**A Real-time Research Project / Societal Related Project**

**Report on**

### SafeStreet – Road damage detection and alert system

Submitted in Partial fulfillment of requirements for B.Tech II Year II Semester course

**By**

**Aarti Deshpande 23BD1A0501**

**Anisha Paturi 23BD1A050E**

**Ch. Parinamika Bhanu 23BD1A0518**

**Ch. Vahini 23BD1A051D**

**M. Javali Reddy 23BD1A05IW**

**Sravani Janak 23BD1A051Y**

**T. Jayashree Indrani 23BD1A051**

**Under the guidance of**

***Ms. Nidhi Srivastav***

***Assistant Professor, Information Technology***





**KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY   
(AN AUTONOMOUS INSTITUTION)  
Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH**

### Narayanaguda, Hyderabad, Telangana-29

**CERTIFICATE**

This is to certify that this is a bonafide record of the project report titled **“SAFESTREET – ROAD DAMAGE DETECTION AND ALERT SYSTEM”** which is being presented as the **Real-time Research Project / Societal Related Project** report by

1. **Aarti Deshpande 23BD1A0501**
2. **Anisha Paturi 23BD1A050E**
3. **Ch. Parinamika Bhanu 23BD1A0518**

**4) Ch. Vahini 23BD1A051D**

**5) M. Javali Reddy 23BD1A05IW**

**6) Sravani Janak 23BD1A051Y**

**7) T. Jayashree Indrani 23BD1A051Z**

In partial fulfillment for the II Year II Semester Course RTRP in KMIT affiliated to the Jawaharlal Nehru Technological University Hyderabad, Hyderabad.

**Mentors**

**Ms. Nidhi Srivastav**

**Program Coordinator**

**Mr. Venkat Boinapally**

**Mr. Srikanth Reddy**

Submitted for Final Project Review held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Vision & Mission of KMIT**

**Vision of KMIT**

* + To be the fountainhead in producing highly skilled, globally competent engineers.
  + Producing quality graduates trained in the latest software technologies and related tools and striving to make India a world leader in software products and services.

# Mission of KMIT

* + To provide a learning environment that inculcates problem solving skills, professional, ethical responsibilities, lifelong learning through multi modal platforms and prepares students to become successful professionals.
  + To establish an industry institute Interaction to make students ready for the industry.
  + To provide exposure to students on the latest hardware and software tools.
  + To promote research-based projects/activities in the emerging areas of technology convergence.
  + To encourage and enable students to not merely seek jobs from the industry but also to create new enterprises.
  + To induce a spirit of nationalism which will enable the student to develop, understand India's challenges and to encourage them to develop effective solutions.
  + To support the faculty to accelerate their learning curve to deliver excellent service to students.

# DECLARATION

We hereby declare that the results embodied in the dissertation entitled **“**Safestreet – road damage detection and alert system” has been carried out by us together during the academic year 2024-25 as a partial fulfillment of the B.Tech II Year II Semester Course “**Real-time Research Project / Societal Related Project**”. We have not submitted this report to any other Course/College.

### Student Name Rollno.

### Aarti Deshpande 23BD1A0501

### Anisha Paturi 23BD1A050E

### Ch. Parinamika Bhanu 23BD1A0518

### Ch. Vahini 23BD1A051D

### M. Javali Reddy 23BD1A051W

### Sravani Janak 23BD1A051Y

### T. Jayashree Indrani 23BD1A051Z

# 

# ACKNOWLEDGEMENT

We take this opportunity to thank all the people who have rendered their full support to our project work. We render our thanks to **Dr. B L Malleswari**, Principal who encouraged us to do the Project.

We are grateful to **Mr. Neil Gogte,** Founder & Director and **Mr. S. Nitin,** Director,for facilitating all the amenities required for carrying out this project.

We express our sincere gratitude to **Ms. Deepa Ganu**, Director Academic for providing an excellent environment in the college.

We are also thankful to **Mr. Venkat Boinapally and Mr. Srikanth Reddy**,

**Real-Time Research Project** Program Coordinator for providing us with

time to make this project a success within the given schedule.

We are also thankful to our Project Mentor **Ms. Nidhi Srivastav**, for her valuable guidance and encouragement given to us throughout the project work.

We would like to thank the entire KMIT faculty, who helped us directly and indirectly in the completion of the project.

We sincerely thank our friends and family for their constant motivation during the project work.

## Student Name Roll no.

1. **Aarti Deshpande 23BD1A0501**
2. **Anisha Paturi 23BD1A050E**
3. **Ch. Parinamika Bhanu 23BD1A0518**
4. **Ch. Vahini 23BD1A051D**
5. **M. Javali Reddy 23BD1A051W**
6. **Sravani Janak 23BD1A051Y**
7. **T. Jayashree Indrani 23BD1A051Z**

# ABSTRACT

SafeStreet is a mobile and web application designed to improve urban infrastructure management by enabling citizens to report road damages directly to the authorized people. This project streamlines the process of identifying, tracking, and resolving issues such as potholes, cracks, and damaged waterlogging.

The system offers a user-friendly interface for civilians to submit detailed damage reports, including images, location and date. These reports are analyzed using Natural Language Processing (NLP) techniques to classify the severity of each case, assisting supervisors in prioritizing responses. Authorities can manage reports via a dashboard that features data visualizations, status updates.

Key features include OTP-based secure login, email notifications, role-based access for users and supervisors, and a backend built with Node.js and MongoDB. By digitizing the reporting workflow and incorporating intelligent classification, SafeStreet enhances transparency, accountability, and efficiency in urban maintenance operations.

