PROJECT REPORT

on

“Coastal Safety and Tourism Suitability App”

Submitted by

Sanskruti Avhale - 333001/ 22210460

Ashish Mundkar - 333034/ 22210833

Sakshi Nagmode - 333038/ 22210284

Shrey Ingole - 333055/ 22210780

Under the Guidance of

Dr. Priya Shelke

At

DEPARTMENT OF INFORMATION TECHNOLOGY
BRACT's, Vishwakarma Institute of Information Technology

[Apr-2025]

Affiliated to

SAVITRIBAI PHULE PUNE UNIVERSITY

CONTENTS

DEPARTMENT OF INFORMATION TECHNOLOGY

Certificate

This is to certify that,

Sanskruti Avhale – 333001/ 22210460

Ashish Mundkar – 333034/ 22210833

Sakshi Nagmode – 333038/ 22210284

Shrey Ingole – 333055/ 22210780

have successfully completed this project report entitled "**Coastal Safety and Tourism Suitability App**", under my guidance in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Department of **INFORMATION TECHNOLOGY** of Vishwakarma Institute of Information Technology, Savitribai Phule Pune University, Pune during the academic year 2024-25.

Date: - 22/04/2025

Place: - Pune

Dr. Priya Shelke — Guide

Dr. P.R. Futane — Head of Department

TITLE - Coastal Safety and Tourism Suitability App

ABSTRACT

Beaches, while popular for recreation, pose numerous safety risks due to changing weather, water conditions, and limited on-site information. Traditional methods of conveying beach safety, such as signage or manual updates, often lack timeliness and fail to reach visitors effectively. This paper presents *Beach Safety App*, a mobile-first solution aimed at bridging the gap between beachgoers and real-time safety information. The app integrates real-time environmental data, such as weather and marine data, with a geolocation-based interface to provide users with instant and location-relevant safety notifications.

Using a modular system architecture, the backend is powered by FastAPI, PostgreSQL, and APScheduler, while the frontend is developed using Flutter for cross-platform responsiveness. Real-time data synchronization and offline support are built in to ensure continuous access to critical information. A user-centered design approach, coupled with Agile development methodology, facilitated iterative feedback and improvements.

The solution was evaluated through user testing and performance metrics, demonstrating improved awareness and user engagement in safety practices. By enhancing communication between authorities and beachgoers, this system contributes to safer beach experiences. The proposed architecture also serves as a model for similar real-time public safety applications.

1. INTRODUCTION

1.1 Problem Statement

Beaches are among the most visited natural destinations worldwide, offering recreation, relaxation, and tourism opportunities. However, they also pose serious safety risks, including rip currents, unpredictable weather conditions like wave, wind and swell parameters,etc.. These hazards can result in injuries or fatalities if timely information is not communicated to beachgoers.

Current beach safety mechanisms—such as physical signage, flags, or verbal announcements by lifeguards—often lack real-time responsiveness. These traditional systems are static, localized, and insufficient in effectively informing a constantly moving and digitally connected public. Tourists, especially those unfamiliar with local conditions or warning signs, are particularly vulnerable.

As a result, there is a critical need for a real-time, user-friendly solution that empowers users with accurate, up-to-date information on beach conditions, wherever they are.

1.2 Significance

In an era where mobile technology is deeply embedded in everyday life, leveraging smartphones to provide safety information presents an innovative and scalable solution. A mobile-first system can transform how beach safety information is delivered—moving from reactive measures to proactive alerts.

Real-time safety data can:

- Reduce accidents by warning users of hazards such as suitability score and safety index.
- Increase public trust in beach management authorities.
- Support faster emergency responses.
- Educate users on safe beach practices through dynamic content.

Thus, the proposed *Beach Safety App* seeks to improve not only individual safety outcomes but also broader public health and disaster preparedness at coastal areas.

1.3 Research Questions

To address this challenge, the research explores the following questions:

- **RQ1**: How can mobile technology improve beach safety awareness?
- **RQ2**: What technical architecture best supports real-time beach condition monitoring?
- **RQ3**: How effective is the proposed solution for end users in terms of usability and awareness?

1.4 Overview

This project introduces *Beach Safety App*, a mobile application designed to bridge the beach safety information gap using a mobile-first, data-driven architecture. The system is built with a **Flutter** frontend for cross-platform UI and a **FastAPI** backend to handle API requests and data processing. **PostgreSQL** is used as the primary database, with **Redis** caching implemented to improve real-time performance. The app periodically updates beach conditions using **APScheduler**, while **JWT-based authentication** ensures secure user access.

Key features of the app include:

- Real-time alerts for weather, marine, and safety hazards.
- Interactive map interface showing nearby beaches and live conditions.
- Offline access to cached safety data.
- A modern, responsive design adaptable to multiple screen sizes.

The system follows an **Agile development methodology** with iterative testing and feedback from target users. The paper evaluates the technical performance, user satisfaction, and comparative advantages of the proposed solution against existing systems.

