RoadRadar: IoT-Driven Road Damage Detection and Reporting

Project Team

Muhammad Rafay 20P-0018
Muhammad Suliman 21P-8047

Session 2021-2025

Supervised by

Dr. Ali Sayyed

Department of Computer Science

National University of Computer and Emerging Sciences
Peshawar, Pakistan

June, 2025

Student's Declaration

We hereby declare that the thesis submitted is our own original work, completed independently without any unauthorized assistance. Only the sources and resources explicitly cited have been utilized. All experts, whether quoted directly or paraphrased, have been properly acknowledged.

If generative AI tools were used, we have clearly indicated the product name, manufacturer, software version, and the specific purpose (e.g., language, enhancement, system research). We take full responsibility for the selection, application, and interpretation of any AI-generated content incorporated into this work. A list of prompts along with corresponding page numbers where they are used is included in the appendix.

We declare that this project titled "*RoadRadar: IoT-Driven Road Damage Detection and Reporting*", submitted as requirement for the award of degree of Bachelors in Software Engineering, does not contain any material previously submitted for a degree in any university; and that to the best of our knowledge, it does not contain any materials previously published or written by another person except where due reference is made in the text.

We understand that the management of Department of Computer Science, National University of Computer and Emerging Sciences, has a zero tolerance policy towards plagiarism. Therefore, We, as authors of the above-mentioned thesis, solemnly declare that no portion of our thesis has been plagiarized and any material used in the thesis from other sources is properly referenced.

We further understand that if we are found guilty of any form of plagiarism in the thesis work even after graduation, the University reserves the right to revoke our BS degree.

Muhammad Rafay Signature: _____

Muhammad Suliman Signature: _____

_____

Verified by Plagiarism Cell Officer

Dated:

Certificate of Approval

The Department of Computer Science, National University of Computer and Emerging Sciences, accepts this thesis titled *RoadRadar: IoT-Driven Road Damage Detection and Reporting*, submitted by Muhammad Rafay (20P-0018), and Muhammad Suliman (21P-8047), in its current form, and it is satisfying the dissertation requirements for the award of Bachelors Degree in Software Engineering.

Supervisor

Dr. Ali Sayyed Signature: _____

Mr. Haroon Zafar

FYP Coordinator
National University of Computer and Emerging Sciences, Peshawar

---

Dr. Qasim Jan

HoD of Department of Computer Science
National University of Computer and Emerging Sciences

Acknowledgements

We would like to express our deepest gratitude to our teachers, whose dedication and guidance have been invaluable throughout our academic journey. Special thanks to Dr. Ali Sayyed, our supervisor, for his unwavering support, insightful feedback, and encouragement that helped shape this work.

We are also immensely grateful to our parents for their endless love, encouragement, and sacrifices. Their unwavering belief in us has been our greatest source of strength and motivation.

Thank you to all who have contributed to our growth and success. Your support and encouragement have made this journey possible.

Muhammad Rafay

Muhammad Suliman

Abstract

The issue being examined pertains to the shortcomings of manual road damage identification, resulting in postponed repairs, increased accident hazards, and accelerated degradation of road infrastructure. To tackle this issue, our approach employs advanced AI methods, particularly lightweight deep learning models (YOLOv5n) and IoT technologies. YOLOv5n is used to examine real-time road images, identify irregularities like potholes and cracks, and deliver accurate classifications. In the meantime, integration of IoT enables GPS-driven location monitoring and cloud-supported reporting. Integrating these cutting-edge technologies into our road damage detection system allows us to provide authorities with immediate feedback, practical insights, and automated reports, facilitating prompt maintenance and improving road safety. Expected main outcomes of this integration involve less manual work, better management of road infrastructure, uniform maintenance strategies, lower accident risks, and increased safety for road users. To sum up, the combination of YOLOv5n and IoT technologies in our approach to road damage detection represents a notable improvement in infrastructure observation. By skillfully employing artificial intelligence, we aim to enhance road condition management and public safety while promoting a culture of technological advancement in smart city development.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This study aims to investigate the incorporation of sophisticated deep learning models and IoT technologies into road damage detection techniques to improve infrastructure management strategies. The issue under examination is the ineffectiveness and inaccuracy of conventional manual road inspection techniques, which do not adequately respond to the increasing challenges of road maintenance promptly. In today's urban development scene, where road safety and infrastructure dependability are crucial, conventional methods frequently fail to provide adequate monitoring and timely responses.

This research aims to utilize lightweight deep learning models like YOLOv5n to identify road anomalies such as potholes and cracks in real-time, building on earlier progress in artificial intelligence, computer vision, and IoT systems. At the same time, the integration of IoT provides accurate location tracking via GPS and instant updates through a cloud-driven dashboard. The thesis argues that integrating these technologies into road damage detection methods will greatly improve the speed, precision, and overall effectiveness of infrastructure management, thereby decreasing accident risks and extending the lifespan

The overall strategy entails creating an extensive road damage identification system that integrates deep learning technologies for anomaly recognition with IoT-enabled real-time reporting systems. The effectiveness of this study will be assessed using established criteria, such as quantifiable enhancements in road maintenance response times, the precision of anomaly detection, and favorable feedback from stakeholders concerning the usability

and efficiency of the system. Recent progress in AI-based monitoring and IoT technologies illustrates the capability of transforming conventional road inspection techniques, creating opportunities for more intelligent and secure urban infrastructure.

1.1 Integration Overview

We have explored the difficulties related to existing road damage detection techniques and the possible advantages of incorporating deep learning models and IoT solutions into infrastructure management strategies. We have concentrated on how these cutting-edge technologies can improve the precision, effectiveness, and promptness of road maintenance, ultimately resulting in safer and more dependable transportation systems.

1.1.1 Objectives and Scope of the Research

Our main objective was to examine how well combining deep learning models with IoT technologies enhances the accuracy and promptness of road damage detection approaches for road maintenance. We have carried out several experiments to assess how well our AI-based road damage detection system performs compared to conventional manual inspection techniques. We have collected input from relevant stakeholders to evaluate the effectiveness and feasibility of the suggested method in actual infrastructure management situations.

Chapter 2

Review of Literature

2.1 Review of Literature

The literature review conducted in this research offers an in-depth analysis of six foundational papers centered on automating road damage detection through IoT and deep learning methodologies. These articles, released from 2020 to 2024, explore the use of sophisticated techniques like machine learning, deep learning, and IoT-enabled systems to improve the precision, effectiveness, and relevance of road damage detection systems. By critically analyzing these papers, this literature review aims to clarify the existing knowledge in this area, highlight important findings, and emphasize their relevance to the wider field of smart infrastructure management

A study[1] published in 2021, presents a deep learning approach using dilated convolution for detecting potholes. This approach enhances precision but encounters difficulties with computational demand and adapting to various road conditions.

A study[2] published in 2023, presents an IoT-driven approach for identifying potholes and bumps on highways. It highlights road safety via immediate notifications but is constrained by the necessity for significant infrastructure improvements.

An approach published in[3] in 2023, propose an IoT-driven technique along with a cloud system for identifying and notifying about potholes in real-time. Nonetheless, the method relies significantly on stable internet access and infrastructure backing. Finally,

An other approach published in[4] in 2024, investigates the use of UAVs and small machine learning methods for immediate road damage identification. Although this approach provides excellent precision and creative UAV application, it is prohibitively expensive and logistically difficult for widespread deployment.

A study[5] published in 2021, explores the combination of image processing methods with Google Maps to visualize and report road damage. Notwithstanding its inventive methodology, the research is limited by the quality of input images and the lack of sophisticated deep learning techniques.

A study[6] published in 2020, investigates utilizing Android smartphone sensors and machine learning methods to identify and categorize potholes. Although it emphasizes encouraging preliminary results, the dependence on user involvement and smartphone-based data collection restricts its feasibility for widespread implementation.

To sum up, the combination of these studies highlights the revolutionary capacity of IoT and deep learning methods in identifying road damage. Although the research shows considerable improvements in accuracy, efficiency, and workload reduction, it also emphasizes important issues like infrastructure needs, practical scalability, and cost-effectiveness. This literature review enhances comprehension of the present situation and offers significant insights for upcoming research and development in this area.

Table 2.1: Literature Review

Year	Paper	Methods	Limitations
2021	Smart Pothole Detection Using Deep Learning Based on Dilated Convolution [1]	This study employed dilated convolution-based deep learning techniques to enhance pothole detection accuracy. It explored image-based methods to identify road damage.	The approach is computationally intensive and struggles with generalizing to unseen road types, which limits its scalability.
2023	IoT-Based Highway Potholes and Bump Detection [2]	This paper proposed an IoT-based system integrating sensors for real-time detection of potholes and humps, focusing on improving highway safety through real-time alerts.	The system primarily focuses on highway contexts and requires significant infrastructure upgrades for practical deployment.
2023	IoT Based Pothole Detection System [3]	The research utilized IoT devices to detect and report potholes, leveraging a cloud-based reporting system to provide real-time data on road damage.	The deployment requires substantial infrastructure support and consistent internet connectivity for effective operation.
2024	Real-time road damage detection and infrastructure evaluation leveraging unmanned aerial vehicles and tiny machine learning [4]	This study employed unmanned aerial vehicles (UAVs) combined with tiny machine learning techniques for real-time road damage detection and infrastructure evaluation.	The approach relies on UAV technology, which can be cost-prohibitive and logistically challenging for large-scale implementations.

Continued on next page

Table 2.1: Literature Review (Continued)

Year	Paper	Methods	Limitations
2021	Monitoring of road damage detection systems using image processing methods and Google Maps [5]	This research used image processing techniques to detect road damage and integrated the results with Google Maps for better visualization and location-based reporting.	The approach is limited by image quality and lacks the benefits of deep learning techniques for enhanced detection accuracy.
2020	Automatic pothole classification and segmentation using Android smartphone sensors and camera images with machine learning techniques [6]	This study utilized Android smartphone sensors and camera images to classify and segment potholes. Machine learning techniques were applied, focusing on improving the accuracy and segmentation of road damage.	The method requires smartphone-based data acquisition, which depends heavily on user participation, potentially limiting widespread adoption.

Chapter 3

Project Vision

3.1 Problem Statement

The lack of an automated system for detecting road damage presents serious risks, such as maintenance delays, higher accident rates, and faster degradation of road infrastructure. Manual inspection methods are labor-intensive, prone to mistakes, and variable, resulting in ineffective resource distribution and increased maintenance expenses. Moreover, the absence of real-time reporting systems obstructs prompt responses from authorities, worsening safety risks for road users and adding to the overall deterioration of road networks.

3.2 Business Opportunity

The opportunity in business exists in offering a road damage detection and reporting solution powered by IoT, utilizing deep learning models and GPS integration to improve the efficiency and safety of road maintenance. The system delivers real-time damage identification, accurate location monitoring, and actionable insights via a cloud-based dashboard by providing a subscription service to municipal governments and infrastructure management firms. This creative solution tackles the issues of manual road inspections, allowing authorities to enhance maintenance processes, lower accident risks, and boost public contentment with road infrastructure management

3.3 Objectives

Creating an automated system for detecting road damage is crucial for transforming infrastructure management methods. The main goal is to combine lightweight deep learning models, like YOLOv5n, with IoT technologies to enable precise anomaly detection and instant reporting. The RoadRadar program seeks to boost the promptness and effectiveness of road upkeep, guaranteeing better road safety, lower accident probabilities, and extended infrastructure durability. Through accurate location tracking and automated reporting, the system intends to enhance maintenance processes and improve resource distribution for road authorities.