**CONTENTS**

DESCRIPTION PAGE [CHAPTER - 1 1](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.2et92p0)

* 1. [INTRODUCTION 2-4](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.tyjcwt)
     1. Purpose of the project 2
     2. [Proposed System 3](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.3dy6vkm)
     3. [Scope of the Project](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.4d34og8) 3
     4. [Architecture Diagram 4](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.2s8eyo1)

[CHAPTER – 2 5](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.17dp8vu)

* 1. [LITERATURE SURVEY 6-7](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.3rdcrjn)

[CHAPTER - 3 8](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.26in1rg)

* 1. SOFTWARE REQUIREMENT SPECIFICATION 9-12
     1. [Functional Requirements 10](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.44sinio)
     2. [Non-Functional Requirements 11](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.2jxsxqh)
     3. Technology Stack
     4. Use case Diagram
     5. Database Design
     6. Wireframes(Mock up Screens) [11](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.z337ya)
     7. Deployment Diagram

[CONCLUSION](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.4f1mdlm) 52

[FUTURE ENHANCEMENTS](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.2u6wntf) 53

[REFERENCES](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.19c6y18) 54

[BIBLIOGRAPHY](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.3tbugp1) (Put Your Project Github link here) 55

# CHAPTER-1

# 1.INTRODUCTION

# Purpose of the project

The goal of the SafeStreet project is to improve the upkeep of the roads by providing a smart and effective way of detecting, analyzing, and reporting damages in real-time. The manual procedures of inspecting a road often involve a lot of time, heavy labor, and varying degrees of precision. These challenges can be resolved with the help of computer vision, deep learning, and modern web technologies which will automate the whole process, enabling SafeStreet to do more.

The system is designed with a mobile application for field workers and a backend operated by a Vision Transformer (ViT) model. Maintenance team members and municipal authorities can now classify the images into set categories which enables image classification to be done more effectively. This helps in timely assistance and accurate worksheets being used which helps in the optimal use of set resources. SafeStreet provides timely alerts and detailed reports, promoting safer roads, minimizes vehicle damage and accidents, and improves overall public infrastructure

management.

# Proposed System

The proposed solution for SafeStreet is to introduce a system with integrated mobile, web and AI technologies to create a complete End-to-End solution for road damage detection and alerting. Each of the following components contributes to the complete functioning system:

* Mobile Application: On-field users can take images of the damaged roads conveniently using the mobile application, which can then be uploaded to the backend for processing.
* ViT Model Backend Analysis: The model analyzes the images using a pre-trained Vision Transformer (ViT) model. It sorts the damage (pothole, crack, erosion) and gauges how serious it is before producing a summary sentence.
* Automated Alert System: An image report is produced and dispatched through automated email to concerned authorities. The email details includes sophisticated damage classification, default severity marker and the exact date and time of image upload
* MERN Stack Web Platform: It has a fully developed web dashboard using MERN stack (MongoDB, Express.js, React.js, Node.js) which enables users to access visual analytics, historical records of road damage data for better insight and decision making.

The system optimizes the manual efforts required for maintenance and further helps in the proactive management of urban structures. This in turn enables faster response times for maintenance and helps in improving urban mobility.

**Proposed/To-be Work-Flow**

Figure 1.1

* 1. **Scope**

The scope of the SafeStreet project is to design and develop an AI-driven road damage detection and reporting system that streamlines road maintenance operations for urban and municipal authorities. The system leverages computer vision techniques, specifically a Vision Transformer (ViT) model, to automatically detect various types of road surface damages such as potholes, cracks, and surface wear from images, enabling continuous, automated monitoring without the need for manual inspections. It includes a mobile application for field workers and citizen users to upload geo-tagged images and receive real-time feedback on road conditions, as well as a web-based dashboard for administrators to visualize reported issues, track repair statuses, assign tasks, and analyze damage patterns over time. The project integrates predictive analytics to forecast high-risk areas based on historical data, supporting proactive maintenance planning.

Built using a MERN stack and integrated with RESTful APIs and cloud-hosted AI models, the solution is scalable and adaptable to different regions. By automating detection and reporting, prioritizing critical issues, and providing data-driven insights, SafeStreet aims to make road maintenance more efficient, cost-effective, and safer for the public. Future enhancements may include integration with IoT devices or drones, multilingual support, and dynamic routing for maintenance teams, further strengthening its role as a smart urban infrastructure solution.

* 1. **Architecture Diagram**

The architecture of the SafeStreet system is designed using a modular and scalable approach that integrates a mobile application, web dashboard, AI model, backend server, and database. At the top layer, users interact with the system through two primary interfaces: the mobile application and the web dashboard.

* The mobile application, built with React Native, is used by field workers and citizen users to capture images of road surfaces and submit reports. It includes features such as email notifications, real-time damage feedback, and interactive maps powered by Leaflet.js to display geolocated issues.
* The web dashboard, developed using React.js, HTML, and CSS, is tailored for administrative users and municipal authorities. It enables them to monitor incoming reports, view analytics, track repair progress, and manage tasks through a user-friendly interface.

Both the mobile app and the web dashboard communicate with the backend, which is built using Node.js and Express.js. The backend handles API requests, manages data flow, and integrates with external services.

For AI-based damage detection, the system utilizes a Vision Transformer (ViT) model hosted on Hugging Face. Images uploaded from the mobile app are sent to this model for inference, and the results are returned to users in real-time.

All application data including user profiles, reported issues, image metadata, and repair logs is stored in a MongoDB database. MongoDB ensures high availability and flexible document-based storage, which is well-suited for the dynamic structure of user-generated reports.