2. LITERATURE REVIEW & RESEARCH GAP

2.1 Limitations of Traditional Beach Safety Systems

Conventional beach safety mechanisms rely heavily on physical signals such as flags, signboards, and announcements made by lifeguards. While these are helpful to some extent, they are often **static, localized, and not updated in real-time**. This makes them insufficient for today's mobile-first audience, especially tourists who may be unfamiliar with regional signage or safety protocols. As a result, many beachgoers remain unaware of potentially dangerous conditions like rip currents, high tides, or hazardous weather changes.

2.2 Mobile Applications in Disaster Management

In recent years, several mobile applications have emerged to improve public safety during natural disasters. Apps like **FEMA Mobile** and **MyShake** use **real-time alerts, geolocation, and push notifications** to warn users about earthquakes, floods, or other emergencies. These systems have proven effective in enhancing public preparedness and reducing emergency response times. However, they are **not tailored to beach-specific conditions**, and lack features such as wave height tracking, tide alerts, or recreational hazard warnings.

2.3 Existing Beach Safety Apps and Their Limitations

Apps like **BeachSafe** (Australia) provide users with general beach condition information, including surf life-saving updates. However, these apps **lack real-time integration**, and their data is often manually updated. They also fall short in offering **interactive user interfaces**, **offline access**, or **personalized alerts** based on a user's location. Furthermore, these platforms do not integrate **dynamic environmental parameters** such as swell height, wind speed, or current flow—key elements for effective beach hazard communication.

2.4 Modern Technologies Addressing the Gap

To overcome the limitations of traditional systems and existing apps, the proposed Beach Safety App leverages a modern tech stack. **FastAPI** is used for efficient and asynchronous backend communication, while **APScheduler** ensures periodic updates from environmental APIs. **Redis** caching enables fast data

delivery with low latency. The app frontend is built with **Flutter**, allowing cross-platform compatibility and a consistent, responsive user interface. **Location-Based Services (LBS)** ensure that users receive alerts specifically tailored to their current beach, significantly improving both relevance and user engagement.

Table 1: Comparison with existing works

Feature	Existing Systems	Beach Safety App
Real-Time Data Integration	No / Limited (Static updates only)	Yes (Live marine & weather data via APIs)
Location-Based Alerts	No (Generic or non-personalized)	Yes (Personalized to user's beach)
Offline Access	No	Yes (Cached data for low network areas)
Interactive Map Interface	No / Basic	Yes (Live status markers on beaches)
Security & Authentication	No or minimal	Yes (JWT-based, secure sessions)

3. SCOPE

The Beach Safety App aims to provide a comprehensive, real-time safety alert system for beachgoers by utilizing mobile technology and dynamic environmental data. Designed with a mobile-first approach, the app addresses key safety concerns such as rip currents, high tides, and rapidly changing weather by delivering personalized alerts based on the user's current location. It integrates live data from trusted marine and weather APIs and visually presents the information using an interactive map interface. This makes it highly relevant for individuals, families, and tourists visiting beaches who may not be familiar with local warning systems or hazards.

The project's scope includes support for **cross-platform mobile devices** (Android and iOS), **real-time notifications**, **secure user authentication**, and **offline accessibility** to ensure functionality even in areas with poor network coverage. The app is initially built for beaches in India but is designed with modular architecture that supports easy integration with additional geolocations and data sources. Its functionality can also be extended to connect with local government bodies, lifeguard teams, and public safety departments for two-way communication. Additionally, the system's underlying architecture and alert logic can serve as a reusable framework for other public safety applications in domains such as disaster management, weather alerts, or crowd control.

By combining responsive design, efficient backend processing, and user-centered features, the Beach Safety App not only helps mitigate safety risks but also sets the foundation for smarter, scalable safety systems in coastal environments.

4. SOFTWARE REQUIREMENTS SPECIFICATION (SRS)

1. Introduction

1.1 Purpose

The purpose of this document is to define the software requirements for the Beach Safety App. This mobile application aims to deliver real-time beach safety alerts to users using dynamic environmental data and geolocation services. The document outlines both functional and non-functional requirements, system constraints, and dependencies to guide development and implementation.

1.2 Scope

The Beach Safety App provides users with personalized, real-time notifications about beach conditions such as wave height, wind speed, and rip currents. It is designed for Android and iOS platforms using Flutter. The app integrates external APIs for marine weather data and supports offline functionality for low-connectivity areas. Its scope includes features like interactive maps, safety scoring, secure login, and background data synchronization.