3.4 Project Scope

- **System Development:**
  - Creating and executing the IoT-based system for identifying road damage.
  - Incorporating lightweight deep learning models like YOLOv5n for detecting anomalies.
  - Creating a cloud-driven interface for immediate reporting and visual representation.
- **Road Damage Detection Algorithms:**
  - Designing and implementing advanced road damage detection algorithms.
  - Developing models to identify potholes, cracks, and various irregularities with great precision.
- **Performance Optimization:**
  - Optimizing the system for real-time processing of video data.
  - Efficient utilization of computational resources for IoT devices.
  - Ensuring scalability for larger road networks.

3.5 Constraints

3.5.1 Usability Constraints

The system should offer an easy-to-use and intuitive dashboard to guarantee acceptance by municipal officials and infrastructure managers. The dashboard must offer straightforward, applicable insights, requiring little training for users to effectively utilize its functionalities.

3.5.2 Performance Constraints

The system needs to handle real-time road damage information swiftly without any lags. The deep learning model needs to be lightweight to guarantee compatibility with IoT devices, all while achieving high accuracy. Moreover, the system must manage extensive road networks and significant data quantities while maintaining performance.

3.5.3 Compatibility Constraints

The solution should be compatible with multiple IoT devices and accommodate GPS modules for accurate location tracking. It must effortlessly blend with current road maintenance processes and cloud storage solutions to ensure smooth functionality.

3.5.4 User Adoption Constraints

The system should showcase evident advantages, including enhanced efficiency and lower maintenance expenses, to promote acceptance by authorities. Thorough training resources and technical assistance must be offered to help stakeholders in adapting to the new system.

3.6 Stakeholders Description

3.6.1 Municipal Authorities

Local governments are the main entities tasked with road upkeep and ensuring public safety. Their objectives encompass enhancing road standards, lowering upkeep expenses, and lessening accident hazards. Nonetheless, they encounter obstacles like constrained funding and dependence on manual evaluation methods.

3.6.2 Infrastructure Managers

Infrastructure managers monitor the state of the road network and distribute resources for maintenance. Their objective is to guarantee effective maintenance operations and enhance resource use while tackling issues like erroneous data and slow reporting.

3.6.3 Technical Teams

Technical teams are tasked with implementing and overseeing the IoT devices and deep learning models. Their objectives consist of making certain the system operates effectively and delivering continuous technical assistance. Challenges involve enhancing system performance and tackling hardware limitations.

3.6.4 Road Users

Road users gain from enhanced road quality and lower chances of accidents. Their objectives are to provide safer and more seamless transportation experiences. Challenges consist of possible setbacks in starting road repairs and the preliminary launch of the system.

3.6.5 Stakeholders Summary

The initiative includes various stakeholders, each with distinct roles and goals. Meeting their requirements, like effective road upkeep and precise anomaly identification, is vital for the project's success. The system seeks to add value for all stakeholders by improving road safety and streamlining maintenance processes.

3.6.6 Key High-Level Goals and Problems of Stakeholders

3.6.6.1 Municipal Authorities

City officials seek to improve road safety, lower maintenance expenses, and increase public contentment. They encounter obstacles like constrained finances and dependence on old inspection techniques.

3.6.6.2 Infrastructure Managers

Infrastructure managers strive to guarantee effective and prompt road upkeep while maximizing resource distribution. Their difficulties encompass erroneous or postponement in damage reports and the intricacy.

3.6.6.3 System Implementation and Optimization Teams

Technical teams strive to implement and support the system efficiently, guaranteeing its enduring dependability. Their difficulties consist of enhancing deep learning models for IoT devices and tackling hardware constraints.

3.6.6.4 Road Users

Road users, such as drivers, cyclists, and pedestrians, are the primary beneficiaries of the Road Radar system. Their main concern is to enjoy safer, smoother, and well-kept roads. By promptly identifying and reporting road surface issues—like potholes, cracks,

and other irregularities—this system seeks to reduce the likelihood of accidents, vehicle harm, and discomfort stemming from degrading infrastructure.

Nonetheless, in the early stages of deployment, road users might face indirect obstacles. These consist of:

Postponed repair actions, as local governments or road upkeep agencies adapt to incorporating live data from the system into their processes.

Momentary disturbances or adjustment problems as the system is tested, optimized, and refined in real traffic scenarios.

Issues with data accuracy arise, particularly when the system produces false positives or overlooks specific damage types during the initial phases of model training.

Regardless of these transitional obstacles, the lasting effects for road users are predominantly favorable. As the system develops and becomes better incorporated into public infrastructure upkeep procedures, users can anticipate quicker road repairs, less vehicle damage, and improved driving safety, especially in regions that were previously subject to inadequate care or postponed maintenance.

Chapter 4

Software Requirements Specifications

This chapter will have the functional and non functional requirements of the project.

4.1 List of Features

1. **Road Damage Detection:**

   deep learning techniques for the automatic detection of road surface issues, such as potholes and fissures.

2. **Real-Time Reporting:**

   - Deliver immediate reports of identified road damage with accurate GPS coordinates.

3. **Cloud-Based Dashboard:**

   - An online platform for displaying road damage information and producing reports for officials.

4. **Integration with IoT Devices:**

   - Smooth integration with IoT devices for immediate video data gathering and processing.

5. **Data Visualization:**

    - A mapbox displaying road damage details, highlighting the key areas that need repairs.

4.2 Functional Requirements

1. **Road Damage Detection:**

    - The the system must identify and classify road anomalies, such as potholes and cracks, using deep learning techniques.

2. **Real-Time Reporting:**

    - Provide immediate alerts and reports to maintenance personnel, outlining the location and extent of damage.

3. **Cloud-Based Dashboard:**

    - Allow users to view detailed road damage reports and monitor repair advancements through a web interface.

4. **Integration with IoT Devices:**

    - Enable IoT devices to collect video footage and transmit information for real-time analysis.

5. **Data Visualization:**

    Mapbox displaying road damage details, highlighting the key areas that need repairs.

4.3 Quality Attributes

1. **Reliability:**

   The system must reliably detect road damage and provide accurate results without mistakes or inconsistencies.

2. **Performance:**

   The system must process video data and generate reports immediately, even for large road networks, without significant delays.

3. **Security:**

   The system must ensure secure data transmission and storage, protecting confidential information from unauthorized access.

4. **Usability:**

   The online dashboard should be easy to navigate and simple, needing minimal effort for users to understand.

5. **Maintainability:**

   The system should be designed to enable easy updates and upgrades over time, supporting continuous development.

6. **Portability:**

   The solution should be compatible with multiple devices, including desktops, tablets, and smartphones.

4.4 Non-Functional Requirements

1. **Reliability:**

   The system must reliably deliver precise outcomes for road damage detection without errors.

2. **Performance:**

   It should process video data and generate reports immediately without significant delay.

3. **Security:**

   All information must be transmitted and stored securely to avoid unauthorized access or breaches.

4. **Usability:**

   The digital platform should be user-friendly and require minimal training for users to operate effectively.

5. **Maintainability:**

   The system needs to be modular and simple to maintain, facilitating the addition of updates and new functionalities.

4.5 Use Cases/ Use Case Diagram

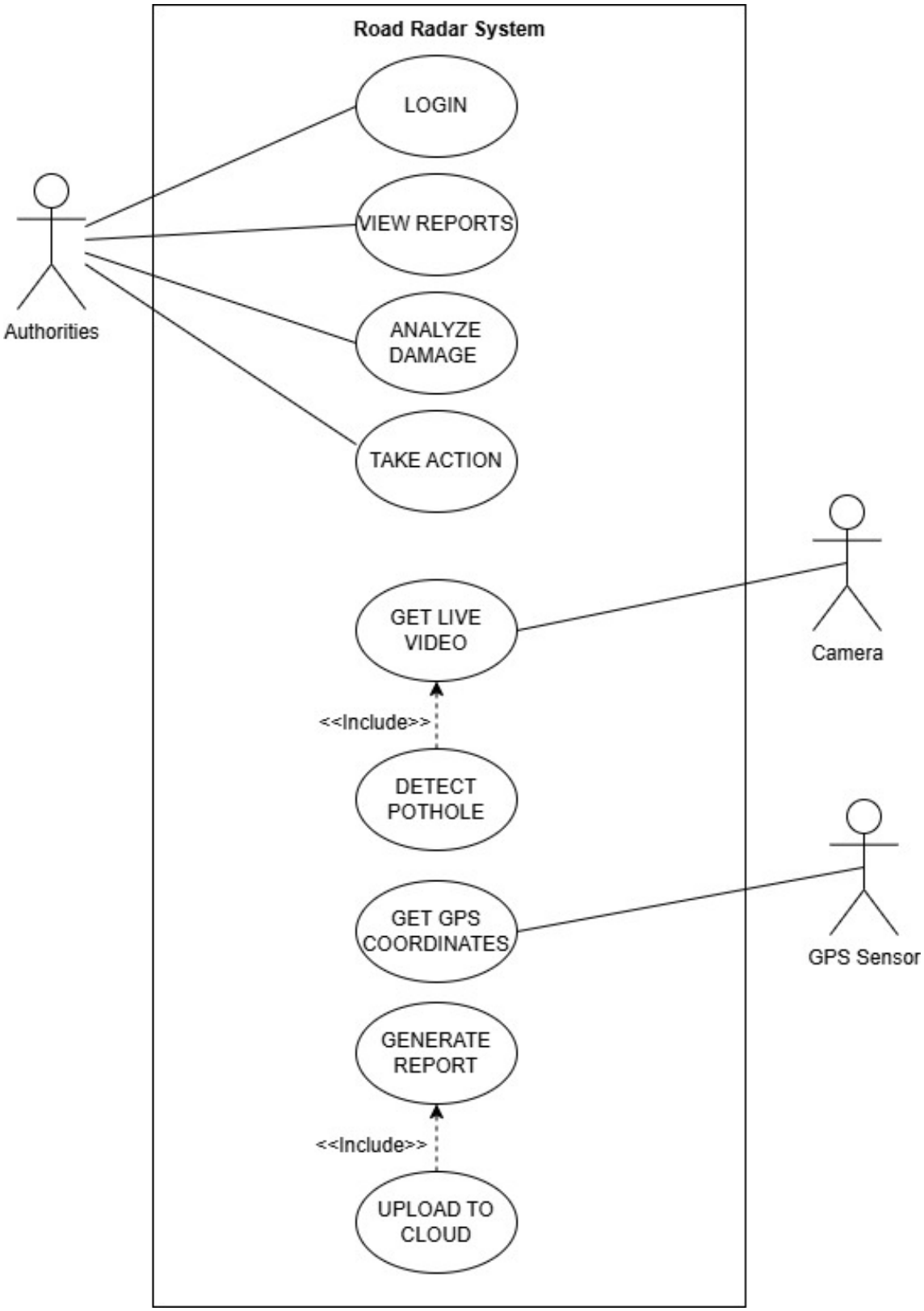

Figure 4.1: Use Case Diagram

4.5.1 Detect Road Damage

Description: The system detects road defects such as cracks and potholes utilizing deep learning algorithms.
Actors: IoT Devices, Municipal Authority
Preconditions: IoT devices are set up and sending video information.
Basic Flow:

1. Video recordings of roadways are captured by IoT devices.
2. The system analyzes the video to identify irregularities.
3. Detected road damage is categorized and recorded.

Postconditions: Damage data is gathered and formatted for reporting.

4.5.2 View Damage Reports

Description: Local officials can access comprehensive reports on identified road defects.
Actors: Municipal Authority
Preconditions: Damage information is accessible in the system.
Basic Flow:

1. The user accesses the web dashboard.
2. Reports include details on damage, such as the site and extent.
3. The user can download or filter the reports when necessary.