This architecture enables seamless interaction between system components, supports real-time damage detection, and ensures scalability for deployment across different cities or regions.

Figure 1.2

# CHAPTER-2

**2. LITERATURE SURVEY**

**2.1 Why Road Maintenance Matters**

We rely on roads every day—whether it's for commuting to work, transporting goods, or just getting from point A to point B. But roads don’t stay in good shape on their own. Cracks, potholes, and worn-out markings slowly creep in, making travel less safe and more frustrating. Traditionally, detecting road damage meant sending people out to manually inspect streets—a process that’s not only slow and costly but also often inaccurate. With cities growing and roads aging, we need smarter, faster ways to keep our infrastructure in check.

**2.2 Smarter Detection with AI and Computer Vision**

Thanks to recent breakthroughs in Artificial Intelligence (AI) and computer vision, spotting road damage no longer has to be a manual chore. AI models—especially advanced ones like Vision Transformers (ViT)—can now analyze road images and accurately detect issues like:

Potholes

Surface cracks (longitudinal, transverse, or fatigue-related)

Faded markings

Eroded edges

Unlike older models that focused on smaller image areas, ViTs look at the whole picture using a technique called self-attention, which makes them great at spotting even subtle damage. In real-world testing on datasets like RDD2020, these models achieved accuracy rates over 90%, proving they’re not just fast—they’re reliable too. SafeStreet builds on this with a multi-label model, meaning it can detect more than one issue in a single photo—just like a human would.

**2.3** **How Mobile Apps Put the Power in Your Hands**

Imagine seeing a pothole and being able to report it in just a few taps. That’s exactly what mobile apps like SafeStreet’s aim to do. Built using React Native, the app lets anyone—citizens or city workers—report issues by:

Taking or uploading a photo

Automatically tagging their location

Optionally describing the damage

The app supports multiple languages and is designed to be intuitive even for users who aren’t tech-savvy. By making it easy to report problems on the go, the app turns every citizen into a contributor to safer roads.

**2.4 The Tech Behind the Scenes: Backend Systems**

While the app gets most of the attention, the real magic happens behind the scenes. SafeStreet’s backend uses cloud-based infrastructure and RESTful APIs to handle everything from image uploads to AI analysis. With a MongoDB database and scalable cloud services like AWS or GCP, the system can handle thousands of reports without slowing down.

Right now, AI processing is done in the cloud for higher accuracy, but there's also support for edge computing in the future—meaning the app could one day run AI directly on users’ phones, especially useful in low-connectivity areas.

**2.5 Talking to a System That Actually Listens: Chatbots & NLP**

Not everyone wants to fill out forms or navigate apps. Some just want to type or speak naturally—and get things done. That’s where chatbots powered by Natural Language Processing (NLP) come in. SafeStreet uses AI models like Gemini or GPT to enable a friendly, conversational interface where users can:

Report an issue

Check on the status of their report

Ask about different types of road damage

Get contact info for local maintenance departments

The chatbot supports multiple languages and is smart enough to handle unclear inputs or escalate to human help when needed. It makes the system more inclusive and easier to use for everyone.

**2.6 Staying Updated: Notifications That Matter**

People want to know what’s happening after they report a problem. SafeStreet keeps everyone in the loop with real-time notifications through:

Email updates

In-app alerts and push notifications

SMS (especially for users in areas with low internet access)

These alerts update users when their issue is received, when action is taken, and when the repair is complete. It builds transparency and trust—two things that are often missing in public service systems.

**2.7 Following Through: From Report to Resolution**

SafeStreet doesn’t stop at just identifying the problem—it sees it through to the end. Here’s how the full cycle works:

A user reports a problem through the app or chatbot

The image is processed by AI to detect and classify the damage

Local authorities review and validate the report

Maintenance teams are assigned and repairs are made

The issue is marked as resolved, with an updated photo

The original reporter gets a follow-up and can rate the resolution

This kind of end-to-end workflow ensures issues are not just tracked—they’re solved, and people are kept informed throughout.

**2.8 Respecting Privacy and Building Trust**

With photos and location data involved, privacy is a serious concern—and SafeStreet takes it seriously. The platform follows global data protection standards like GDPR and India’s DPDP Bill. Key measures include:

Asking for clear user consent before collecting data

Encrypting all data in storage and transit

Automatically blurring faces or license plates in images

Limiting access to sensitive data through role-based permissions

By putting privacy first, SafeStreet ensures that helpful technology doesn’t come at the cost of personal safety.

**2.9 How SafeStreet Compares to Other Systems**

Many platforms exist out there—some use Google Street View, some rely on crowd-sourced photos, others use AI. But most only cover part of the picture. Here’s how SafeStreet stands out:

|  |  |  |
| --- | --- | --- |
| Feature | SafeStreet | Others |
| Real-time AI detection | Yes | No or Limited |
| Multi-label damage classification | Yes | No |
| Mobile & web support | Yes | Only web |
| End-to-end lifecycle tracking | Yes | No |
| Chatbot | Yes | Rare |
| Data privacy compliance | Yes | Not always clear |

SafeStreet is built as a complete platform, combining the best of AI, user experience, and public service design.

**2.10 Final Thoughts: A Smarter Way to Fix Our Streets**

Road maintenance doesn’t have to be slow, frustrating, or inefficient. With tools like SafeStreet, we can turn every citizen into a sensor, every report into an insight, and every issue into an opportunity for smarter city management. By combining cutting-edge AI, thoughtful design, and real-world usability, SafeStreet isn’t just another reporting app—it’s a step toward building cities that work better for everyone.

# CHAPTER-3

**3.1** **Functional Requirements**

1. User Registration and Authentication

- Users can register with valid personal email addresses.

