A MAJOR PROJECT REPORT ON

Machine Learning for Road Traffic Accident Classification and Spatial Network Analysis for Safe Routing

Submitted in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Computer Science and Engineering

by

NOTAM SWETHA SRI (21A81A05H5)

EROTHU VINUTHNA (21A81A05E4)

MENDA ANUSHA (21A81A05G7)

TATINA BHANU BHAVANI (21A81A05J2)

DULAPALLI KEERTHI PRIYA (21A81A05E3)

Under the Esteemed Supervision of

Dr.V.S.Naresh M.Tech.,Ph.D

Professor & Dean(R&D).

Department of Computer Science and Engineering (Accredited by N.B.A.)

SRI VASAVI ENGINEERING COLLEGE(Autonomous)

(Affiliated to JNTUK, Kakinada)

Pedatadepalli, Tadepalligudem-534101, A.P 2024-2025

SRI VASAVI ENGINEERING COLLEGE (Autonomous)

Department Of Computer Science and Engineering

Pedatadepalli , Tadepalligudem

Certificate

This is to certify that the Project Report entitled "Machine Learning for Road Traffic Accident Classification and Spatial Network Analysis for Safe Routing" Submitted by N.Swetha Sri (21A81A05H5) , E.Vinuthna (21A81A05E4), M.Anusha (21A81A05G7) ,T.Bhanu Bhavani (21A81A05J2) , D.Keerthi Priya (21A81A05E3) for the award of the degree of Bachelor of Technology in the Department of Computer Science and Engineering during the academic year 2024-2025.

Name of Project Guide
Dr.V.S.Naresh, M.Tech.,Ph.D.,
Professor & Dean(R&D).

Head of the Department
Dr. D Jaya Kumari, M.Tech.,Ph.D.,
Professor & HOD.

External Examiner

DECLARATION

We here by declare that the project report entitled **"Machine Learning for Road Traffic Accident Classification and Spatial Network Analysis for Safe Routing"** submitted by us to Sri Vasavi Engineering College(Autonomous), Tadepalligudem, affiliated to JNTUK Kakinada in partial fulfilment of the requirement for the award of the degree of B.Tech in Computer Science and Engineering is a record of Bonafide project work carried out by us under the guidance of **Dr.V.S.Naresh (Professor & Dean(R&D))**. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree in this institute or any other institute or University.

Project Associates

N. Swetha Sri (21A81A05H5)

E. Vinuthna (21A81A05E4)

M. Anusha(21A81A05G7)

T. Bhanu Bhavani (21A81A05J2)

D. Keerthi Priya (2 1A81A05E3)

ACKNOWLEDGEMENT

First and foremost, we sincerely salute to our esteemed institute **SRI VASAVI ENGINEERING COLLEGE,** for giving us this golden opportunity to fulfill our warm dream to become an engineer.

Our sincere gratitude to our project guide **Dr.V.S.Naresh(Professor & Dean(R&D))**, Department of Computer Science and Engineering**,** for his timely cooperation and valuable suggestions while carrying out this project.

We express our sincere thanks and heartful gratitude to **Dr. D. Jaya Kumari**, Professor & Head of the Department of Computer Science and Engineering, for permitting us to do our project.

We express our sincere thanks and heartful gratitude to **Dr. G.V.N.S.R. Ratnakara Rao**, Principal, for providing a favourable environment and supporting us during the development of this project.

Our special thanks to the management and all the teaching and non-teaching staff members, Department of Computer Science and Engineering, for their support and cooperation in various ways during our project work. It is our pleasure to acknowledge the help of all those respected individuals.

We would like to express our gratitude to our parents, friends who helped to complete this project.