Postconditions: Reports are employed for maintenance scheduling.

4.5.3 Visualize Data on Map

Description: Damage to roads is represented on an interactive map to enhance decision-making.

Actors: Municipal Authority

Preconditions: Damage information is tagged with geographic coordinates and can be visualized.

Basic Flow:

1. The user opens the map display on the online dashboard.
2. Damage sites are shown as indicators.
3. Users can click on markers to access detailed information.

Postconditions: Map data is utilized to rank maintenance tasks.

4.6 Test Plan (Test Level, Testing Techniques)

4.6.1 Test Levels

The testing activities will be carried out at the following levels:

- Testing and Refinement
- Real-Time Response Testing
- Performance Optimization
- Final Testing

4.6.2 Testing Techniques

Different testing methods will be used throughout the various stages:

4.6.2.1 Real-Time Response Testing

- Evaluate the system's capability to identify and communicate road damage instantly through IoT devices. (e.g., Raspberry Pi).

- Verify the accuracy of anomaly detection and location tracking based on GPS.
- Simulate dynamic road conditions to evaluate system responsiveness and precision.

4.6.2.2 Performance Optimization

- Load Testing to evaluate system performance under increased data entry rates.
- Stress testing to confirm stability under extreme conditions, including extended continuous operation.
- Scalability Testing to evaluate system performance with expanded road networks or numerous IoT devices.
- Testing resource utilization to enhance processing efficiency on Raspberry Pi and cloud systems.

4.6.2.3 Final Testing

- System Integration Testing to guarantee seamless data transfer among IoT devices, cloud platforms, and the web interface.
- Practical Assessment by attaching the IoT device to a vehicle and observing the system's performance under real road conditions.
- Compatibility Testing to ensure the system operates smoothly on different devices and operating environments.
- Security assessments to safeguard data integrity and block unauthorized access to confidential information.

Chapter 5

Iteration Plan

5.1 FYP 1 Iteration Plan

- **Start of FYP 1:**
  - Timeline: 10 August
  - Description: Official commencement of the Final Year Project with initial discussions and planning.
- **Project Proposal:**
  - Timeline: 10 August to 31 August
  - Description: Create the project proposal, detailing the goals, range, methods, and anticipated results.
- **Project Proposal Defense:**
  - Timeline: 25 August to 08 September
  - Description: Submit the project proposal to the supervisor and committee for their approval and comments.

- **Literature Review and UML Diagrams:**
  - Timeline: 25 August to 31 October
  - Description: Perform a comprehensive analysis of current studies concerning IoT-enabled road damage identification, deep learning techniques, and real-time applications. Create UML diagrams for the architecture and workflow of the system.
- **Data Collection and Pre-Processing:**
  - Timeline: 10 September to 25 November
  - Description: Collect pertinent road damage datasets and prepare the data, involving annotation and normalization, for application in training and evaluating the model.
- **Front-End Development:**
  - Timeline: 25 September to 31 October
  - Description: Create and build the online dashboard for displaying road damage information, incorporating functionalities such as data visualization and reporting.
- **Model Training:**
  - Timeline: 30 October to 08 December
  - Description: Train the YOLOv5n model for detecting road damage with the preprocessed dataset and enhance the model for better accuracy and performance.

- **Final Presentation:**
  - Timeline: 30 October to 02 December
  - Description: Create and present the concluding presentation for FYP 1, highlighting the advancements, outcomes, and future strategies for the project.
- **Documentation:**
  - Timeline: 25 October to 26 November
  - Description: Gather all materials associated with the project, such as the literature review, UML diagrams, data preprocessing procedures, and system design, into a detailed project report.