- Users can log in using email and password.

- OTP verification is required for password reset.

2. Image Upload and Damage Reporting

- Users can upload images of road damages via mobile app.

- Metadata such as location, date, and time is captured with each upload.

- AI model classifies damage type and severity automatically.

- Users can submit damage reports which are stored in the backend.

3. Damage Report Management

- Authorities can view, filter, and search damage reports by severity, status, date, and location.

- Authorities can update report status (Pending, Ignore, Resolved).

- Summary reports are generated and accessible.

4. Notifications and Communication

- Automated email notifications are sent for report updates and OTPs.

- NLP-powered chatbot assists users with queries and support.

- Future support for push notifications and SMS alerts.

5. AI and NLP Integration

- Integration with Vision Transformer (ViT) for image classification.

- Chatbot powered by Google Gemini generative AI.

- NLP classification of user intents for chatbot interactions.

6. Security and Privacy

- Secure storage and transmission of user data.

- Role-based access control for different user types.

- Compliance with data protection regulations.

7. User Interface

- Responsive and intuitive mobile app UI using React Native and Expo.

- Web dashboard for authorities with React.

- Multi-language support planned for future releases.

8. Backend Services

- RESTful APIs for authentication, uploads, reports, and chatbot.

- MongoDB for data persistence.

- Error handling and logging mechanisms.

9. Performance and Scalability

- Efficient AI inference for timely responses.

- Scalable backend architecture to handle growing users and data.

10. Additional Features (Future Enhancements)

- Offline mode for mobile app.

- Integration with government databases.

- Advanced analytics and reporting.

- Augmented reality features for damage visualization.

**3.2 Non-Functional Requirements**

1. Performance

- The system shall process image uploads and return damage classification results within a minute.

- The chatbot shall respond to user queries within 15 seconds.

- The web and mobile applications shall load primary screens within 10 seconds under normal network conditions.

2. Scalability

- The backend services shall support concurrent usage by up to 10,000 users without degradation in performance.

- The system architecture shall allow easy scaling of AI model inference services and database storage.

3. Security

- All data transmissions shall be encrypted using TLS/SSL protocols.

- User authentication shall enforce strong password policies and OTP verification.

- Role-based access control shall restrict sensitive operations to authorized users only.

- Sensitive data such as passwords and OTPs shall be stored securely using hashing and encryption.

4. Usability

- The mobile and web applications shall have intuitive user interfaces with consistent navigation.

- The system shall provide clear error messages and guidance for user actions.

- Accessibility standards (e.g., WCAG) shall be considered to support users with disabilities.

5. Reliability

- The system shall have an uptime of 99.9% excluding scheduled maintenance.

- Automated backups of databases shall be performed daily to prevent data loss.

- The system shall gracefully handle failures and provide meaningful feedback to users.

6. Maintainability

- The codebase shall follow standard coding conventions and be well-documented.

- Modular architecture shall be used to facilitate updates and feature additions.

- Logging and monitoring tools shall be integrated to assist in troubleshooting.

7. Portability

- The mobile app shall support both Android and iOS platforms.

- The web application shall be compatible with major browsers (Chrome, Firefox, Edge, Safari).

8. Compliance

- The system shall comply with relevant data protection regulations (e.g., GDPR).

- User data privacy shall be ensured throughout data collection, storage, and processing.

9. Extensibility

- The system design shall allow integration of additional AI models and third-party services in the future.

- APIs shall be designed to support new features without breaking existing functionality.

10. Localization

- The system shall support multiple languages to cater to diverse user bases (planned for future releases).

**3.3 Technology Stack**

Frontend

- React Native: Framework for building native mobile apps using React.

- Expo: Toolchain for React Native development, simplifying build and deployment.

- React Navigation: Routing and navigation for React Native apps.

- TypeScript: Typed superset of JavaScript used for safer and scalable code.

- Expo Router: File-based routing for Expo apps.

- React: JavaScript library for building user interfaces (used in web app).

- Vite: Build tool for frontend web development.

- CSS: Styling for web app components.

Backend

- Node.js: JavaScript runtime for backend services.

- Express.js: Web framework for Node.js to build RESTful APIs.

- Python Flask: Lightweight Python web framework for AI backend services.

- MongoDB: NoSQL database for storing user data, reports, and metadata.

- Mongoose: ODM for MongoDB in Node.js.

- Nodemailer: Node.js module for sending emails.

- Natural: NLP library for intent classification in chatbot.

- Google Gemini AI: Generative AI used for chatbot support.

- Vision Transformer (ViT): AI model for image classification of road damages.

AI and Machine Learning

- Vision Transformer (ViT): Transformer-based model for image recognition.

- Google Gemini: Generative AI for chatbot conversational capabilities.

- Natural Language Processing (NLP): Techniques for intent classification and chatbot interaction.

Tools and Utilities

- Git: Version control system.

- npm: Node.js package manager.

- Babel: JavaScript compiler for compatibility.

- ESLint: Linting tool for JavaScript/TypeScript.

- Prettier: Code formatter.

- Jest: Testing framework for JavaScript.

- Expo CLI: Command line interface for Expo.

- Ngrok: Tool to expose local servers to the internet for testing.

Deployment and Hosting

- MongoDB Atlas: Cloud-hosted MongoDB service.

Development Environment

- Node.js: Runtime environment.

- Python: Programming language for AI backend.

- Expo Go: Mobile app for testing Expo projects on devices.

Other Libraries and Dependencies

- React Native Reanimated: Animation library for React Native.

- React Native Gesture Handler: Gesture handling for React Native.