Project Associates

N. Swetha Sri (21A81A05H5)

E. Vinuthna (21A81A05E4)

M. Anusha(21A81A05G7)

T. Bhanu Bhavani (21A81A05J2)

D. Keerthi Priya (2 1A81A05E3)

ABSTRACT

Road accidents are a major global concern, leading to loss of life, property damage, and economic costs. With advancements in data analytics and machine learning (ML), it is now possible to analyze complex patterns in accident data to predict and potentially reduce future incidents. This project focuses on developing an ML-based system for road traffic accident prediction combined with spatial network analysis to enhance safe routing. Extensive datasets containing variables such as weather conditions, traffic density, time of day, road types, and driver behaviors are collected and pre-processed. Statistical analysis and exploratory techniques help identify key factors influencing accident severity and frequency. Predictive modelling employs algorithms like Logistic Regression, Random Forest, and Neural Networks to forecast accident likelihood under specific conditions, evaluated using performance metrics such as accuracy, precision, recall, and F1-score. A key innovation is the integration of GPS technology with spatial network analysis, enabling real-time identification of accident-prone areas and suggesting safer alternative routes for drivers. Based on real- time user inputs and location data, the system predicts accident severity and, in high- risk scenarios or actual accidents, automatically triggers an SMS alert to the certain listed contacts with live location. This rapid response mechanism minimizes emergency response times, while the safe routing feature proactively reduces accident risks. By combining predictive analytics with real time spatial data, the project contributes to smarter, safer transportation systems and improved road safety outcome.

TABLE OF CONTENTS

LIST TO FIGURES

CHAPTER-1

INTRODUCTION

1.INTRODUCTION

1.1 Introduction:

Traffic accidents are a significant global concern, leading to fatalities, injuries, and economic loss. Traditional accident analysis focuses on historical data, lacking real-time risk assessment and predictive analytics. This project introduces a proactive accident prediction system, leveraging, machine learning and spatial network analysis, to enhance road safety. The system aims to predict accident-prone areas, assess real-time risk factors, and recommend safer routes to travelers. By integrating real-time monitoring and predictive modeling, the system enhances accident prevention measures, ensuring a safer commuting experience

1.2 Motivation:

The increasing number of road accidents worldwide has heightened the urgency for an effective prevention system. Traditional systems often rely on reactive measures, such as post-accident analyses, which fail to provide timely interventions. The integration of artificial intelligence (AI) and machine learning (ML) in accident prediction can significantly improve the accuracy and timeliness of accident prevention measures. By leveraging advancements in AI and ML, the proposed system aims to predict accident-prone areas and recommend safer routes, ultimately reducing the incidence of road accidents and fatalities.

Existing accident prediction systems have several limitations, including the lack of real-time risk assessment and the inability to provide safe route recommendations. Additionally, current systems do not efficiently integrate multiple data sources, such as weather conditions, traffic density, and road conditions, leading to inaccurate risk assessments. The system's ability to provide real-time risk assessments and safe route recommendations will enhance road safety and optimize traffic management.

1.3 Objective:

This project focuses on developing a machine learning-based system to predict accident likelihood by analyzing historical data, weather, traffic density, and road conditions. It will integrate real-time data for dynamic risk assessment and timely interventions. A spatial network analysis module will recommend safer routes by identifying accident-prone areas. The system will be deployed as a web or mobile application, offering real-time risk assessments, safe route suggestions, and emergency alerts. It will assist government agencies in planning road safety measures and help insurance companies assess risk factors for fair policy pricing.

1.4 Project Outline

Chapter-1	Introduction
Chapter-2	Literature Survey
Chapter-3	System Study and Analysis
Chapter-4	System Design
Chapter-5	Technologies
Chapter-6	Implementation
Chapter-7	Testing
Chapter-8	Screen shots
Chapter-9	Conclusion and Future Work
Chapter-10	References

1.5 Scope

This project enhances road safety by using machine learning and spatial network analysis for accident prediction and classification.