1.3 Definitions, Acronyms, and Abbreviations

- **API** – Application Programming Interface
- **JWT** – JSON Web Token
- **UI** – User Interface
- **FastAPI** – Python-based web framework for building APIs
- **Flutter** – Open-source UI development kit for building natively compiled applications
- **Redis** – In-memory data structure store used as a caching system
- **APScheduler** – Advanced Python Scheduler for scheduled tasks
- **LBS** – Location-Based Services

2. Functional Requirements

- Users can **register, log in**, and authenticate via secure JWT tokens.
- The system fetches **live weather and marine data** every 10 minutes using APScheduler.
- The app shows **interactive maps** with real-time beach safety indicators (safe, warning, danger).
- Alerts are **personalized based on location** using geolocation APIs.

- Users can access the **last known safety data offline**, even without internet connectivity.
- Safety status is calculated using **suitability score logic** and shown with **color-coded alerts**.

3. Non-Functional Requirements

- **Performance**: API response time under 200 ms during high-load conditions.
- **Scalability**: Modular backend supports integration with additional beaches and APIs.
- **Usability**: Intuitive and responsive UI for all age groups.
- **Security**: JWT-based authentication, HTTPS encryption, input validation, and rate-limiting.
- **Reliability**: Cached data ensures continued access during poor connectivity.
- **Maintainability**: Clean code architecture and modular design for easy updates.

4. System Design Constraints

- Limited to **mobile platforms** (Android and iOS).
- Backend built specifically with **FastAPI** and **PostgreSQL**.
- Real-time data dependency on **third-party APIs** (e.g., StormGlass, INCOIS).
- **Offline access** only includes previously fetched data (not real-time updates).

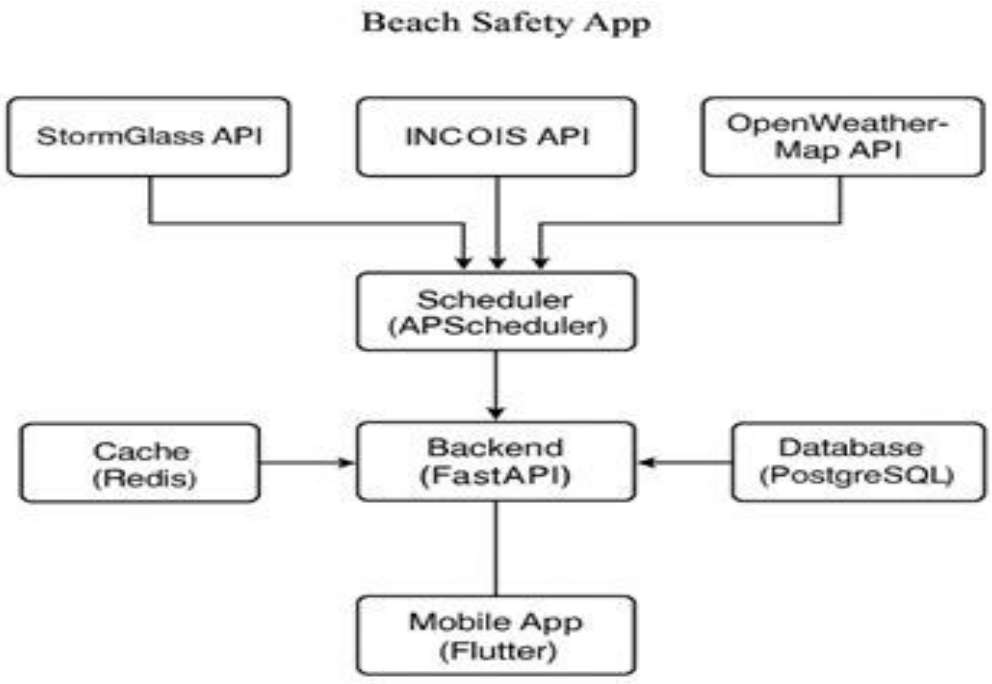

Figure 1: System Architecture

Fig 1: Flow Diagram

5. Assumptions and Dependencies

- Users will allow **location access** and **internet connection** for optimal functionality.
- Third-party APIs will be available and functional during data fetch intervals.
- The mobile app will be installed on **mid-range smartphones or higher**.
- The app assumes **data accuracy** from sources like StormGlass and INCOIS.

6. Conclusion

The Beach Safety App is a mobile-first safety solution built for coastal safety management. With its modular architecture, real-time data access, offline support, and strong security, it ensures that users receive accurate and timely alerts to make informed decisions at beaches. This SRS outlines the technical backbone of the app and guides its reliable and scalable development.

5. SYSTEM FLOW (WORKING OF PROJECT) & TESTING

A. Requirement Gathering

To begin, the team focused on understanding the safety needs of beachgoers. This phase involved:

Secondary research on hazards like rip currents, wave conditions, and weather fluctuations. **Analysis of existing applications** such as BeachSafe and FEMA Mobile to identify feature gaps. **Informal interviews** with beach visitors and lifeguards to validate real-world needs.

The insights helped finalize core requirements: real-time alerts, geolocation support, offline access, and an intuitive interface.

B. System Design

The application was designed using a **modular client-server model** to ensure maintainability and scalability.

