the system did not produce memory leaks, performance losses or log errors in long-term use (8 hours of continuous operation). In addition, it was documented that the connection between the mobile application and the server was automatically re-established after momentary interruptions and that no data loss occurred.

In addition, the caching strategies in the system were also tested and it was confirmed that user locations on the panel could be updated instantly on the map. Despite the continuous logging of the data flow services running in the background, no significant slowdown in system performance was observed.

In terms of reliability, it was observed that the security responses sent by the users were recorded correctly and displayed on the panel without errors. The fact that the timestamps in the responses were also fully compatible with the system clock revealed that the system synchronization was working properly. In addition, it was documented with the test results that the notifications directed to emergency persons and units in the event of any failure reached a 100% success rate.

As a result, the system demonstrated high success in terms of both performance and reliability. Thanks to its capabilities such as being able to respond quickly at critical moments, not experiencing data loss, and being able to work stably with a user-friendly interface, a robust communication infrastructure that can be used in times of disaster has been established. In future studies, expansion of cache management and parallel processing capabilities can be evaluated to further increase performance.

4.4. Improvement Recommendations

Although the system has generally demonstrated high performance and reliability, various improvements are suggested for wider scale usage and preparation for disaster recovery. Particularly, the near-limit response times observed in scenarios with high user density reveal the need for performance optimization at the infrastructure level. In this context, parallel processing and caching strategies can be used more effectively in backend services.

On the mobile application side, it is suggested that user interfaces be restructured to be simpler and more guiding in accordance with stressful situations. Making interactions more intuitive for users to quickly report security situations will increase the effectiveness of the system. In addition, it is important to activate offline operating modes and concurrency algorithms to increase usability in environments with low connection quality.

Despite the strong map integration observed in the management panel, the addition of advanced visualization techniques such as clustering of user locations and density maps will strengthen decision support mechanisms. At the same time, adding filtering and instant search capabilities to the user monitoring functions in the panel will speed up the intervention process for emergency managers.

Finally, in order to ensure the sustainability of the system and facilitate its integration with next-generation disaster management systems, API-based integration opportunities with institutional structures should be developed.

5. Conclusion and Future Work

5.1. General Assessment and Contributions of the Project

This project presents a comprehensive software platform to improve post-disaster communication and coordination. It successfully integrates sensor-based event detection, mobile-based user interaction, and a web-based dashboard for emergency personnel.

Key contributions include:

- Real-time environmental monitoring through IoT sensor integration.
- Automated alert generation and delivery through scalable notification systems.
- Simple and intuitive mobile interaction for users to report their safety status.
- Centralized data visualization and dispatch coordination for emergency teams.

The architecture follows software engineering principles such as low coupling, high cohesion, abstraction, modularity, and reusable components. The use of standardized interface documentation, data validation strategies, and structured error handling ensures the maintainability and extensibility of the system in future deployments.

5.2. Future Improvements and New Project Ideas

While the current system lays a strong foundation, there are several promising directions for future development and enhancement:

- **Offline Communication Support**: A critical next step is to allow the system to function without internet access. This includes implementing offline message storage, Bluetooth-based peer-to-peer communication, and mesh networking for local broadcasting in affected zones.
- **Artificial Intelligence for Risk Prediction**: By leveraging historical sensor data and user responses, machine learning models can be trained to predict potential disasters or identify anomalies earlier, allowing proactive responses.

- **Integration with Government and Institutional Systems**: Establishing API-based integration with organizations such as disaster management agencies, hospitals, or law enforcement can create a fully automated alert and resource dispatch network.
- **Wearable Device Compatibility**: Integrating the system with wearable devices (e.g., smartwatches) can offer more reliable data collection on user activity, vitals, and safety status.
- **Advanced Mapping and GIS Integration**: Incorporating geospatial analytics can help visualize risk zones, user clusters, and response units in more detail, aiding decision-making during crises.
- **Public Awareness Campaign Tools**: A simplified version of the system could be used for educational simulations or drills, helping communities prepare for potential disasters.

With these extensions, the system can evolve into a robust, intelligent, and resilient solution capable of saving lives and minimizing disruption during natural disasters.

6. Glossary

Term	Definition
Scalability	The ability of a system to handle increased workload or number of users without compromising performance.
Fault Tolerance	The capability of a system to continue operating properly even in the event of partial hardware or software failures.
Latency	The delay between a user's action and the system's response, often measured in milliseconds or seconds.
Load Testing	A type of performance testing used to determine how a system behaves under a specific expected load.
Session Timeout	A security and performance feature that ends inactive user sessions after a predetermined period.
Caching	A technique used to store frequently accessed data temporarily to reduce access time and server load.
API Response Time	The duration it takes for a backend API to process a request and return a response.
Resilience	The system’s ability to recover quickly from disruptions, ensuring continued functionality during and after failures.
Real-Time Notification	Instant alerts or messages sent by the system in response to user actions or sensor input, without delay.
Data Synchronization	The process of ensuring consistency and accuracy of data between different system components or devices.
Push Notification	A message sent from a server to a client application (such as a mobile app), even when the app is not actively in use.
Modularity	A design principle that structures a system as a set of distinct, interchangeable components, making it easier to develop, test, and maintain.

7. References

[1] Alesheikh, A. A., Helali, H., & Behroz, H. A. (2002, July). Web GIS: technologies and its applications. In Symposium on geospatial theory, processing and applications (Vol. 15, pp. 1-9). Hannover, Germany: ISPRS.

[2] Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. Computer networks, 54(15), 2787-2805.

[3] Bandyopadhyay, D., & Sen, J. (2011). Internet of things: Applications and challenges in technology and standardization. Wireless personal communications, 58, 49-69.

[4] Chen, R., Sharman, R., Rao, H. R., & Upadhyaya, S. J. (2008). Coordination in emergency response management. Communications of the ACM, 51(5), 66-73.

[5] Zeng, L. (2009). Designing the User Interface: Strategies for Effective Human-Computer Interaction by B. Shneiderman and C. Plaisant: Pearson Addison-Wesley, 2009. xviii+ 606 pages. ISBN: 978-0-321-53735-5.

[6] Babovic, Z. B., Protic, J., & Milutinovic, V. (2016). Web performance evaluation for internet of things applications. IEEE Access, 4, 6974-6992.

[7] Meissner, A., Luckenbach, T., Risse, T., Kirste, T., & Kirchner, H. (2002, June). Design challenges for an integrated disaster management communication and information system. In The First IEEE Workshop on Disaster Recovery Networks (DIREN 2002) (Vol. 24, pp. 1-7). New York: IEEE.

[8] Nielsen, J. (1994). Usability engineering. Morgan Kaufmann.

[9] Bertolino, A. (2007, May). Software testing research: Achievements, challenges, dreams. In Future of Software Engineering (FOSE'07) (pp. 85-103). IEEE.

[10] Beizer, B. (1995). Black-box testing: techniques for functional testing of software and systems. John Wiley & Sons, Inc..

[11] Rafael, S., Santiago, E., Rebelo, F., Noriega, P., & Vilar, E. (2022, June). Bio-Centred interaction design: a new paradigm for human-system interaction. In International Conference on Human-Computer Interaction (pp. 69-79). Cham: Springer International Publishing.

[12] Hossain, M. S., & Muhammad, G. (2016). Cloud-assisted industrial internet of things (iiot)–enabled framework for health monitoring. Computer Networks, 101, 192-202.