Key factors analyzed include :

- Weather conditions
- Traffic density
- Road type
- Driver behaviour

A major innovation is the integration of:

- Real-time GPS data
- Spatial network analysis for safe route recommendations

This proactive approach helps:

- Drivers make informed decisions
- Reduce accident risks

CHAPTER-2

LITERATURE SURVEY

2.LITERATURE SURVEY

Title: A model for predicting traffic accident severity using Convolutional Neural Networks (CNN).

Author: Lu Wenqi, Luo Dongyu, and Yan Menghua (2017)

The study focused on leveraging deep learning techniques to improve the accuracy of accident severity prediction, demonstrating the effectiveness of CNN in handling complex traffic data.

Title: The Traffic Accident Prediction based on Neural Network

Author: Fu Huilin and Zhou Yucai (2017)

This compared the traditional linear analysis approach with a Back Propagation (BP) Neural Network for traffic accident prediction. They found that linear analysis methods fail to reveal the real situation accurately, leading to unsatisfactory prediction results. Their study highlights the advantages of using neural networks over traditional methods.

Title: Accident Severity Classification Using Machine Learning Techniques

Author: Kumar et al. (2021)

They developed a machine learning-based accident severity classification model. Logistic regression and decision trees were used to classify accidents into minor, major, or fatal categories. Their approach demonstrated the effectiveness of machine learning techniques in analyzing accident severity based on historical data.

Title: A Graph-Based Safe Routing Model for Urban Traffic Management

Author: Lee & Park (2019)

They proposed a graph-based safe routing model for urban traffic management. The model utilizes historical accident data to identify routes with lower accident risks, helping to improve road safety and optimize traffic flow by providing safer travel paths.

Title: Road Accident Severity Prediction Using Ensemble Machine Learning Models

Author: Singh and Patel (2022)

They introduced a machine learning-based approach for road accident severity prediction using Random Forest and XGBoost. Their study demonstrated the capability of ensemble learning techniques in enhancing prediction accuracy and providing better insights into accident severity levels.

CHAPTER-3

SYSTEM STUDY AND ANALYSIS

3.SYSTEM STUDY AND ANALYSIS

3.1 PROBLEM STATEMENT:

Current accident prediction systems are reactive, lack of real-time monitoring, and do not provide safe routing options. Many existing models fail to consider dynamic factors such as weather conditions, real-time traffic flow, and road conditions, leading to inaccurate accident risk assessments. The absence of an automated emergency alert system further delays response time, increasing the severity of accidents.

3.2 EXISTING SYSTEM:

- Uses historical accident data without real-time updates.
- Lacks real-time data integration from traffic, weather, and road conditions.
- Does not suggest safer alternative routes, leaving users unaware of potential risks.
- Emergency response is manual and delayed, causing inefficiencies in rescue operations.
- No predictive analytics for accident risk assessment, relying instead on static reports.

3.3 LIMITATIONS IN EXISTING SYSTEM:

- **Reactive Approach** – Existing methods analyze past accidents instead of predicting and preventing future incidents.
- **Lack of Real-Time Risk Assessment** – Current systems do not incorporate live data like weather, traffic, and road conditions for accident prediction.
- **Inefficient Emergency Response** – Delayed accident reporting slows down emergency response, increasing casualty risks.
- **Limited Safe Routing Mechanisms** – Navigation systems focus on the shortest route rather than avoiding accident-prone areas.
- **Minimal Use of Machine Learning and Spatial Analysis** – Traditional systems lack advanced ML models to accurately predict and analyze accident-prone zones.

3.4 PROPOSED SYSTEM:

The proposed system integrates ML algorithms, real-time data, and spatial network analysis to predict accidents and enhance safety. This system provides dynamic risk assessment, safe route recommendations, and automated emergency alerts to minimize accident risks and improve road safety. Additionally, it utilizes GPS integration to analyze user routes and suggest safer alternatives in real-time. The ML algorithms will analyze historical accident data, traffic patterns, weather conditions, and road quality to identify potential accident hotspots and predict the likelihood of accidents.