- React Native Safe Area Context: Handling safe area insets.

- React Native Screens: Native navigation primitives.

- Axios: Promise-based HTTP client for API requests.

- Dotenv: Environment variable management.

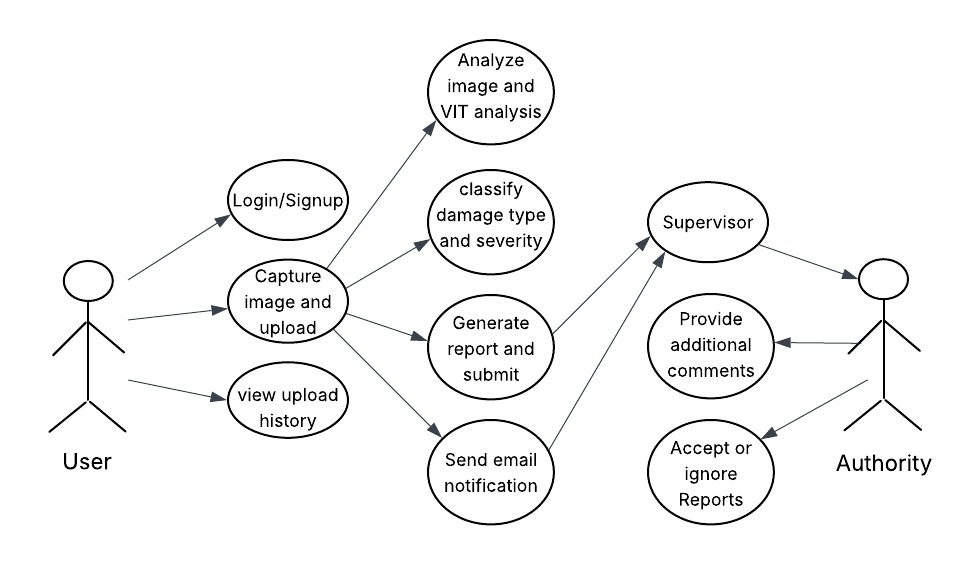
 **3.4 Use case Diagram**

Figure 3.1