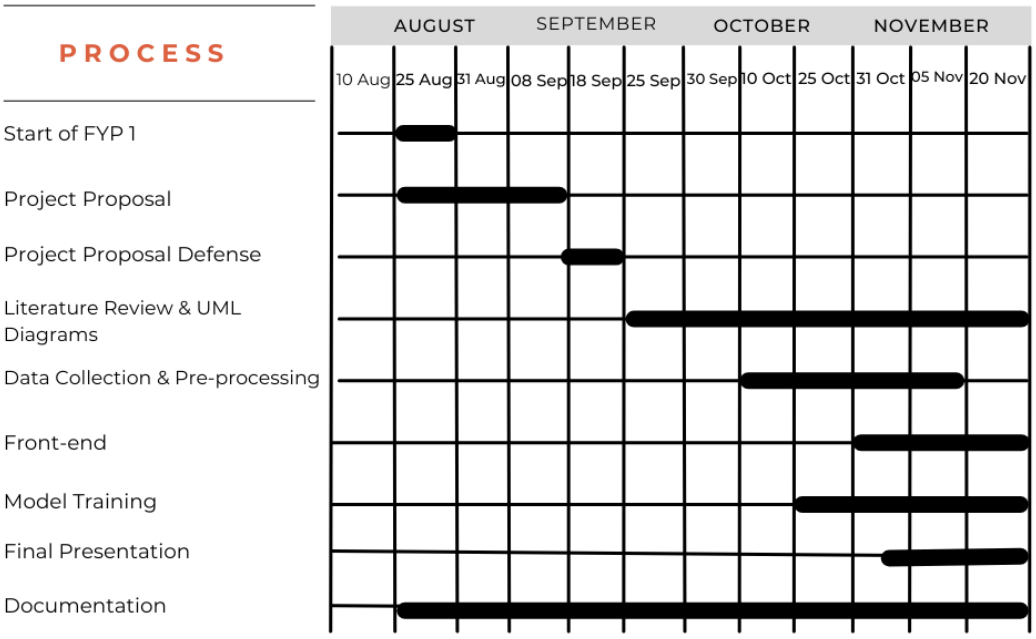

Figure 5.1: Gantt Chart

Chapter 6

Iteration 1

6.1 Architecture Diagram

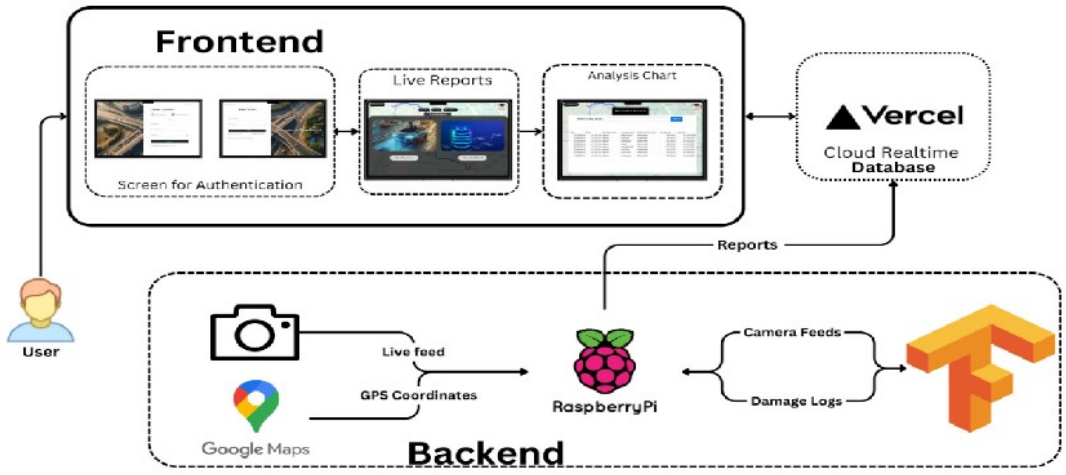

Figure 6.1: Architecture Diagram

6.2 Activity Diagram

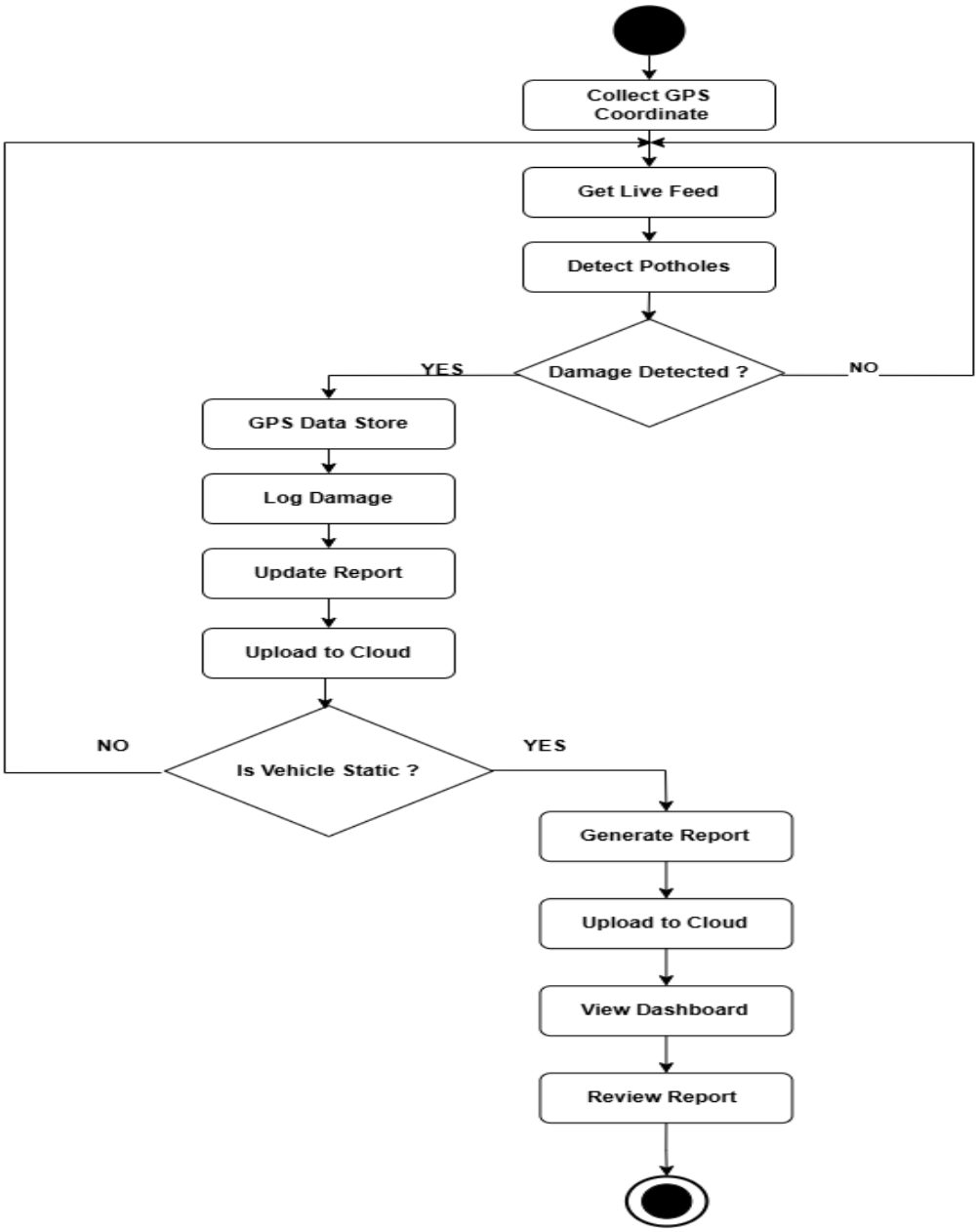

Figure 6.2: Activity Diagram

6.3 Sequence Diagram

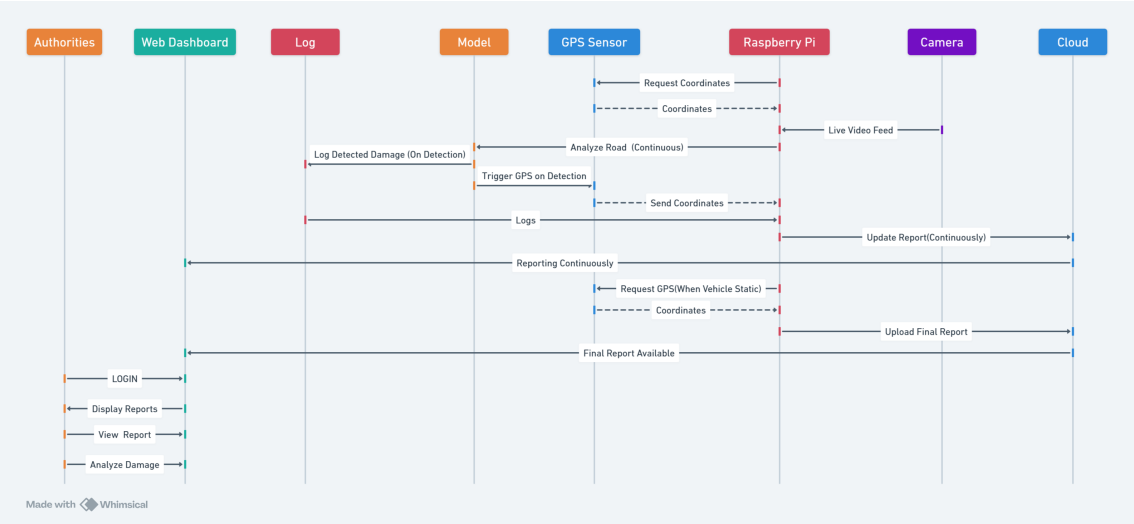

Figure 6.3: Sequence Diagram

6.4 Development Phase

6.4.1 Unit Test

Unit testing will include creating and running tests for specific components of the system to verify that each part operates correctly. This will involve creating test cases for the online dashboard, integrating IoT devices, and developing deep learning models for detecting road damage. Automated unit tests will be developed to swiftly verify code modifications and detect possible regressions in features like data visualization, GPS reporting, and anomaly detection.

6.4.2 Suites or Test Cases

To organize related unit and integration tests, test suites will be created. Every suite will confirm particular system interactions and functionalities. To provide thorough coverage of the system's functionality, test cases will cover a range of scenarios, including edge

cases like low-light video detection or uneven road surfaces, in addition to conventional usage cases.

6.5 Maintainable Phase

6.5.1 CI/CD

Testing and deployment procedures will be automated through the use of Continuous Integration (CI) and Continuous Deployment (CD) techniques. With an emphasis on the online dashboard, IoT data flow, and deep learning integration, continuous integration (CI) will entail automatically constructing and testing the system anytime code changes are made. The deployment of the tested code and models to the cloud environment will be automated by CD, guaranteeing the effective and dependable delivery of system upgrades.

6.5.2 System-Level Test Suites and Test Cases

To verify that every component of the system functions as intended, system-level test suites will be created. These tests will guarantee smooth communication between the online dashboard, cloud-based processing, and IoT devices. End-to-end situations including dashboard presentation, GPS-based damage localization, and real-time video data recording will all be covered in the test cases. The system's resilience will be assessed by performance and stress testing, particularly in situations involving high data input rates or prolonged operation times.

Chapter 7

Iteration 2

7.0.1 Preprocessing, Feature Extraction, and Frontend Development

With an emphasis on data preparation, feature extraction, and the preliminary creation of the web interface, significant progress was made in improving important aspects of the "Road Radar" project during the second iteration of FYP-I. This phase's primary objectives were to turn unprocessed video and sensor data into useful inputs, find important characteristics like the kinds and severity of cracks, and establish a user-friendly online site for log visualization and real-time monitoring.

7.0.1.1 Data Preprocessing and Feature Engineering

Significant progress was made in improving the "Road Radar" project's essential elements during the second iteration of FYP-I, with a particular emphasis on feature engineering, data preprocessing, and the preliminary creation of the web interface. Establishing a fundamental web-based platform and guaranteeing the integrity of the dataset used for training and analysis were the objectives of this phase.

Data preparation was essential in this stage to ensure accuracy and consistency. Road surface video streams and associated sensor readings (such as GPS and gyroscope data) were among the raw input data that was cleaned and organized. In order to deal with missing frames, remove noise that can impair the model's performance, and eliminate

inconsistencies, this step was essential. To match the setup used for model training, video frames were taken out and scaled to 512x512, a typical resolution.

After preprocessing, the visual and sensory data were subjected to feature engineering approaches in order to extract useful information. Features including crack types (longitudinal, transverse, alligator, and potholes), detection confidence levels, and GPS positions were found and recorded using a refined YOLOv5n model. To improve the dataset and add context, other patterns including crack distribution, frequency, and spatial mapping were taken into consideration.

This phase's objective was to convert unstructured road data into intelligent, organized information that enables the detection system to produce precise, real-time forecasts. In order to create an intelligent, data-driven road inspection tool, this organized output forms the foundation for log production, on-screen annotations, and web-based reporting.

7.0.1.2 Web Pages Development of the Website

The creation of an intuitive user interface for the Road Radar system on the website's main page is part of this iteration. Viewing detection logs, tracking vehicle movement using GPS coordinates, and keeping an eye on road conditions in real time are all features of the website. The homepage's main features will be the ability to register and log in, a dashboard showing the results of road fault detection, and visual maps showing regions that frequently have potholes or cracks. To ensure a smooth and educational user experience, the interface will also display summaries of discovered fracture kinds, detection probabilities, and GPS-based tracking.

The website will also allow users to access real-time road monitoring data and modify alert settings. To guarantee that the platform works on a range of devices, such as desktop PCs, tablets, and smartphones, special attention will be paid to responsive design. The top page will be the primary location for communicating with the Road Radar system and keeping an eye on its operations; it will include summaries of detected road problems, recent GPS data, and real-time detection updates.

7.0.1.3 System Architecture and Diagrams

In Iteration 2, The completion of the system architecture and the development of crucial diagrams that outline the general layout and functionality of the Road Radar system will be major priorities in Iteration 2. The YOLO-based object detection model, GPS and gyroscopic sensors, data logging techniques, Firebase cloud storage, and the interactive website dashboard are only a few of the essential components that will be integrated in the system design. It will demonstrate how real-time data is gathered from the Raspberry Pi, processed by the model, and sent to the cloud and local storage for analysis and monitoring.

To provide a clear knowledge of system behavior and interactions, further diagrams will be created, including use case diagrams, system sequence diagrams, and class diagrams. The use case graphics will show how various users engage with the platform, including how to see reports, log GPS, and detect in real time. System sequence diagrams will illustrate how sensors, the detection model, data handling scripts, and Firebase interact step-by-step. Last but not least, class diagrams will show the software's logical organization and the connections between the modules in charge of interface management, logging, and video processing.

The logging architecture, system integration, and data processing pipelines will be firmly established by the conclusion of Iteration 2, providing a solid basis for real-time testing on the Raspberry Pi. In the following stages, the Road Radar project's ongoing development and improvement will be guided by these diagrams and the entire architecture.

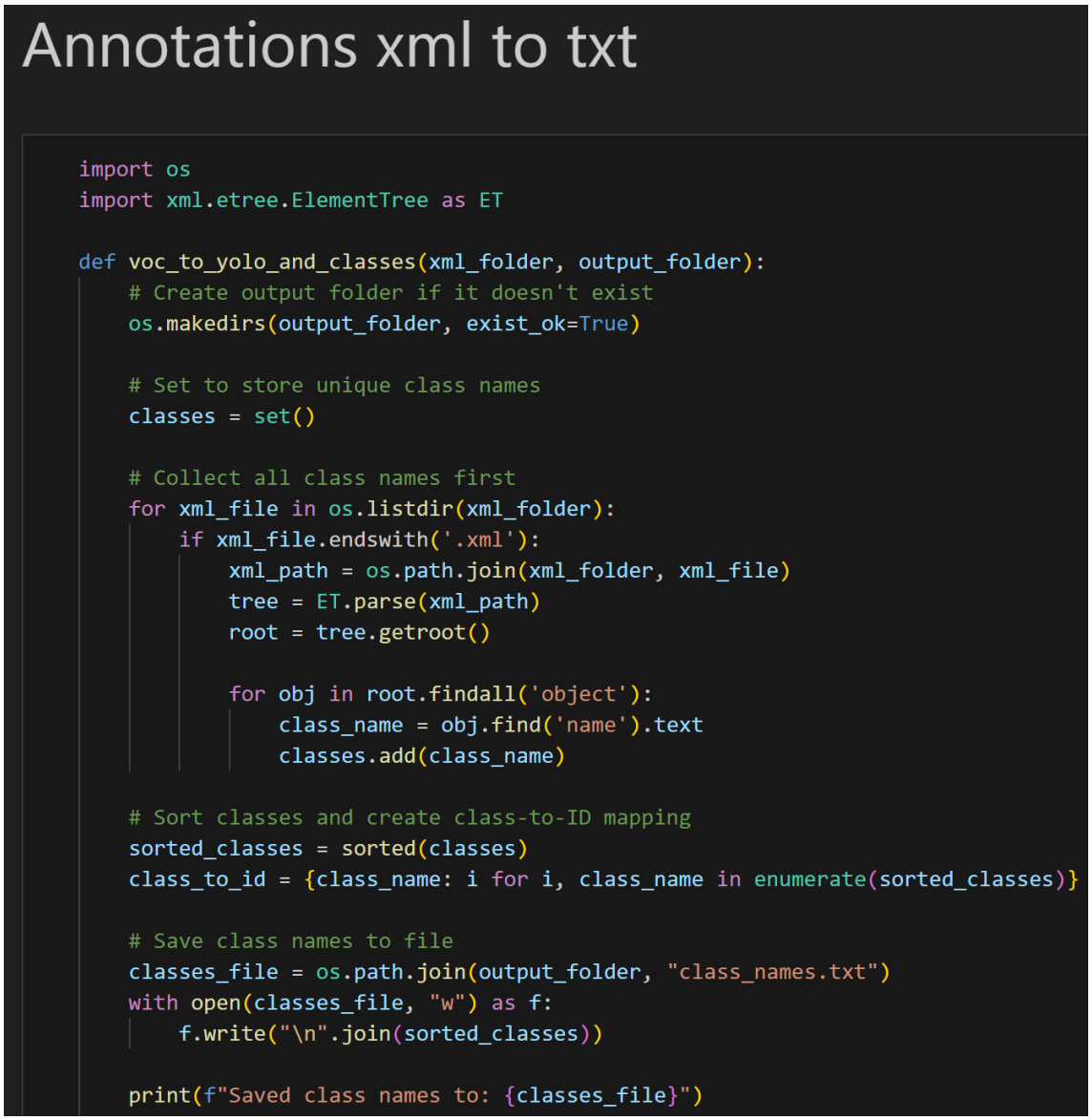

Figure 7.1: Dataset & Preprocessing

Class Mapping

```python
import os

# Path to your YOLO annotation files
annotation_folder = "/mnt/f/final2/China_MotorBike/China_MotorBike/train/labels"

# Updated class mapping
class_mapping = {
    "D00": "0",
    "D01": "5",
    "D0w0": "6",
    "D10": "1",
    "D11": "4",
    "D20": "2",
    "D40": "3",
    "D43": "7",
    "D44": "8",
    "D50": "9"
}

# Function to update YOLO annotations
def update_class_names(annotation_folder, class_mapping):

    if not os.path.exists(annotation_folder):
        print(f"Annotation folder {annotation_folder} does not exist.")
        return

    for file_name in os.listdir(annotation_folder):
        if file_name.endswith(".txt"):
            file_path = os.path.join(annotation_folder, file_name)

            with open(file_path, "r") as file:
                lines = file.readlines()

            updated_lines = []
            for line in lines:
                parts = line.strip().split()
                if parts:  # Ensure line is not empty
```