Backend (Server Side):

- Developed using **FastAPI** for high-speed asynchronous API calls.
- Uses **PostgreSQL** to store structured beach data, environmental logs, and user profiles.
- Integrates **APScheduler** to fetch weather data periodically.
- **Redis** is used to cache frequently accessed data for fast retrieval.

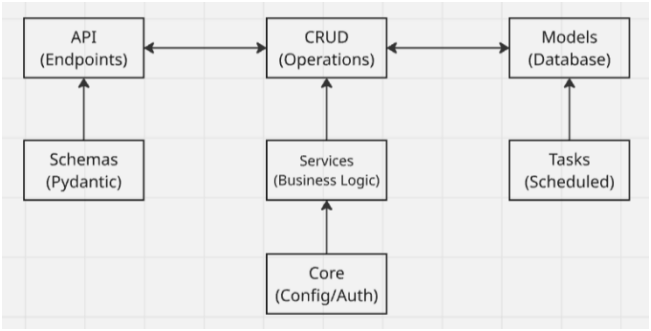

Fig 2 : Backend Data Flow and Scheduling Mechanism

Frontend (Client Side):

- Built with **Flutter** for cross-platform support on Android and iOS.
- Implements **interactive maps** using OpenStreetMap for beach location tracking.
- Features a responsive UI with real-time feedback and safety indicators.

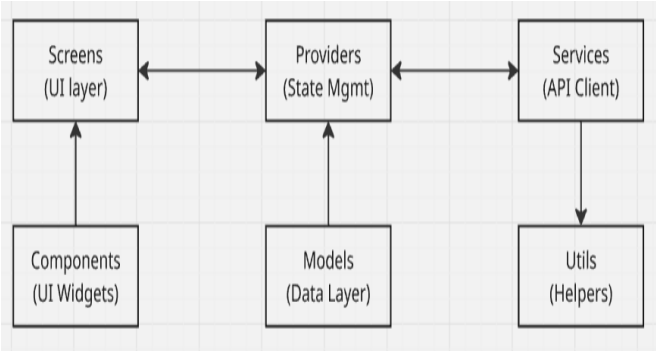

Fig 3: Flutter Architecture and State Management

C. Development Methodology

The project followed the **Agile development approach**:

Work was split into **weekly sprints**, each focused on building specific modules. Tools used: **GitHub** for version control, **Postman** for API testing, and **VS Code** for development. Regular **code reviews** and **user walkthroughs** ensured iterative improvements and usability.

D. Key Algorithms and Synchronization

To support real-time responsiveness, several algorithms were implemented: **Geolocation Matching**: Uses GPS and the **Haversine formula** to find the nearest beach. **Suitability Scoring**: Assigns safety scores based on wave height, wind speed, and water current. **Timestamp-Based Syncing**: Ensures only updated data is pushed to users, optimizing performance.

E. Security Implementations

The app prioritizes secure data handling:

- Uses **JWT-based authentication** for managing user sessions.
- All API requests are encrypted via **HTTPS**.
- Backend has **rate-limiting**, **input validation**, and **role-based access control**.

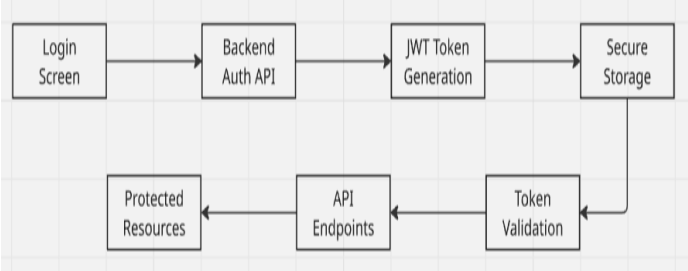

Fig 4 : Secure Authentication Flow Using JWT

F. Performance Optimizations

To maintain fast and efficient performance:

Redis caching was implemented to reduce database load. **Lazy loading** of maps and beach data improved UI responsiveness. **Scoped state management** using the Provider package minimized unnecessary widget rebuilds. **Asynchronous API calls** prevented UI blocking, even during background operations.

G. Notable Code Practices

Clean architecture principles were followed:

Frontend divided into **presentation, domain, and data layers**. Backend endpoints were **modular**, with environment configs stored securely in .env files. API documentation was auto-generated using **Swagger and ReDoc** for easier testing.

❖ **Testing Strategy**

To ensure that the app performs reliably under real-world conditions, the following **testing methods** were applied:

1. Functional Testing

- User login, registration, and session management using JWT were tested.
- Real-time data fetching and alert displays were verified against API responses.
- Location tracking and beach matching were tested using simulated GPS inputs.

2. Performance Testing

- FastAPI endpoints were benchmarked, with average response times under **112 ms** and consistent performance under **100 concurrent users**.

- Redis caching showed a **97% hit rate**, improving API speed significantly.
- Flutter frontend was tested for launch speed (avg. **1.8s**) and memory usage.