* 1. **Database Design**

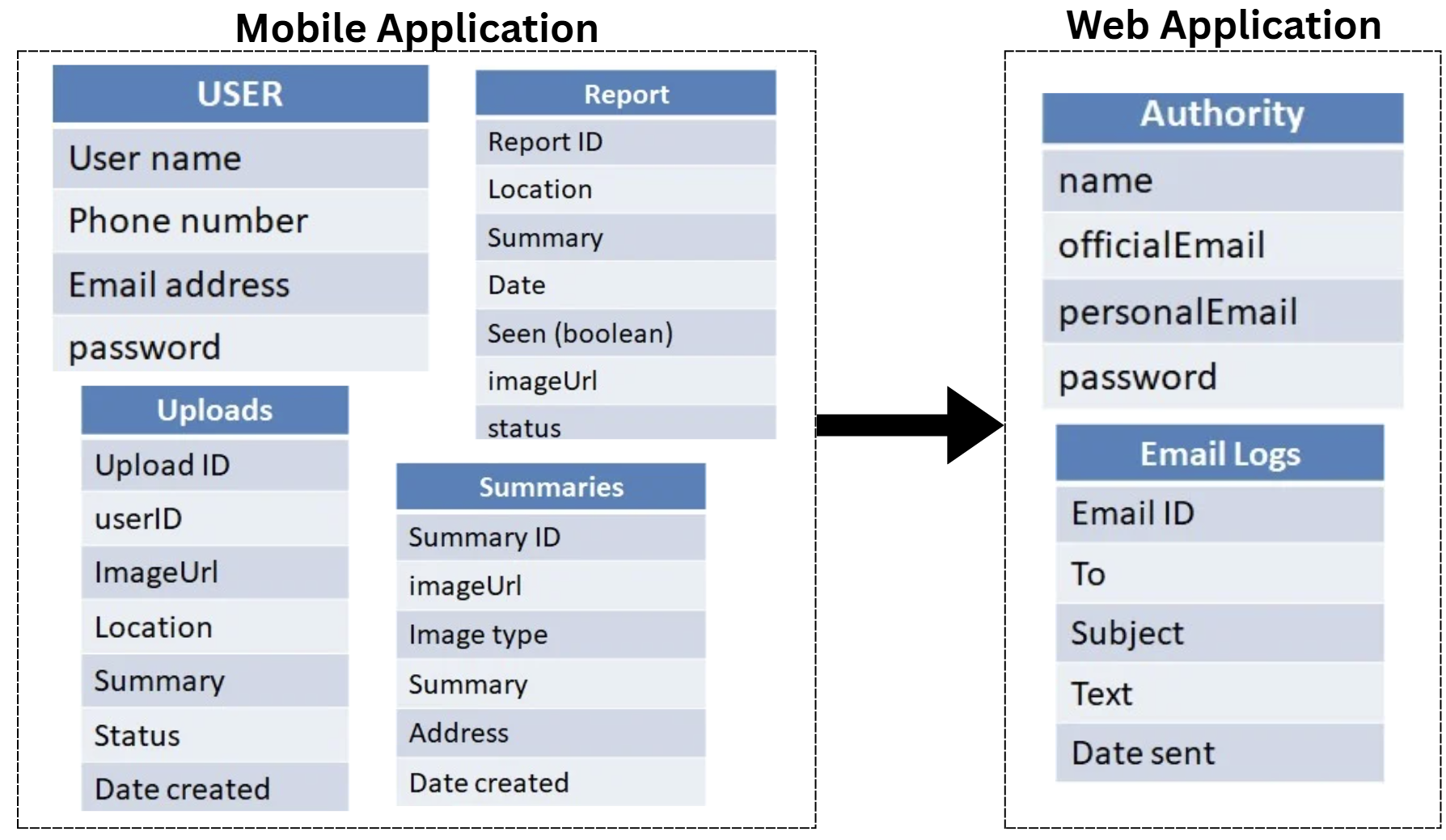


Figure 3.2

**3.6 Wireframes(Mock up Screens)**

**Web**



Figure 3.3

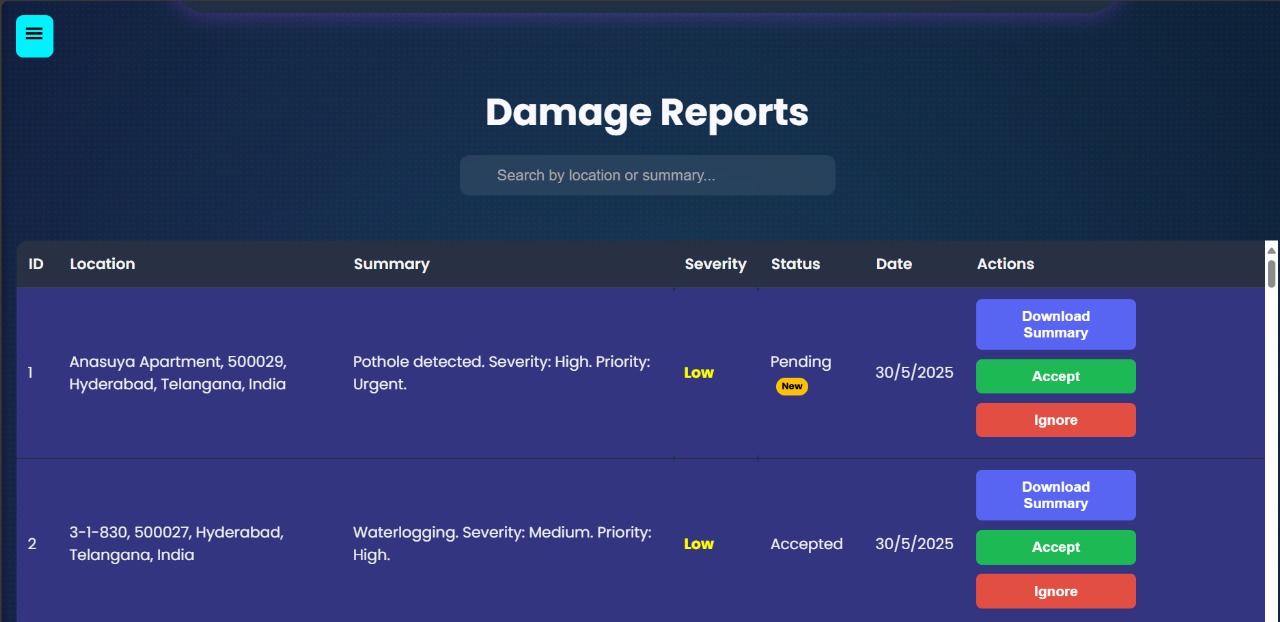
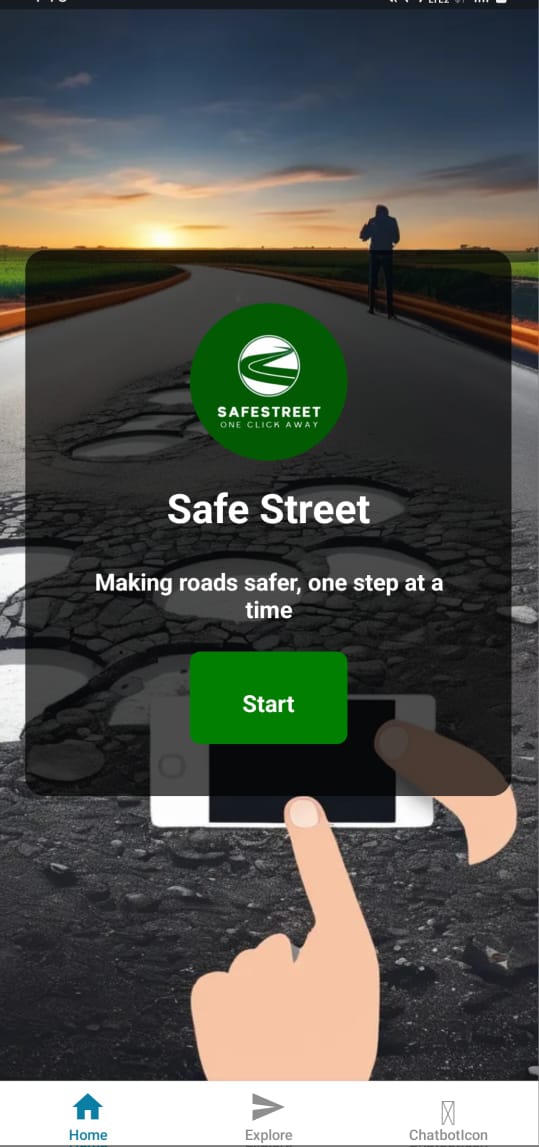
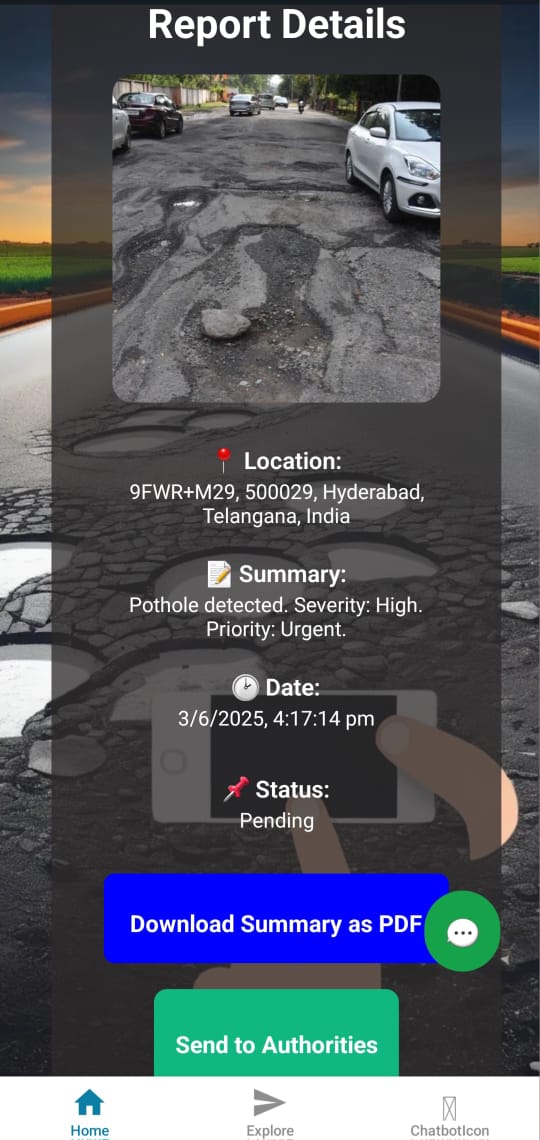
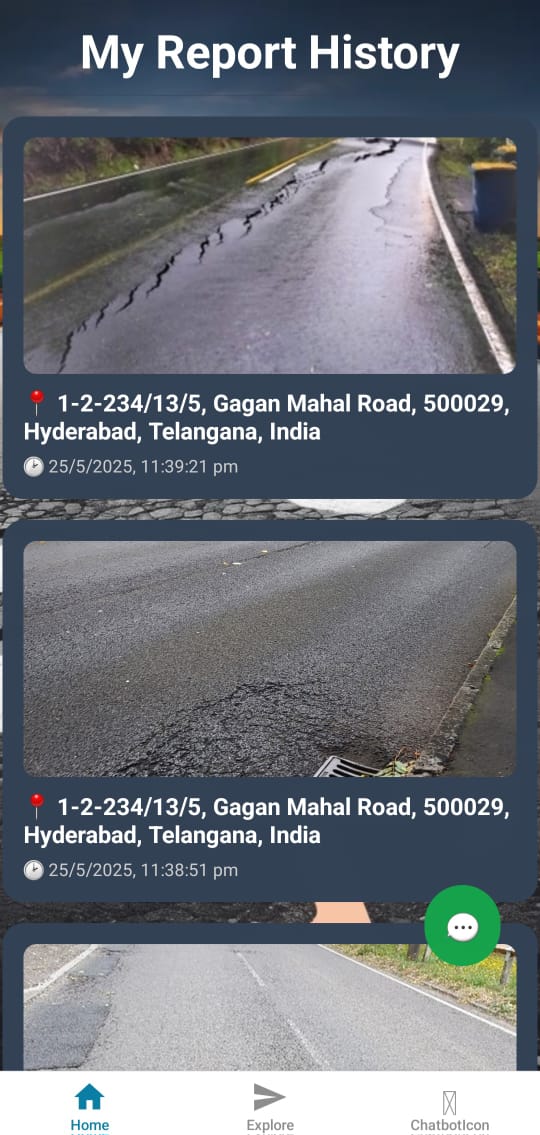