The real-time data integration ensures that the system continuously updates its risk assessments based on the latest information from traffic sensors, weather stations, and GPS devices. The spatial network analysis component will identify the safest routes for users by considering accident-prone areas and traffic density. Automated emergency alerts will be sent to authorities and users in case of high-risk situations, ensuring a prompt response to potential accidents. By combining these features, the proposed system aims to create a comprehensive and proactive solution for accident prevention and road safety enhancement.

3.5 ADVANTAGES OF PROPOSED SYSTEM:

- **Accident Risk Prediction** – ML algorithms analyze historical and real-time data to predict accident-prone areas, helping users take preventive measures.
- **Real-Time Safety Updates** – Continuous data integration ensures up-to-date risk assessments based on traffic, weather, and road conditions.
- **Safe Route Recommendations** – Provides alternate, safer routes by analyzing accident hotspots and traffic density.
- **Automated Emergency Alerts** – Notifies authorities and users in high-risk situations, ensuring a faster response to emergencies.
- **GPS Integration for Navigation** – Analyses user routes in real-time and suggests safer alternatives, reducing accident risks.

- **Data-Driven Decision Making** – Helps authorities plan better road safety measures by identifying accident trends and risk factors.
- **Enhanced Traffic Management** – Reduces congestion by recommending safe and efficient routes, improving overall traffic flow.
- **Improved Road Safety Awareness** – Alerts drivers about potential dangers ahead, promoting cautious driving behaviour.
- **Weather and Road Condition Analysis** – Factors in adverse weather conditions and poor road quality to enhance risk predictions.
- **Scalability and Adaptability** – Can be expanded to integrate additional data sources, making it adaptable to different locations and evolving road conditions.

3.6 FUNCTIONAL REQUIREMENTS:

- **Accident Data Analysis** – The system should collect, store, and analyze historical accident data to identify trends and high-risk areas.
- **Predictive Modelling** – Machine Learning algorithms should predict accident probabilities based on historical data, traffic patterns, weather conditions, and road quality.
- **Real-Time Monitoring** – The system should continuously gather live data from GPS, traffic sensors, and weather stations to update risk assessments dynamically.
- **Safe Routing Recommendation** – The system should provide users with optimized, low-risk route suggestions based on real-time accident risk analysis and traffic conditions.
- **User Interface** – A user-friendly application should display safe route recommendations, and emergency alerts.
- **Integration with External Systems** – The system should connect with traffic management systems, emergency services, weather APIs, and navigation apps for seamless data exchange.
- **Traffic and Road Condition Analysis** – The system should assess real-time road conditions, traffic congestion, and weather hazards to enhance accident predictions.

3.7 NON FUNCTIONAL REQUIREMENTS:

- **Scalability** – The system should handle increasing users, vehicles, and data sources efficiently.
- **Reliability** – Must provide accurate and consistent accident predictions and safe route recommendations.
- **Performance** – Should process real-time data with minimal latency for quick decision-making.
- **Security** – Must ensure data privacy, encryption, and protection from unauthorized access.
- **Usability** – The interface should be user-friendly and easy to navigate for all drivers.
- **Availability** – The system should operate 24/7 with minimal downtime.
- **Interoperability** – Must seamlessly integrate with external systems like GPS navigation, emergency services, and traffic management platforms.

3.8 SYSTEM REQUIREMENTS

3.8.1 Hardware Requirements

- **Server Processor** – Intel Xeon/AMD Ryzen (multi-core)
- **RAM** – Minimum 16GB
- **Storage** – SSD, 512GB+
- **GPU** – NVIDIA RTX 3090 or higher (for ML models)
- **User Device** – GPS-enabled smartphone (Android/iOS)
- **Network** – High-speed internet connection
- **IoT Sensors (if applicable)** – Traffic cameras, weather sensors, GPS tracker