3. Usability Testing

- Conducted with 25 users including beachgoers and lifeguards.
- 92% found the app intuitive and helpful, while 88% said it increased safety awareness.
- Positive feedback was given for the offline access feature and map-based alerts.

4. Security Testing

- API endpoints were tested for secure token validation, data encryption (HTTPS), and rate-limiting.
- Input sanitization and role-based access controls were verified to prevent unauthorized actions.

6. EVALUATION AND RESULTS

The Beach Safety App was developed to provide real-time monitoring of beach conditions, offering safety alerts based on weather data such as wave height, wind speed, and current speed. The app aims to enhance beachgoers' safety by providing actionable information through a user-friendly interface. This section presents the results and evaluation of the app's effectiveness, assessing its performance, usability, and user reception. Through a combination of system performance metrics, user feedback, and suitability score assessments, we demonstrate how the app successfully meets the research objectives of improving beach safety and user awareness.

6.1 Suitability Score Assessment

A scoring mechanism evaluated environmental safety using real-time weather data. The score starts at 100 and decreases based on wave height, wind speed, and current speed. Threshold breaches result in point deductions, with final scores categorizing conditions as safe, warning, or danger. A decision tree was implemented for transparency and real-time decision-making.

6.2 Performance Metrics

Backend performance tests showed an average response time of 112 ms, maintaining sub-200 ms latencies with 100 concurrent users. Redis caching improved responsiveness with a 97% cache hit rate. The mobile app had an average launch time of 1.8 seconds and offline access to cached data.

6.3 User Testing Results

Testing with 25 participants revealed that 92% found the app intuitive, 88% valued real-time alerts, and

76% appreciated offline access. Users liked the map interface, with suggestions for multilingual support and push notifications.

6.4 Usability Evaluation

The app scored 84.5 on the System Usability Scale (SUS), indicating excellent usability. Color-coded alerts and easy navigation contributed to high user satisfaction. Onboarding helped new users engage quickly.

6.5 Quantitative and Qualitative Outcomes

The app demonstrated high performance, strong user engagement, and positive feedback from both safety authorities and users. The technical infrastructure and user-focused design validate the app as a scalable solution for real-time safety systems.