Figure 3.4

**App**



* 1. **Deployment Diagram**

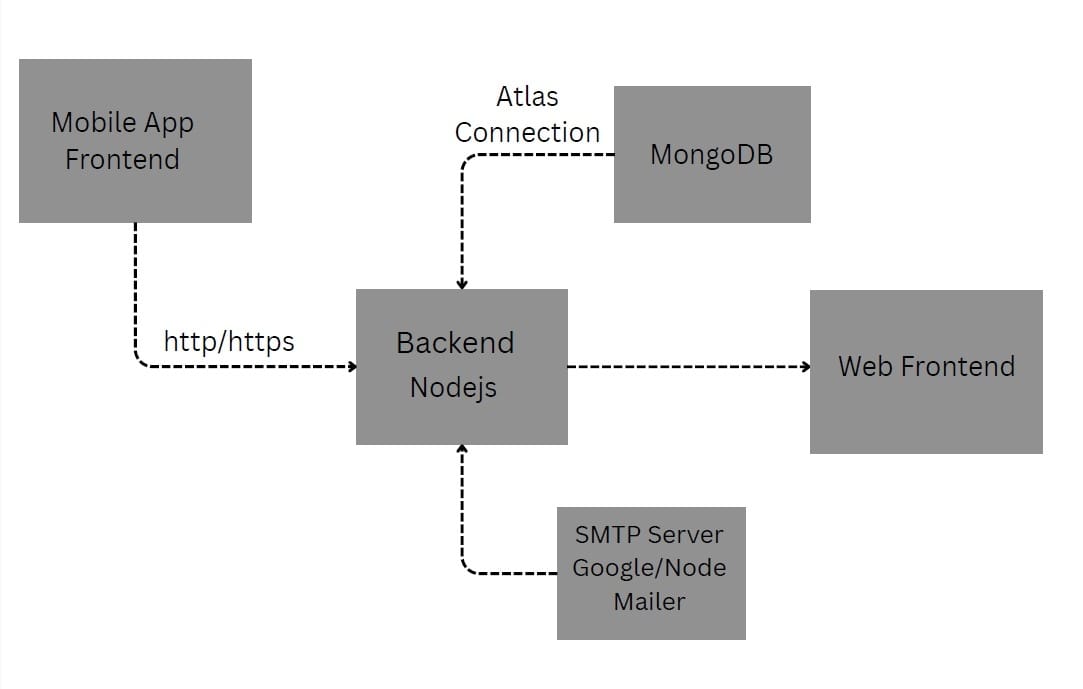


Figure 3.