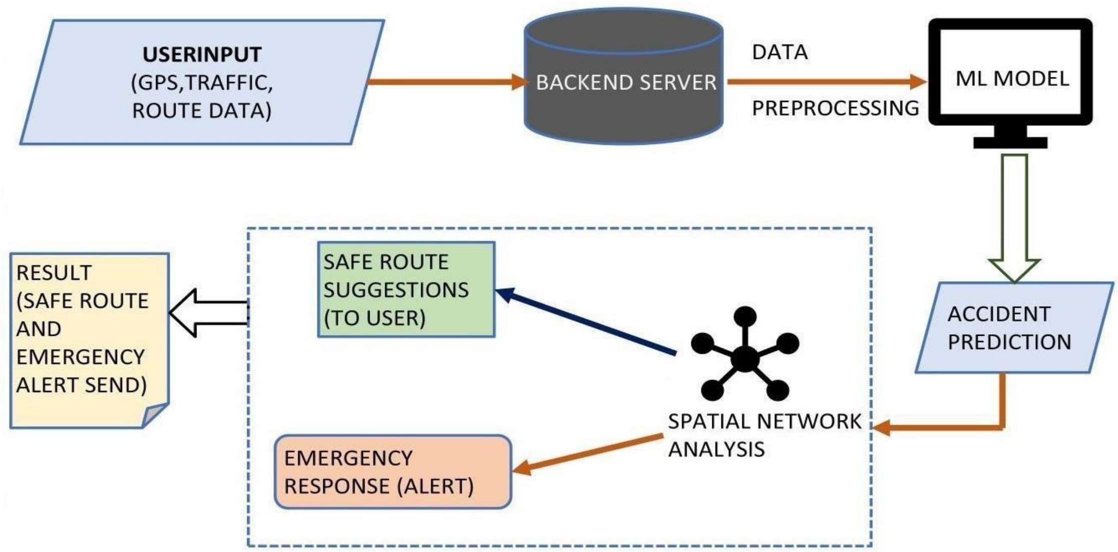
3.8.2 Software Requirements

- **Operating System** – Windows, Linux, or macOS (for development and deployment)
- **Programming Languages** – Python (for ML algorithms), JavaScript
- **Frameworks & Libraries** – TensorFlow (ML models), Flask (backend),Numpy.Pandas,Scikit-Learn,NetworkX,
- **APIs & Integration** – Google Maps API (for GPS and navigation),Traffic API (for real-time traffic updates)
- **Development Tools** – Jupyter Notebook (for ML development), Git/GitHub (for version control)

CHAPTER-4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:-



This system architecture diagram represents a road accident analysis, classification, and prediction system using machine learning, GPS data, and spatial network analysis to provide safe route suggestions and emergency response alerts.

Explanation of Components and Flow:

1. **User Input (GPS, Traffic, Route Data)**
   - The system receives real-time data from users, including GPS location, traffic conditions, and route data.
   - This data is sent to the Backend Server for further processing.
2. **Backend Server**
   - Stores and preprocesses the incoming data.
   - Prepares the data to be fed into the Machine Learning (ML) Model.
3. **Machine Learning Model**
   - Processes the data and performs Accident Prediction based on historical accident data, traffic conditions, and route-specific risks.
   - The prediction helps determine high-risk areas.
4. **Spatial Network Analysis**
   - Uses accident prediction data to analyze the road network and determine safe alternative routes.
   - Provides Safe Route Suggestions to users.
   - If an accident is detected or predicted, it triggers the Emergency Response System.
5. **Safe Route Suggestions (To User)**
   - Based on accident risk analysis, the system suggests the safest possible route to the user.
6. **Emergency Response (Alert System)**
   - If an accident occurs or is predicted in a high-risk zone, an emergency alert is sent to authorities.
   - The system ensures quick response to accidents.
7. **Final Result (Safe Route & Emergency Alert Sent)**
   - The user receives safe route recommendations.
   - Emergency responders receive an alert if needed.