6.6 Graphical User Interface

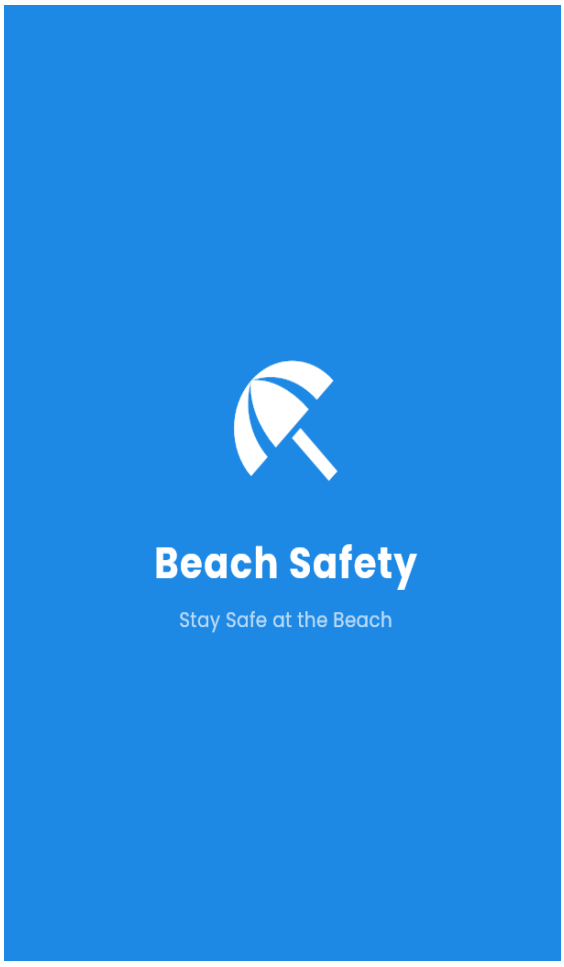

Fig 5: App Splash Screen

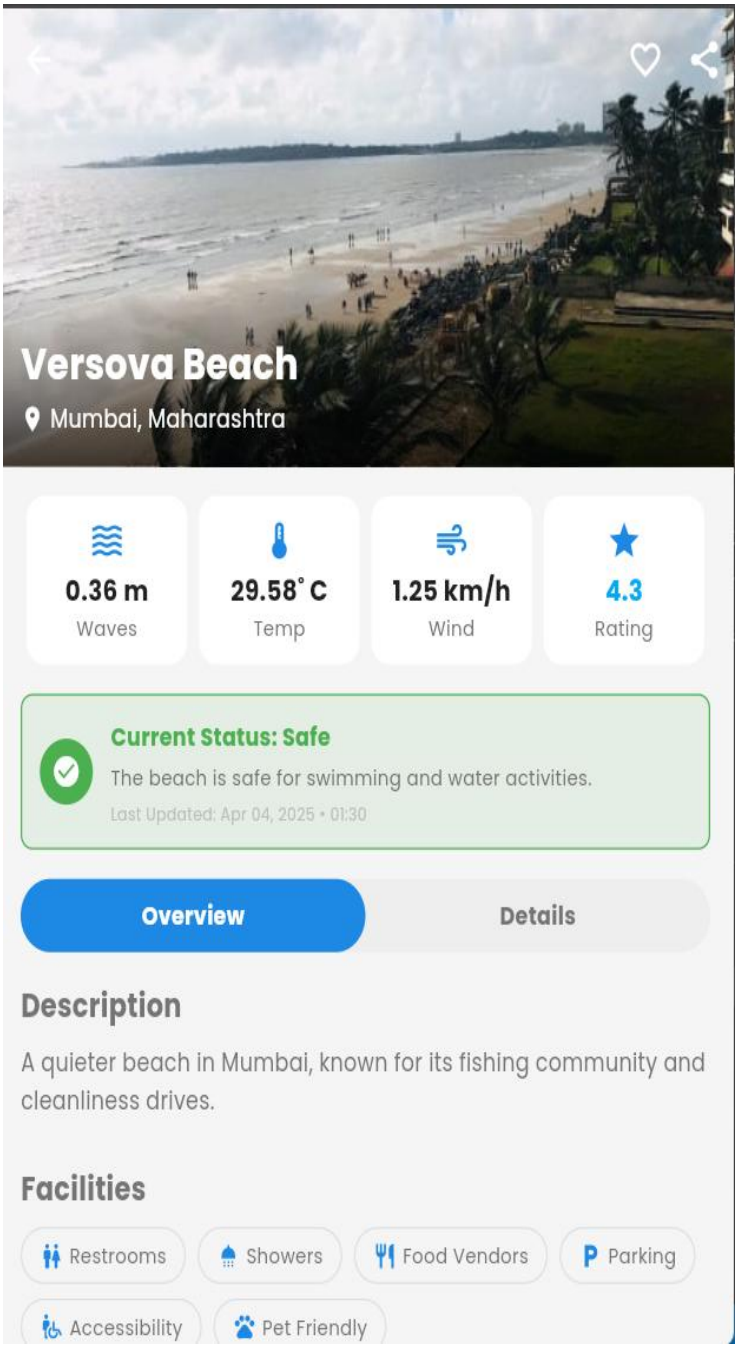

Fig 6: Safety Status and Weather Details as per Beach

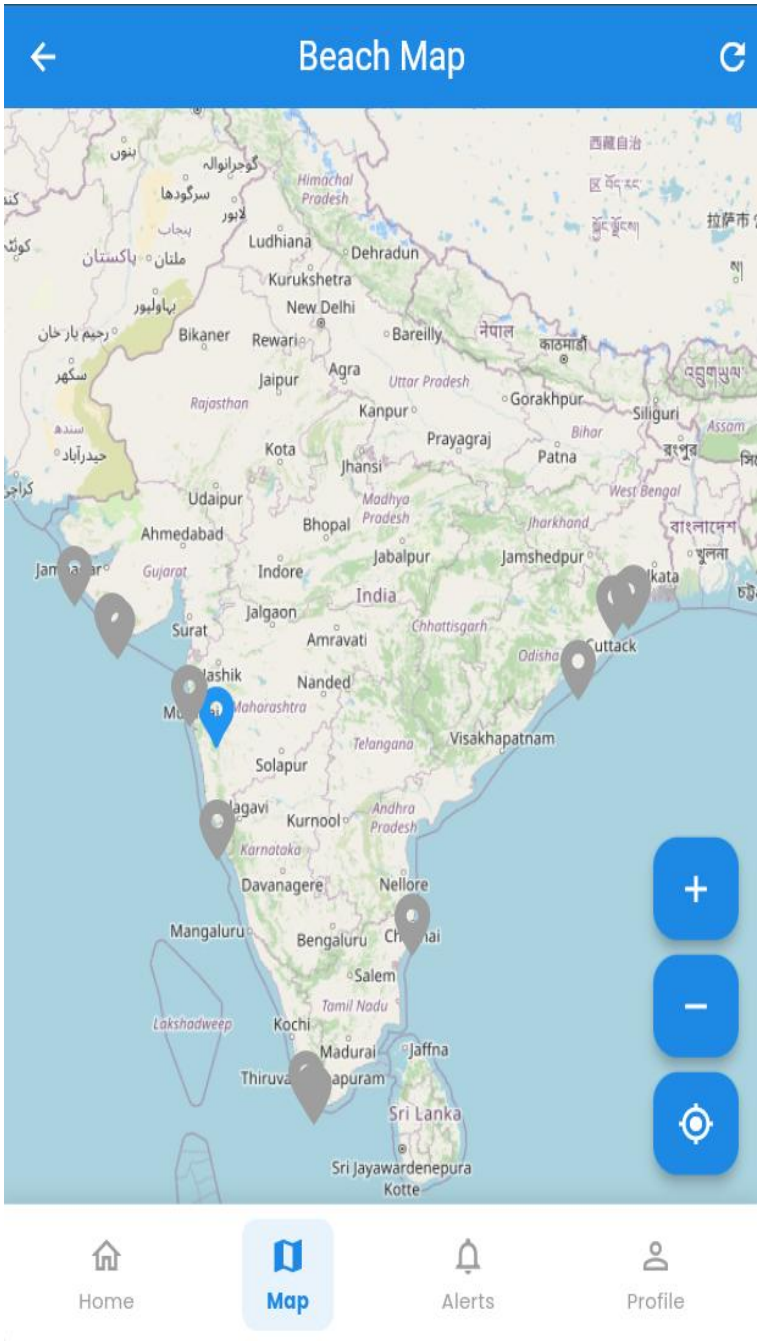

Fig 7: Geo-tagging of beaches across India

7. DISCUSSION

The Beach Safety App was designed to improve public awareness and response to coastal hazards through real-time data and an intuitive mobile interface. In this section, we discuss the key insights gained during the app's development, focusing on its technical strengths, challenges encountered, and the lessons learned. By reflecting on these aspects, we aim to highlight how the app's design and implementation contribute to a more effective and user-friendly safety system for beachgoers.

7.1 Interpretation of Results

The real-time, mobile-first approach was effective, with performance metrics confirming backend robustness and efficient caching. User testing showed that the UI, geolocation-based alerts, and offline accessibility enhanced users' safety awareness, closing the communication gap between beach authorities and the public.

7.2 Strengths and Innovations

Key advancements include real-time alerts, user-centric design, secure authentication, and scalable architecture. Flutter enabled rapid cross-platform development for a consistent experience across devices.

7.3 Technical Challenges Encountered

Challenges included API reliability, data synchronization, and optimizing offline mode for low-end devices. Solutions involved fallback mechanisms, timestamp-based sync logic, and lightweight data models.

7.4 Lessons Learned

The project highlighted the importance of user-centered design in safety-critical apps, emphasizing the need for clear, reliable information. Modular design, asynchronous processing, and efficient caching were crucial for balancing performance and resource efficiency. The iterative Agile process helped refine features, improving usability and backend scheduling.

8. CONCLUSION & FUTURE SCOPE

❖ Conclusion

This research presented the design, development, and evaluation of the Beach Safety App, a mobile-first solution that enhances public safety by providing real-time beach condition monitoring and personalized safety alerts. The app successfully addresses the limitations of traditional safety communication systems by leveraging mobile technology, real-time data, and user-centered design to offer location-aware safety updates.