Figure 7.2: Class Mapping

Data Spliting (Train-Val-Test)

```python
import os
import shutil
import random

# Define source directories
SOURCE_IMAGE_DIR = "/mnt/f/final2/China_MotorBike/China_MotorBike/train/images"
SOURCE_LABEL_DIR = "/mnt/f/final2/China_MotorBike/China_MotorBike/train/labels"

# Define base output directory
OUTPUT_DIR = "/mnt/f/final2/DATASET"

# Define train, val, and test subdirectories dynamically
SPLITS = ["train", "val", "test"]
for split in SPLITS:
    os.makedirs(os.path.join(OUTPUT_DIR, split, "images"), exist_ok=True)
    os.makedirs(os.path.join(OUTPUT_DIR, split, "labels"), exist_ok=True)

# Get all image files (assuming .jpg and .png images)
image_files = [f for f in os.listdir(SOURCE_IMAGE_DIR) if f.endswith((".jpg", ".png"))]
random.shuffle(image_files)  # Shuffle to ensure random distribution

# Calculate split indices
total_images = len(image_files)
train_split = int(0.7 * total_images)
val_split = int(0.2 * total_images)

train_files = image_files[:train_split]
val_files = image_files[train_split:train_split + val_split]
test_files = image_files[train_split + val_split:]

# Function to move images and corresponding labels
def move_files(files, src_img_dir, src_label_dir, dest_img_dir, dest_label_dir):
    for file in files:
        # Move image
        shutil.move(os.path.join(src_img_dir, file), os.path.join(dest_img_dir, file))
```

Figure 7.3: Data Split

Chapter 8

Iteration 3

In Iteration 3, The lightweight detection model was successfully deployed and tested on actual video inputs in Iteration 3, marking a significant advancement for the Road Radar project. Approximately 80 percent of the system's development was finished during this period, which also saw significant advancements in data logging, interface integration, and real-time detection performance. After being trained on photos of road damage, such as cracks and potholes, the model showed good performance in video inference, generating both structured logs and visual outputs.

The improvement of the user interface and overall system experience was the main focus of this iteration. The completed Firebase integration made it possible to save and retrieve real-time logs via the dashboard, including crack kind, detection confidence, and GPS locations. Additionally, batching data logs improved system performance and efficiency for real-time use on devices with limited resources, such as the Raspberry Pi 4.

8.1 Model Training and Initial Results

The completion of the road crack detecting system's model training represents a noteworthy turning point in this iteration. A preprocessed dataset of road crack photos that have been tagged for several kinds of cracks, such as longitudinal, transverse, alligator, and pothole cracks, is used to train the model. In order to identify dangerous road conditions,

the model aims to precisely categorize these cracks and offer real-time detection in road video streams.

By learning from annotated photos of road fractures, the model modifies its parameters during training to reduce errors and increase prediction accuracy. The YOLOv5n model, which is renowned for its speed and efficiency, was chosen because it can operate on devices with limited resources, such as the Raspberry Pi 4. Various model configurations are assessed, and hyperparameters are adjusted to get optimal performance in identifying different kinds of cracks.

Following training, a different testing dataset is used to validate the model's performance and evaluate its practicality. The model's efficacy in crack detection is assessed using key performance metrics like accuracy, precision, recall, and F1 score. Furthermore, real-time testing on video feeds guarantees that the system can quickly detect and categorize cracks while recording gyro sensor and GPS coordinate data for additional examination. After that, these outcomes are shown on a web interface, giving users immediate feedback on how well the system is working.

Although there is still work to improve and optimize the system, preliminary results show that the model can identify cracks in a variety of road situations. This entails enhancing accuracy for various fracture kinds and modifying detection thresholds. These first findings will direct the subsequent stage of enhancements, guaranteeing that the model grows more resilient and dependable for use in practical situations.

8.2 Website Completion and User Interface Refinement

In Iteration 3, the website will be almost finished with notable enhancements to both functionality and design. By this point, the user interface will be completely functional, enabling users to track the road radar system's status and view real-time crack detection results. A dashboard on the website will provide the GPS coordinates and currently detected crack kinds, which are updated constantly as the system processes video frames. To guarantee safe access and appropriate data management, user authentication and performance monitoring will also be put into place.

Making the user interface more responsive and intuitive is a major component of this iteration. Desktop computers, tablets, and smartphones are just a few of the devices that will be able to easily access the website thanks to its responsive design. Through the interface, users will be able to view detection results, examine logs, and modify parameters including sensor settings and detection thresholds.

Real-time road condition information, such as the kinds of cracks found, confidence levels, and a map displaying the GPS positions of found cracks, will be shown on the front-end. The system would also enable users to examine comprehensive reports on road conditions, track vehicle itineraries, and visualize crack detection history. Based on the system's outputs, users will be able to make well-informed judgments regarding road upkeep or repairs.

The website will be completely operational and user-friendly by the conclusion of Iteration 3, offering a dependable platform for tracking and communicating with the Road Radar system. Simplicity will be prioritized, making sure that users can navigate the interface with ease and without the need for technical knowledge.

8.3 Model Trained on Road Condition Data

Iteration 3 will concentrate on combining sensor data with the picture data used for crack detection in order to improve the model's capacity to precisely identify and categorize road cracks. The model will include real-time data from GPS and gyro sensors, which offer useful context for identifying road conditions, in addition to visual data from road cameras. The model will be better equipped to evaluate the extent of cracks and ascertain their possible influence on road safety by incorporating this sensor data.

To enable the model to understand patterns related to crack formation and the dynamic behavior of roads over time, the system will be trained using a combination of sensor data and annotated road photographs. The correlation between sensor readings, road conditions, and crack kinds will be examined using machine learning methods. For example, regions with high traffic or frequent temperature changes may be areas where the road is deteriorating more quickly, as indicated by strong crack detections combined with atypi-

cal gyro readings.

Furthermore, the model will learn from the temporal data, comprehending the evolution of fractures and the roles played by particular road kinds or places. The device will be able to map identified cracks to precise locations by integrating GPS data, producing an extensive road condition log that can be utilized for planning future maintenance and repairs.

The system will improve its decision-making process for identifying and categorizing fractures by gaining a richer understanding of road conditions through the integration of sensor data with visual inputs. The influence of this integration on the detection system's accuracy and dependability will be assessed, especially in difficult road situations with fluctuating conditions.

8.4 80% Development Completion

Around 80 percent of the Road Radar system's development will be finished at this point. This implies that all of the system's essential parts—such as the web interface, real-time data logging, sensor integration, and fracture detection model—will function. GPS integration with the fracture detection model will yield comprehensive information about road conditions. The technology will be able to log pertinent sensor data, identify different kinds of road cracks, and show real-time updates on the internet.

Based on user input, the remaining 20 percent of the effort will be devoted to system improvement, performance optimization, and the development of sophisticated features. The accuracy of the crack detection model will be improved, sensor data processing will be improved, and the website's performance will be improved for improved user interaction. The website's interface will also be enhanced for a better user experience, making it simple to use and straightforward for tracking road conditions.

To make sure the system can manage massive volumes of video and sensor data in real-time, stress testing will be done. Debugging and fixing any problems that may come up during testing, especially in various road situations and conditions, will also be part

of this step. To guarantee scalability for upcoming improvements, like the inclusion of more sensors or better detection algorithms, the system architecture will be examined and modified.

The Road Radar system will be completely functional by the end of this phase, able to track vehicle routes, identify road cracks, and provide real-time feedback via the website. Extensive testing, system completion, and being ready for deployment in real-world situations will become the main priorities.

8.5 Key Focus Areas in Iteration 3

The completion of the website, improvement of the crack detection model, and integration of all system components into a fully functional solution will be the main goals of Iteration 3. Making sure the model accurately detects different kinds of cracks using both historical and real-time video data will be one of the primary objectives. For more context-aware insights, this involves optimizing the model's capacity to accurately recognize cracks and link them to GPS and gyro sensor data.

In order for users to engage with the system, see historical road condition logs, and track detection results in real-time, the website must have an intuitive, user-friendly interface. One of the main goals of this iteration will be to make sure that non-technical users can easily access and understand system outputs.

To reach the 80% development milestone, system scalability and stability will be key priorities in addition to these functionalities. The group will make sure that the design of the system can support several concurrent user requests, fast sensor updates, and high-resolution video feeds without experiencing any performance issues. In order to enable real-time detection and data processing in resource-constrained contexts, optimization efforts will also be made to guarantee that the model operates well, particularly when installed on a Raspberry Pi 4.

During this phase, user and stakeholder feedback will be actively gathered to guide final adjustments and inform system refinements. With an emphasis on deployment, real-world

testing, and overall system readiness, these enhancements will aid in getting the Road Radar system ready for the last stage.

```
# Import settings
from ultralytics import settings

# Update paths
settings.update({
    "datasets_dir": "/mnt/f/final2/DATASET/data.yaml",
    "runs_dir": "/mnt/f/final2/runs",
    "weights_dir": "/mnt/f/final2/weights"
})

# Verify changes
print("Datasets Path:", settings["datasets_dir"])
print("Runs Path:", settings["runs_dir"])
print("Weights Path:", settings["weights_dir"])
```

```
Datasets Path: /mnt/f/final2/DATASET/data.yaml
Runs Path: /mnt/f/final2/runs
Weights Path: /mnt/f/final2/weights
```

Figure 8.1: Installing Dependencies

```python
import os
from ultralytics import YOLO

data_yaml = settings["datasets_dir"]
epochs = 150
batch_size = 40
img_size = 512
model_save_dir = "/mnt/f/final2/best_model"


os.makedirs(model_save_dir, exist_ok=True)


model = YOLO("yolov5n.pt")


model.train(
    data=data_yaml,
    epochs=epochs,
    batch=batch_size,
    imgsz=img_size,
    project=model_save_dir,
    name="yolov5n_finetuned",
    save=True,
    exist_ok=True
)

# Validation
model.val(
    data=data_yaml,
    imgsz=img_size
)

# Testing
model.val(
    data=data_yaml,
    split="test",
    imgsz=img_size
)


best_model_path = os.path.join(model_save_dir, "yolov5n_finetuned", "weights", "best.pt")
if os.path.exists(best_model_path):
    print(f" YOLOv5n fine-tuned model saved at: {best_model_path}")
else:
    print("Best model not found!")
```