4.2 UML Diagrams

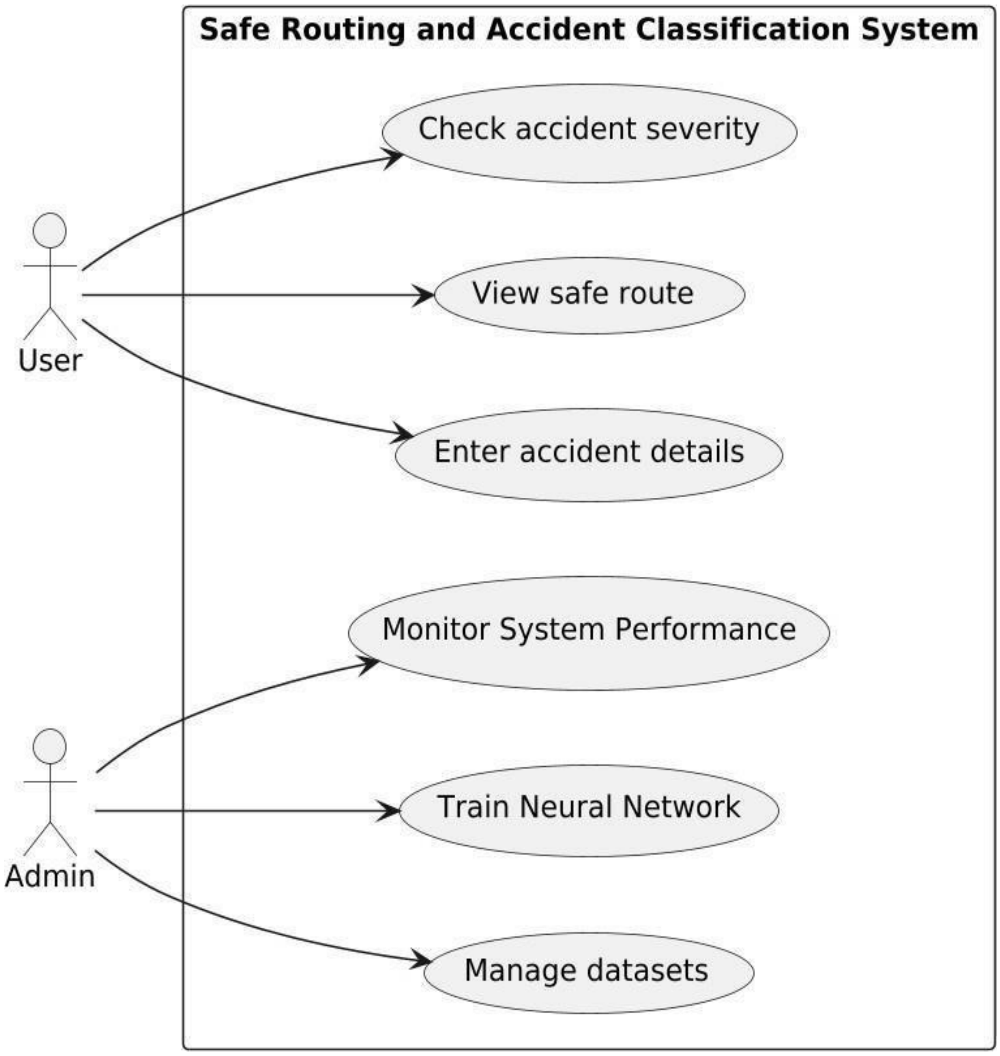
4.2.1 USE CASE DIAGRAM

User Actions

- Enter Accident Details: User logs in, reports accident details, and submits the form.
- View Safe Route: User inputs locations; the system suggests the safest route.
- Check Accident Severity: User provides details; the system classifies severity using a neural network.