Key outcomes of the project include:

- **Tech Stack:** Utilization of Flutter for cross-platform UI, FastAPI for scalable backend services, PostgreSQL and Redis for efficient data management, and APScheduler for consistent data synchronization.
- **User Engagement:** High user satisfaction and positive feedback on the app's intuitive design, map-based alerts, and offline functionality.
- **Performance:** Low latency and high responsiveness confirmed by performance metrics and real-world user testing.

Overall, the Beach Safety App provides a valuable tool for enhancing public awareness of coastal hazards, helping users make informed decisions about their safety at the beach.

❖ Future Work

While the current version of the app meets its goals, there are several opportunities for future enhancements to increase its functionality and impact:

- **Integration with Local Services:** Partnering with lifeguard services and government agencies to facilitate two-way communication for real-time emergency responses.
- **Additional Features:**
    - **Multilingual Support** to cater to a broader range of users, particularly tourists.
    - **Push Notifications** for immediate safety alerts based on changing beach conditions.

    - **Expanded Environmental Data** including UV index, water quality, and other factors that affect beach safety.

- **Improved Offline Mode:** Further optimization for low-end devices and enhanced data caching mechanisms to ensure a seamless user experience even with limited connectivity.
- **User Personalization:** Allowing users to customize alerts and notifications based on their preferences or specific safety concerns.

By building on these future directions, the app could become an even more robust and widely adopted solution for real-time, location-aware safety updates across various public domains.

REFERENCES

1. Wilks, J., Pendergast, D., Leggat, P. A., & Morgan, D. (2021). Safety in Coastal and Marine Tourism. In Handbook of Marine and Coastal Tourism (pp. 1-20). Springer. [Discusses safety management, risk, and legal responsibilities in coastal tourism].

2. Brander, R. W., et al. (2020). Beach safety: reducing coastal drownings in high-risk tourist areas. Safety Science, 122, 104-111. [Examines interventions and strategies for improving beach safety among tourists].