[**CONCLUSION**](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.4f1mdlm)

The SafeStreet project marks a significant advancement in road maintenance and damage detection by integrating state-of-the-art AI technologies with intuitive mobile and web platforms. This system transforms the traditionally manual and labor-intensive process of road damage reporting and management into a scalable, efficient, and user-friendly solution.

The mobile application, developed using React Native and Expo, enables users to effortlessly upload images of road damages. These images are analyzed by advanced AI models, including Vision Transformers (ViT), to accurately classify damage types and assess severity. The Python Flask backend supports AI-powered damage classification and offers chatbot assistance powered by Google’s Gemini generative AI, enhancing user engagement and support.

Conversely, the web application, built with React and Node.js, provides a comprehensive management platform for authorities. It features secure user authentication with OTP verification, robust damage report management, automated email notifications, and an NLP-driven chatbot for effective query resolution. The backend utilizes MongoDB for data persistence and employs natural language processing to interpret user intents, thereby improving system responsiveness and usability.

Throughout development, modern technologies and best practices were employed to ensure the system’s maintainability, scalability, and security. The project exemplifies the potential of combining AI, mobile computing, and web technologies to address complex infrastructure challenges.

Looking ahead, the project offers numerous avenues for enhancement, such as real-time video-based damage detection, multi-modal AI models, offline functionality, advanced analytics dashboards, multi-language support, integration with governmental databases, and augmented reality features. These enhancements will further establish SafeStreet as a pioneering and comprehensive solution for road maintenance.

In conclusion, SafeStreet not only automates and enhances the accuracy of road damage detection but also fosters effective communication and management through its integrated platforms. It stands as a valuable tool for both authorities and citizens, contributing to safer and better-maintained roadways.

[**FUTURE ENHANCEMENTS**](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.2u6wntf)

1. Integration of real-time damage detection using video streams from dashcams or drones.

2. Enhanced AI models with multi-modal data (images, sensor data, GPS) for better accuracy.

3. Offline mode support in the mobile app for data collection in areas with poor connectivity.

4. Advanced analytics dashboard in the web app with predictive maintenance insights.

5. Multi-language support for wider accessibility.

6. Integration with government or municipal databases for automated work order generation.

7. User role management and permissions in the web platform.

8. Push notifications and SMS alerts in addition to email.

9. Continuous learning pipeline to improve AI models with new data.

10. Augmented reality (AR) features in the mobile app for better damage visualization.

11. Integration with GIS (Geographic Information Systems) for spatial analysis and mapping of road damages.

12. Implementation of blockchain for secure and transparent reporting and maintenance records.

13. AI-powered severity prediction to prioritize repairs based on traffic and environmental factors.

14. Integration with IoT devices for continuous road condition monitoring.

15. Social media integration for public reporting and awareness campaigns.

16. Automated scheduling and dispatching of repair crews based on report data.

17. Enhanced security features including multi-factor authentication and data encryption.

18. Customizable reporting templates and export options (PDF, Excel).

19. Voice command support in the mobile app for hands-free reporting.

[**REFERENCES**](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.19c6y18)

**Academic Papers and Articles**

1. Maeda, H., Sekimoto, Y., Seto, T., Kashiyama, T., & Omata, H. (2018). Road Damage Detection Using Deep Neural Networks with Images Captured Through a Smartphone. arXiv preprint arXiv:1801.09454.

2. Dosovitskiy, A., et al. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv preprint arXiv:2010.11929.

3. Zhang, Y., et al. (2021). Vision Transformer for Road Damage Detection. IEEE Transactions on Intelligent Transportation Systems.

4. Jurafsky, D., & Martin, J. H. (2021). Speech and Language Processing (3rd ed.). Draft chapters available online.

5. Vaswani, A., et al. (2017). Attention is All You Need. Advances in Neural Information Processing Systems.

**Libraries, Frameworks, and Tools**

1. React Native - https://reactnative.dev/

2. Expo - https://expo.dev/

3. Flask - https://flask.palletsprojects.com/

4. React - https://reactjs.org/

5. Node.js - https://nodejs.org/

6. Express.js - https://expressjs.com/

7. MongoDB - https://www.mongodb.com/

8. Mongoose - https://mongoosejs.com/

9. Natural (NLP library) - https://github.com/NaturalNode/natural

10. Nodemailer - https://nodemailer.com/

11. Vision Transformer (ViT) - https://github.com/google-research/vision\_transformer

**Additional Resources**

1. Expo Router Documentation - https://expo.github.io/router/docs

2. React Router Documentation - https://reactrouter.com/

3. MongoDB Atlas - https://www.mongodb.com/cloud/atlas

4. Nodemailer Email Sending Guide - https://nodemailer.com/about/

[**BIBLIOGRAPHY**](https://docs.google.com/document/d/1vNwnyGx9h1yMU6_gVJ2-YRqX4bTSdHyv/edit#heading%3Dh.3tbugp1)