Admin Actions

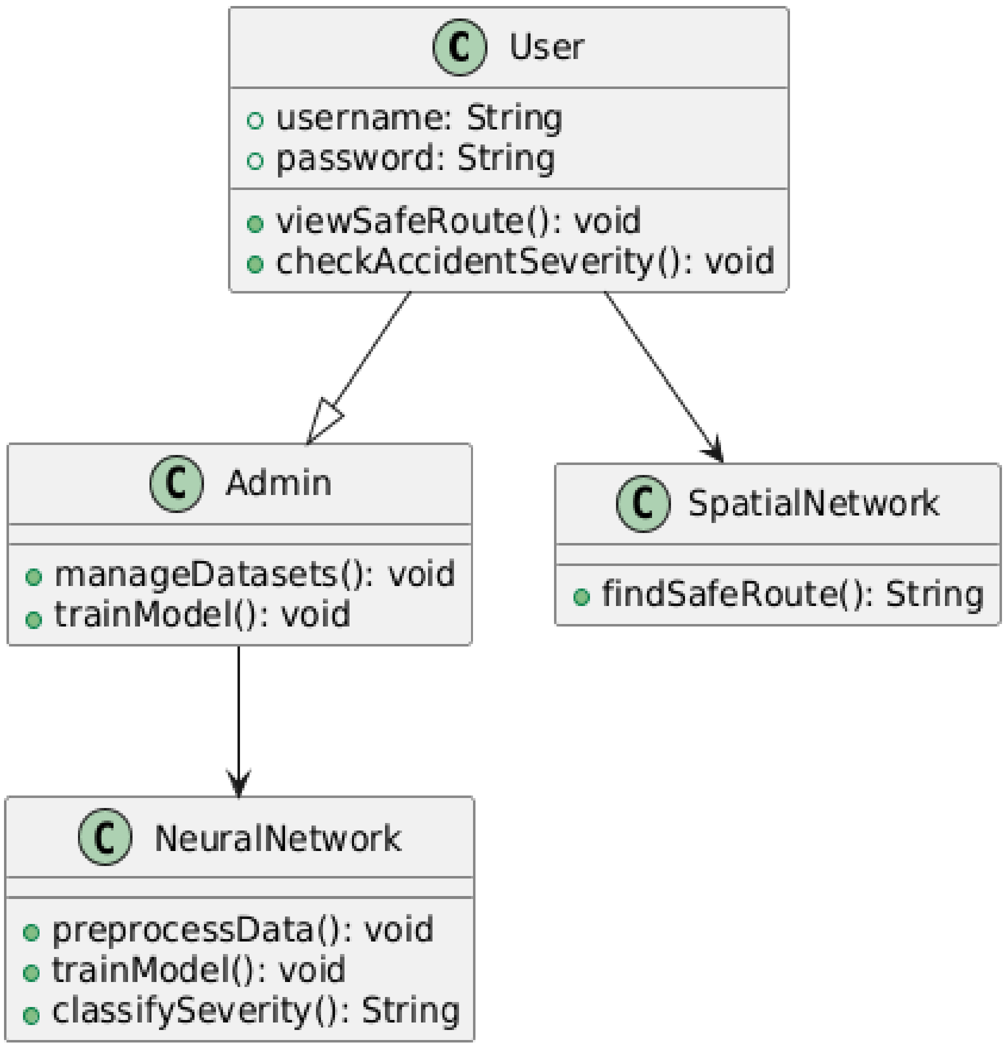
- Manage Datasets: Admin uploads, updates, and preprocesses accident data.
- Train Neural Network: Admin trains the model with historical data and reviews performance.
- Monitor System Performance: Admin checks logs, updates models, and ensures system stability.