Figure 8.2: Model Training(a)

```
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
train: Scanning /mnt/f/final2/DATASET/train/labels.cache... 7759 images, 32 backgrounds, 0 corrupt: 100%|██████████| 7759/7759 [00:00<?, ?it/s]
val: Scanning /mnt/f/final2/DATASET/val/labels.cache... 2217 images, 7 backgrounds, 0 corrupt: 100%|██████████| 2217/2217 [00:00<?, ?it/s]
Plotting labels to /mnt/f/final2/best_model/yolov5n_finetuned/labels.jpg...
optimizer: 'optimizer=auto' found, ignoring 'lr0=0.01' and 'momentum=0.937' and determining best 'optimizer', 'lr0' and 'momentum' automatically...
optimizer: SGD(lr=0.01, momentum=0.9) with parameter groups 69 weight(decay=0.0), 76 weight(decay=0.000625), 75 bias(decay=0.0)
TensorBoard: model graph visualization added ✅
Image sizes 512 train, 512 val
Using 8 dataloader workers
Logging results to /mnt/f/final2/best_model/yolov5n_finetuned
Starting training for 150 epochs...

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      1/150      3.14G      2.301       3.78      1.996        146        512: 100%|██████████| 194/194 [00:51<00:00,  3.76it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 28/28 [00:13<00:00,  2.02it/s]
                   all       2217       5319       0.18      0.204      0.104     0.0387

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      2/150      3.63G      2.127       2.92      1.766        154        512: 100%|██████████| 194/194 [00:50<00:00,  3.82it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 28/28 [00:19<00:00,  1.42it/s]
                   all       2217       5319      0.352      0.184     0.0981     0.0338

      Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
      3/150      3.63G      2.162       2.768      1.804        126        512: 100%|██████████| 194/194 [00:58<00:00,  3.33it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 28/28 [00:21<00:00,  1.30it/s]
                   all       2217       5319      0.249      0.186       0.12     0.0426
```

Figure 8.3: Model Training(b)

Confusion Matrix Normalized

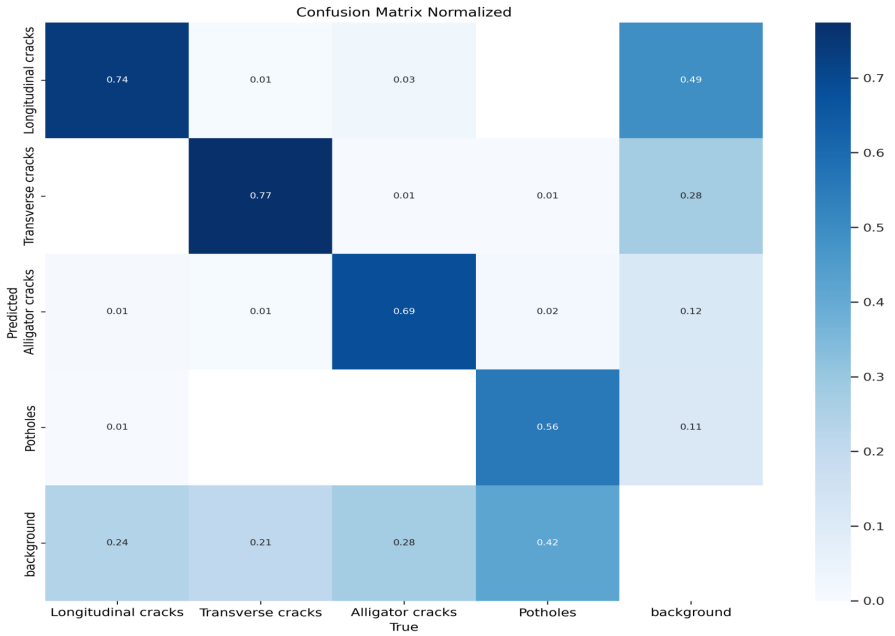

Figure 8.4: Results

Chapter 9

Iteration 4

During this stage, user and stakeholder input will be actively gathered to guide system improvements. The overall functionality of the website will be finished and made fully functional during this iteration. The web interface will be the main way that users communicate with the Road Radar system. Users can examine real-time logs of crack detections, including the type of crack, confidence score, and GPS coordinates of each detection, through an easy-to-use dashboard that will be provided by the front-end.

Data flow from video input to log generation will be seamless because to the close integration between the website's backend and the crack detection algorithm. The website will immediately record and display the detection results as the model processes frames. These logs will give users the crucial information they need to effectively monitor and assess road deterioration.

As soon as a new crack is discovered, the interface is updated to alert users of important detections and guarantee that all pertinent information is available instantly. Stakeholders in charge of road maintenance and safety may make prompt, well-informed decisions because to the model's close connection with the online interface.

When finished, the model will work autonomously with the website, continuously examining traffic video and logging findings without requiring human involvement. This iteration will feature thorough documentation in addition to finishing development and integration. In order to provide a strong basis for future scaling and deployment, this will

address the system architecture, backend logic, and the processes by which the interface updates its logs in real time.

This iteration will also prioritize polishing the user interface design and completing the integration processes. To provide a reliable and maintainable solution, the system will adhere to accepted technical standards and development criteria. The entire system architecture, integration process, and interface behavior will all be covered in the thorough documentation that will be created. This documentation will be used as a guide for potential feature additions and future maintenance.

When this iteration is finished and the model and website function as a completely integrated solution, the Road Radar system will have been successfully deployed. Road condition records will be available for users to view and examine in real-time; each entry will show the GPS coordinates of the detection, the type of crack, and the confidence score. Stakeholders may readily monitor road conditions without requiring extensive technical knowledge thanks to this simplified interaction.

Simple access will be provided by the web interface via the home page's Login and Register options. Existing users can safely log in with their credentials, while new users can create an account by providing their name, email address, and password. Users can access all detection records and engage with the platform's capabilities on the dashboard that appears after login in. All users involved in road condition monitoring will have a seamless and easy-to-use experience thanks to the interface's clear and straightforward design.

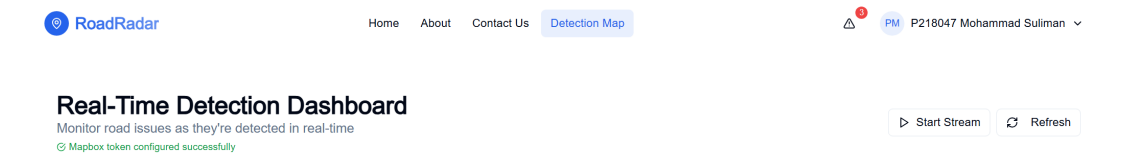

Figure 9.1: Website Bars

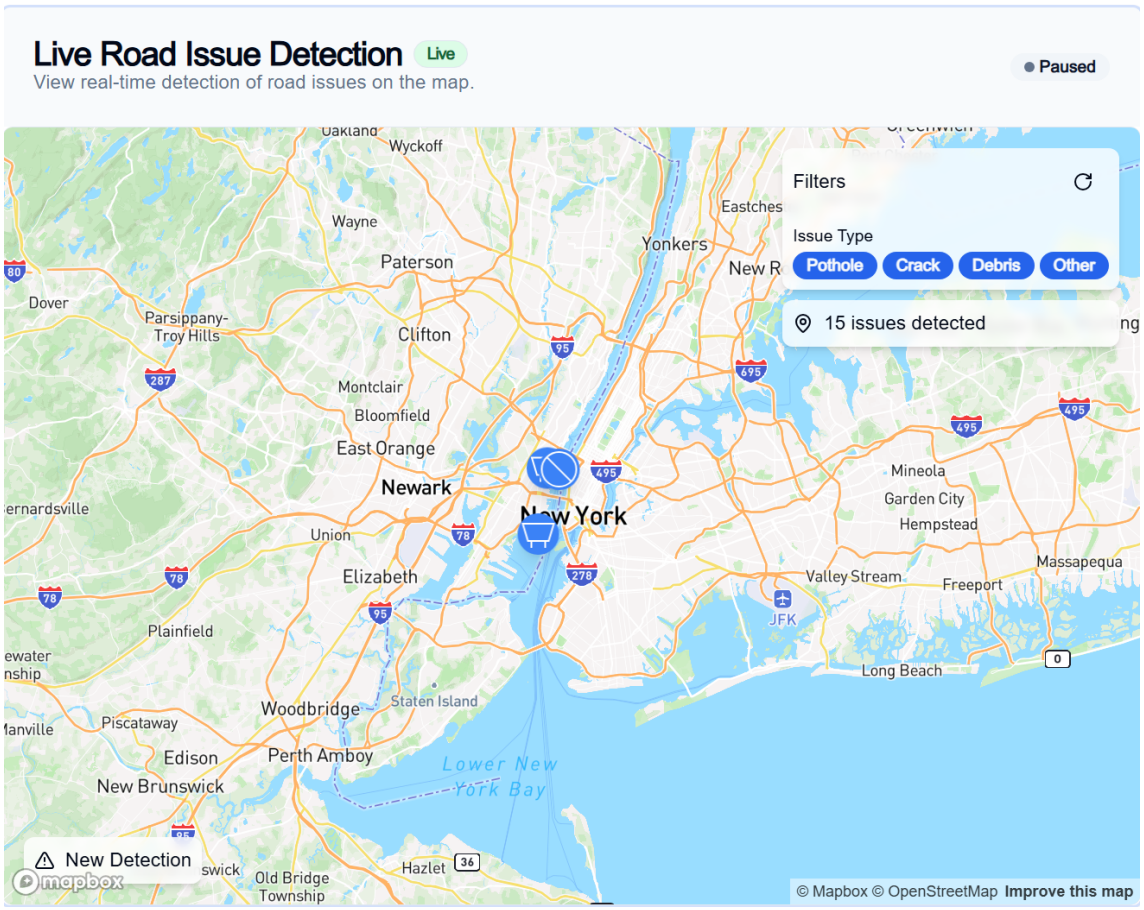

Figure 9.2: Mapbox

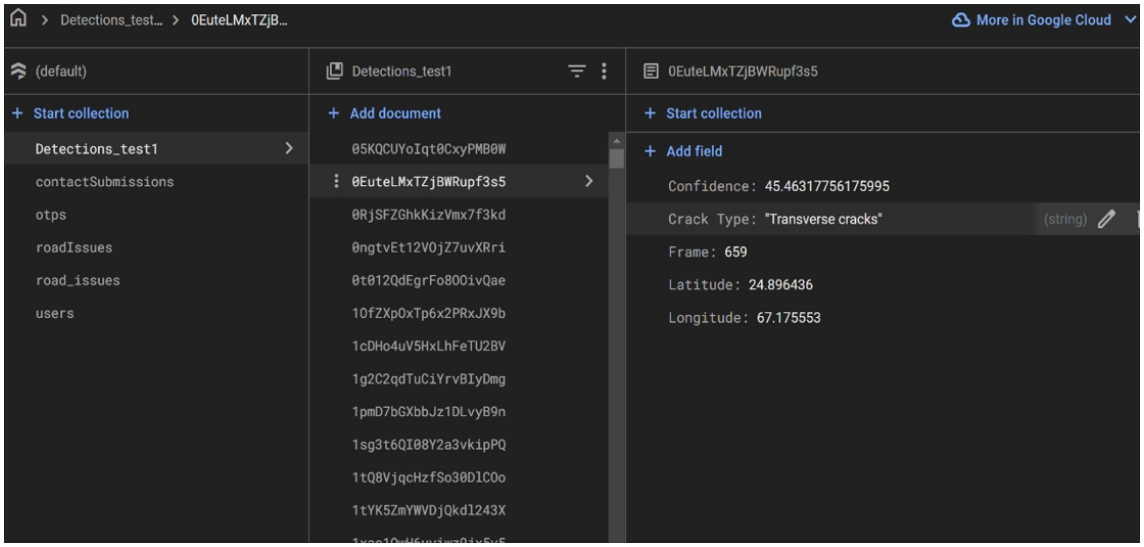

Figure 9.3: Logs

Chapter 10

Implementation Details

10.1 Implementation Details

10.1.1 Frontend

The Road Radar website's front end was created with React.js and Next.js, which provide a cutting-edge and effective foundation for creating quick, responsive web apps. Because the design is totally responsive, it works on a variety of platforms, such as smartphones, tablets, and PCs.

Users can interact with the system with ease thanks to the user interface's clear and simple design. Users can view the dashboard, which shows real-time detection logs, by logging in or registering from the homepage. These records provide rapid access to vital information by include specifics like the kind of crack, confidence score, and GPS locations for every detection.