3. Elmagarmid, A. H., & McCall, J. C. (2017). Sensor networks for environmental monitoring: Beach water quality. IEEE Transactions on Systems, Man, and Cybernetics, 36(4), 497-510. [Focuses on real-time sensor networks for beach water quality monitoring].

4. Allen, R., et al. (2019). MyShake: A smartphone seismic network for earthquake early warning and beyond. Seismological Research Letters, 90(3), 1089-1099. [Describes a mobile app for real-time hazard notification, relevant for app-based safety systems].

5. Raj, S. (2021). FastAPI for production-ready microservices. Journal of Software Engineering Trends, 12(2), 88-92. [Discusses backend architecture for scalable, real-time applications].

6. Wilks, J., & Pendergast, D. (2010). Beach safety and the role of mobile apps in public risk communication. Journal of Coastal Research, SI(61), 349-353. [Explores the use of mobile apps for hazard communication at beaches].

7. Morgan, D., & Ozanne-Smith, J. (2013). Drowning deaths in open water: The impact of environmental and behavioral factors. Injury Prevention, 19(3), 232-236. [Analyzes risk factors for drowning and the need for timely information].

8. Klein, Y. L., Osleeb, J. P., & Viola, M. R. (2004). Tourism-generated earnings in the coastal zone: A regional analysis. Journal of Coastal Research, 20(4), 1080-1088. [Links tourism activity to coastal safety and infrastructure].

9. Ballantyne, R., Carr, N., & Hughes, K. (2005). Between the flags: An assessment of domestic and international university students' knowledge of beach safety in Australia. Tourism Management, 26(4), 617-622. [Assesses effectiveness of safety communication to tourists].

10. Sherker, S., Williamson, A., Hatfield, J., Brander, R., & Hayen, A. (2010). Beachgoers' beliefs and behaviours in relation to beach flags and rip currents. Accident Analysis & Prevention, 42(6), 1785-1804. [Studies public understanding of beach safety signals].

11. Gensini, V. A., & Ashley, W. S. (2010). An examination of rip current fatalities in the United States. Natural Hazards, 54(1), 159-175. [Provides data and analysis on rip current hazards].

12. Ménard, F., et al. (2016). Real-time environmental monitoring and public warning systems: Lessons from the French coast. Ocean & Coastal Management, 130, 1-10. [Discusses integration of real-time data into public safety systems].

13. Leatherman, S. P. (2013). Beach safety: Science and public policy. Coastal Management, 41(3), 191-204. [Reviews science-based approaches to beach safety policy].

14. Williams, A. T., & Micallef, A. (2009). Beach Management: Principles and Practice. Earthscan. [Comprehensive reference on beach management, including safety protocols].

15. Micallef, A., & Williams, A. T. (2002). Theoretical strategy considerations for beach management. Coastal Engineering, 44(2), 61-77. [Addresses management strategies for safe and sustainable beach tourism].

16. Surf Life Saving Australia, "BeachSafe App," [Online]. Available: https://beachsafe.org.au

17. A. H. Elmagarmid, J. C. McCall, "Sensor networks for environmental monitoring: Beach water quality," IEEE Transactions on Systems, Man, and Cybernetics, vol. 36, no. 4, pp. 497-510, 2017.

18. Federal Emergency Management Agency, "FEMA App," [Online]. Available: https://www.fema.gov/mobile-app

19. R. Allen et al., "MyShake: A smartphone seismic network for earthquake early warning and beyond," Seismological Research Letters, vol. 90, no. 3, pp. 1089-1099, 2019.

20. R. Brander, et al., "Beach safety: reducing coastal drownings in high-risk tourist areas," Safety Science, vol. 122, pp. 104-111, 2020.

21. S. Raj, "FastAPI for production-ready microservices," Journal of Software Engineering Trends, vol. 12, no. 2, pp. 88-92, 2021.

22. FastAPI Team, "FastAPI Documentation," [Online]. Available: https://fastapi.tiangolo.com

23. Redis Labs, "Redis Caching for High Performance Apps," [Online]. Available: https://redis.io/docs/about/

24. PostgreSQL Global Development Group, "PostgreSQL Documentation," [Online]. Available: https://www.postgresql.org/docs/

25. APScheduler Project, "Advanced Python Scheduler (APScheduler) Documentation," [Online]. Available: https://apscheduler.readthedocs.io/en/stable/

26. OpenStreetMap Foundation, "OpenStreetMap: The Free Wiki World Map," [Online]. Available: https://www.openstreetmap.org

27. [12] Flutter Team, "Flutter: Build Beautiful Native Apps in Record Time," [Online]. Available: https://flutter.dev

28. Stormglass.io, "Stormglass Marine Weather API," [Online]. Available: https://stormglass.io

29.Indian National Centre for Ocean Information Services (INCOIS), "Ocean State Forecast and Services," [Online]. Available: https://www.incois.gov.in