Modular, maintainable code is made possible by the usage of React components, and server-side rendering and optimal speed are made possible by Next.js, which is particularly helpful for real-time changes and quicker page loads. A seamless and effective user experience across the platform is guaranteed by this configuration.

10.1.2 Backend

Firebase is used for real-time data storage and communication, and Python is used for the core detection logic in the Road Radar system's backend. The crack identification model recognizes various forms of road surface damage by processing video frames in real time using a lightweight deep learning framework (such as YOLOv5n).

Key data, such as the crack type, confidence score, and GPS coordinates, are retrieved and sent right away to Firebase, which serves as the main real-time database, as soon as a crack is discovered. This eliminates the requirement for conventional server-side frameworks like Flask and enables detection logs to be updated instantaneously and made accessible to the web interface.

Users will always view the most recent detection results as they are generated thanks to Firebase's ability to facilitate smooth synchronization between the frontend and backend. Additionally, it streamlines data management and user authentication, increasing system scalability and maintenance ease.

The detection system, GPS module, and web interface all communicate effectively thanks to this real-time backend configuration, which is tailored for operating on hardware with limited resources, like the Raspberry Pi 4.

10.1.3 Database

All detection-related data is stored and managed by the Road Radar project using a Firebase Realtime Database. Real-time data synchronization between the frontend user interface and the backend detection system is made possible using Firebase's cloud-based NoSQL database.

Every time a crack is found, the Firebase database is immediately updated with pertinent data, including the crack type, confidence score, and GPS coordinates. This guarantees that users can immediately examine the most recent detection logs on the web interface.

By doing away with the need for intricate SQL queries or ORM layers, Firebase further streamlines data processing. It is the perfect option for real-time apps like Road Radar

because of its smooth frontend integration, which enables effective data retrieval, user authentication, and scalability.

10.1.4 Security

Because the Road Radar system handles sensitive user data and real-time position information, security is an essential component. Only authorized users may access detection data and system functions thanks to the platform's usage of Firebase Authentication for secure user login and registration management.

All important information is sent over HTTPS, which creates a secure channel of communication between the client and Firebase. This includes GPS-based detection logs and user credentials. Furthermore, Firebase provides an additional degree of security against unwanted access by automatically encrypting data while it is in transit and at rest.

Firebase prohibits any direct interaction with backend processes that would jeopardize security, and access control rules make sure that users can only access their own data. A safe and legal environment is maintained through periodic evaluations of Firebase security setups and regulations.

All of these safeguards work together to keep Road Radar safe, reliable, and resistant to common flaws.

10.1.5 User Experience

The Road Radar platform's user interface is intended to be seamless, easy to use, and very accessible. Users may easily traverse the system and concentrate on the most important information—road surface conditions—thanks to the simple and clear interface.

A simple dashboard that shows real-time detection data, including the crack type, confidence score, and GPS positions, is shown to users. Without any technical knowledge, consumers can easily understand the detection findings thanks to the information's straightforward arrangement.

Whether using a PC, tablet, or smartphone, customers may easily access the platform

thanks to the design's adaptability across a variety of platforms. Although there aren't any intricate sensor graphs or graphic outputs, the focus on important information and clarity guarantees a productive and distraction-free experience. The platform should be usable by road safety inspectors, local authorities, and regular users who require fast information about road conditions.

Chapter 11

User Manual

11.1 Home Screen

The Road Radar web interface's Home Screen gives users quick access to all of the platform's essential functions. Users can go from this page to:

Login: To check detection logs and access their dashboard, current users can safely log in.

Register: By entering their name, email address, and password, new users can create an account.

Dashboard Access: Users are taken to the dashboard, which shows real-time road damage detection logs, after logging in.

In order to ensure accessibility on a variety of platforms, including PCs, tablets, and smartphones, the layout is made to be clear, responsive, and intuitive. The home screen is the primary gateway to the Road Radar system, regardless of whether a user is returning or visiting for the first time.

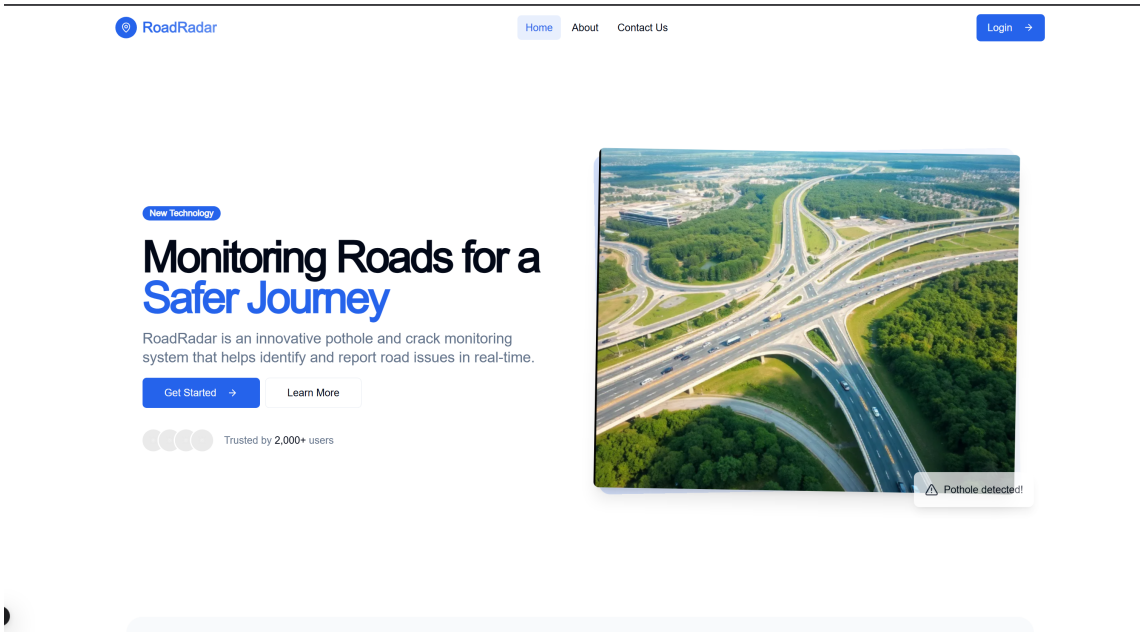

Figure 11.1: Home Screen

11.2 Login

Registered users can safely access their Road Radar account by using the Login option. To access the system, users must enter their email address and password on the login page.

Firebase Authentication powers the login process, handling credential verification over secure, encrypted protocols to protect user data. Since all data is sent over HTTPS, it is shielded from possible interception while in transit.

Unauthorized access is less likely thanks to Firebase's integrated security features, which include session management and real-time authentication monitoring. Although multi-factor authentication isn't used at the moment, the system adheres to data privacy and access control best practices to guarantee user accounts stay safe.

Following a successful login, users are taken to the dashboard, where they can engage with the system and check detection records.

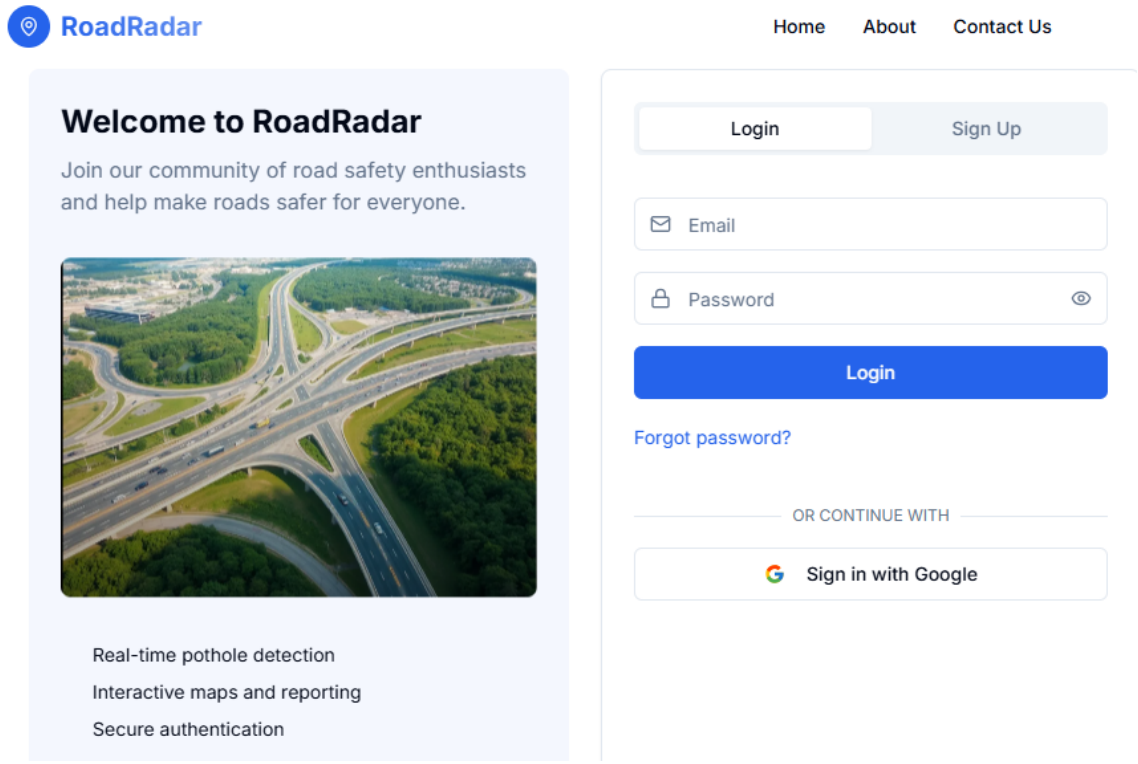

Figure 11.2: Login

11.3 Sigup

By filling out the registration form with their name, email address, and password, new users can register for an account on the Road Radar platform. The system uses Firebase Authentication to start the registration process after submission.

A verification email is automatically sent to the registered email address in order to guarantee account legitimacy and stop unwanted access. Before being allowed access to the system, users must click the offered link to validate their email address.

After verification, users can log in and use the entire website, including the information sections and detection dashboard. Only authorized users will be able to safely access and utilize the platform thanks to this registration and verification procedure.

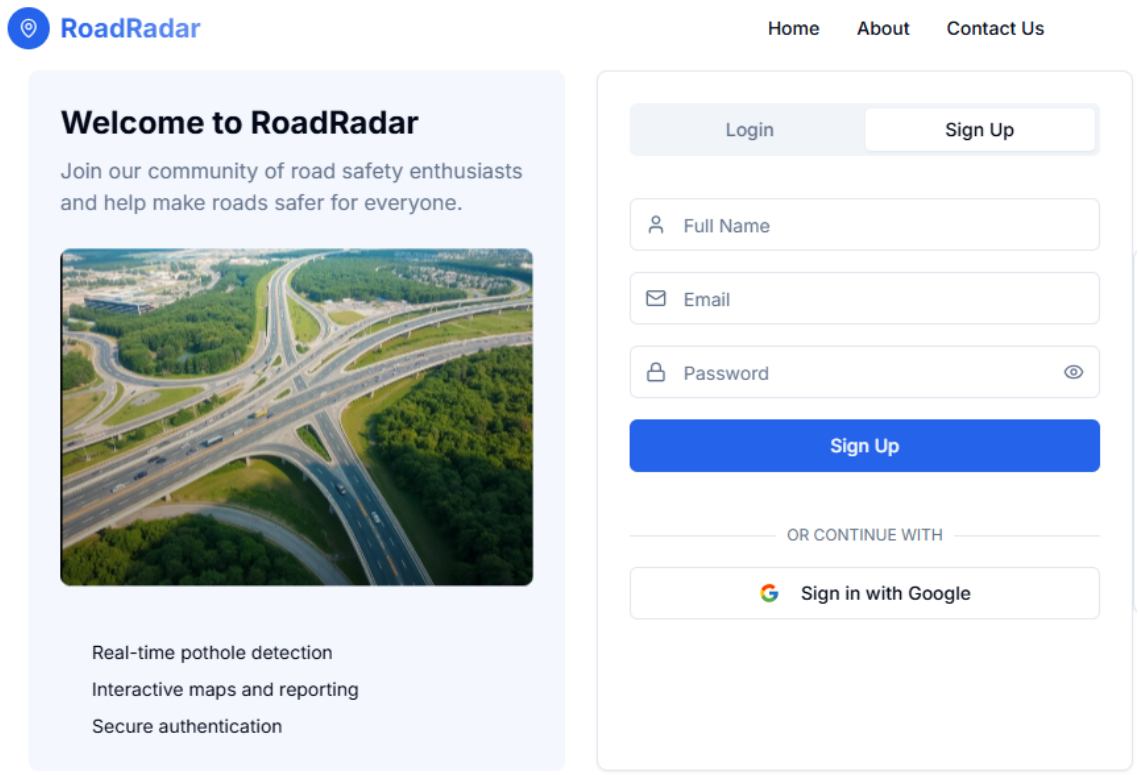

Figure 11.3: Signup

11.4 Dashboard

All of the Road Radar web interface's key features are accessible from a single page through the Dashboard, which serves as the main control panel. It is intended to provide users with an easy-to-use interface for account management and real-time detection log viewing.

Among the dashboard's essential elements are:

Users can swiftly navigate back to the main homepage by using the "Home" button.

User Information: Provides user information for convenience, like name and email.

Detection Logs: Provides real-time information about road crack detections, such as GPS coordinates, crack type, and confidence score.

Bot necessary, bot activation turns on or off the road damage detection model (for live

deployment scenarios).

Speak with Support: you a direct way to contact technical support for help.

Logout: This safely ends the user's session.

The dashboard is completely functional and responsive on all devices, enabling users to monitor critical data, manage their activities, and engage with the system in a safe and effective manner.

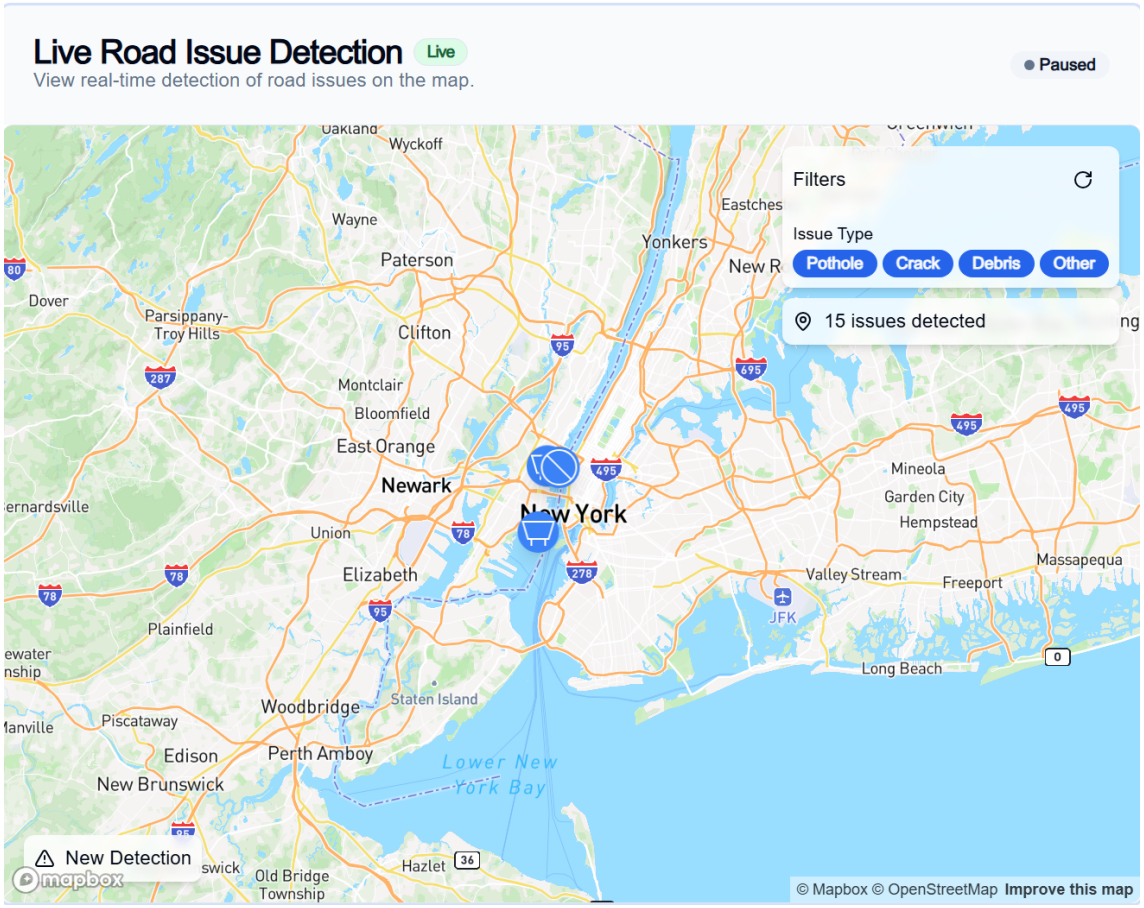

Figure 11.4: Dashboard

11.5 Results

Users can view all of the road damage detection logs gathered by the Road Radar system in the See Results area. This section shows real-time crack identification and geographic tracking analytics rather than trading performance.

Every log entry contains:

Crack Type: The type of road damage identified, such as an alligator crack,longitudinal crack, transverse crack, or pothole.

Confidence Score: The degree of detection confidence that shows how confident the model is in the outcome.

GPS coordinates: The precise place where damage was found.

These records assist users in tracking the locations of damaged road segments and monitoring the detection performance. The real-time updates facilitate maintenance planning and allow for prompt decision-making. Firebase is used to store all recorded data for safe, cloud-based access.

Chapter 12

Conclusions and Future Work

12.1 Conclusions

In order to improve road maintenance procedures and raise public safety, we have created an Internet of Things (IoT)-driven system for automatic road damage identification and reporting. Our main goals were to offer a cloud-based dashboard for visualizing road damage data and to develop a system that combines lightweight deep learning models—specifically, YOLOv5n—with IoT technologies for real-time anomaly identification.

The project included a number of essential elements: 1. The development and application of deep learning methods for detecting road damage. 2. Integration of Internet of Things devices (like Raspberry Pi) for anomaly detection and real-time data collecting. 3. The creation of a cloud-based dashboard to help with timely maintenance decisions and the visualization of road damage information.

According to our findings, the system can accurately and quickly identify and report road irregularities like potholes and cracks. The cloud-based dashboard ensures effective resource allocation by giving municipal authorities an easy-to-use interface to view and prioritize maintenance jobs. The system's usefulness in practical situations is further increased by the incorporation of GPS-enabled IoT devices, which enable accurate damage location.

Additionally, the system lessens the need for manual road inspections, which speeds up

maintenance and minimizes human error. This research shows how smart systems may enhance road safety and infrastructure management by utilizing deep learning and IoT technologies.

All things considered, this research demonstrates the revolutionary potential of fusing AI and IoT technologies for infrastructure monitoring. Future developments in automated infrastructure management and smart city solutions are made possible by the success of this project.

12.2 Future Work

While this project has achieved its primary objectives, several areas for future work and improvement have been identified:

12.2.1 Enhanced Model Training

To increase the YOLOv5n model's accuracy and resilience in a range of situations, future iterations might concentrate on training it on a wider variety of datasets, such as photos taken in varying lighting and weather conditions. The model's capacity to generalize would be improved by enlarging the dataset to include a wider range of road types and damage patterns.

12.2.2 Advanced Dashboard Features

Additional capabilities like automated maintenance work scheduling, predictive analytics for predicting trends in road deterioration, and interaction with current municipal administration systems could improve the cloud-based dashboard.

12.2.3 Scalability and Multi-Device Integration

The system's scalability for larger road networks would be enhanced by expanding it to accommodate several IoT devices at once. Creating effective communication protocols to manage real-time data from several devices without sacrificing performance may be the focus of future research.

12.2.4 Integration with Smart City Frameworks

Its usefulness would be increased by integrating the technology into larger smart city frameworks. To offer a more complete infrastructure management solution, this can involve integration with environmental monitoring networks, traffic monitoring systems, and urban planning tools.

12.2.5 Real-World Deployment and Feedback

Implementing the technology in practical settings, such pilot programs with local government agencies, would yield insightful information for improvement. The effectiveness and usefulness of the system would be enhanced by taking into account input from road maintenance teams and other stakeholders.

Optimizing the hardware configuration and increasing energy economy are two important factors for upcoming upgrades to the Road Radar system. It is essential to guarantee low power consumption because the system is built to operate constantly—processing video feeds, identifying road damage in real time, and uploading records to the cloud—especially when it is installed on mobile platforms like cars or roadside units.

Subsequent versions of the system can include:

Low-power hardware elements that can process models quickly while using less power, including more energy-efficient Raspberry Pi models or specialized AI accelerators like the Google Coral or NVIDIA Jetson Nano.

Power management strategies include dynamic throttling of processing power during low-

demand times, adaptive frame rate control based on speed or environment, and planned sleep cycles when no activity is detected.

In off-grid deployments, solar or other power sources can prolong operational lifetime and lessen reliance on vehicle power or battery repairs.

In addition to decreasing the need for manual maintenance, these improvements would make the system more scalable and sustainable for widespread use.

In summary, the creation of an Internet of Things (IoT)-powered system for detecting and reporting road damage is a major breakthrough in the application of smart technologies to improve infrastructure management, real-time monitoring, and road safety. The Road Radar platform has the potential to be a key component of creating safer, smarter, and more responsive cities with more research and development in energy efficiency, AI modeling, and system integration.

Bibliography

[1] K. R. Ahmed. Smart pothole detection using deep learning based on dilated convolution. *Sensors*, 21(12):1–17, 2021.

[2] L. Bhamare, G. Varade, H. Mehta, and N. Mitra. Iot-based highway potholes and hump detection. *Research Centre, Department of Electronics & Telecommunication Engineering, Oriental University, Indore, India, and Research & Development, Aethertec Innovative Solutions, Pune, India*, 2023.

[3] S. Kadam, S. Jha, P. Rai, and V. Bagade. Iot-based pothole detection system. *Department of Electronic and Telecommunication Engineering, Alamuri Ratnamala Institute of Engineering and Technology, Thane, Maharashtra, India*, 2023.

[4] M. W. Khan, M. S. Obaidat, K. Mahmood, D. Batool, H. M. S. Badar, M. Aamir, and W. Gao. Real-time road damage detection and infrastructure evaluation leveraging unmanned aerial vehicles and tiny machine learning. *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[5] R. Sulistyowati et al. Monitoring of road damage detection systems using image processing methods and google map. *IOP Conference Series: Materials Science and Engineering*, 1010:012017, 2021.

[6] S. Thiruppathiraj, U. Kumar, and S. Buchke. Automatic pothole classification and segmentation using android smartphone sensors and camera images with machine learning techniques. *2020 IEEE Region 10 Conference (TENCON)*, November 2020.