4.2.2 CLASS DIAGRAM

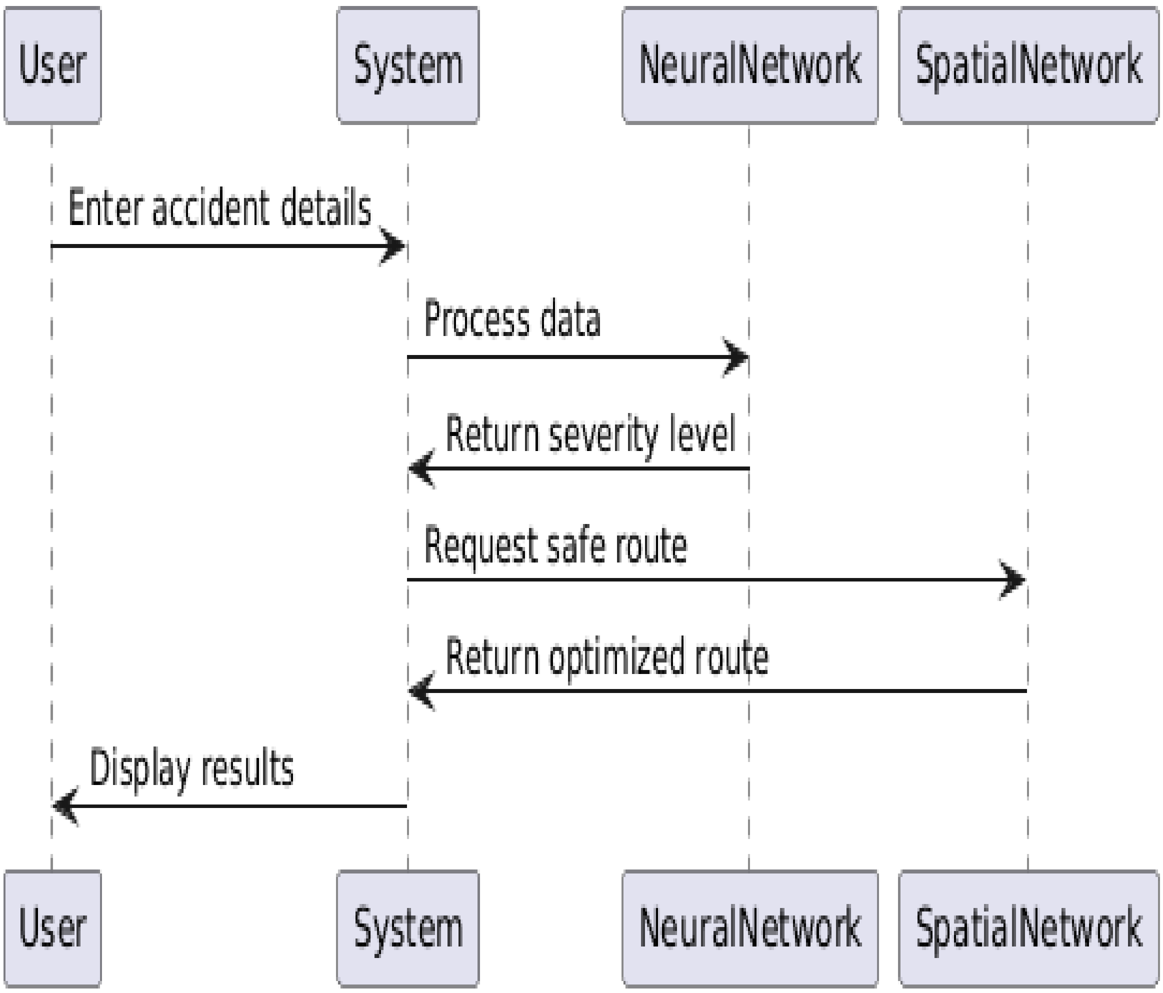
Steps for Class Diagram

- User logs in and calls viewSafeRoute() (via SpatialNetwork) and checkAccidentSeverity() (via NeuralNetwork).
- Admin manages datasets (manageDatasets()) and retrains the model (trainModel()).
- SpatialNetwork processes findSafeRoute() for safe routing.
- NeuralNetwork handles preprocessData(), trainModel(), and classifySeverity() for accident classification.



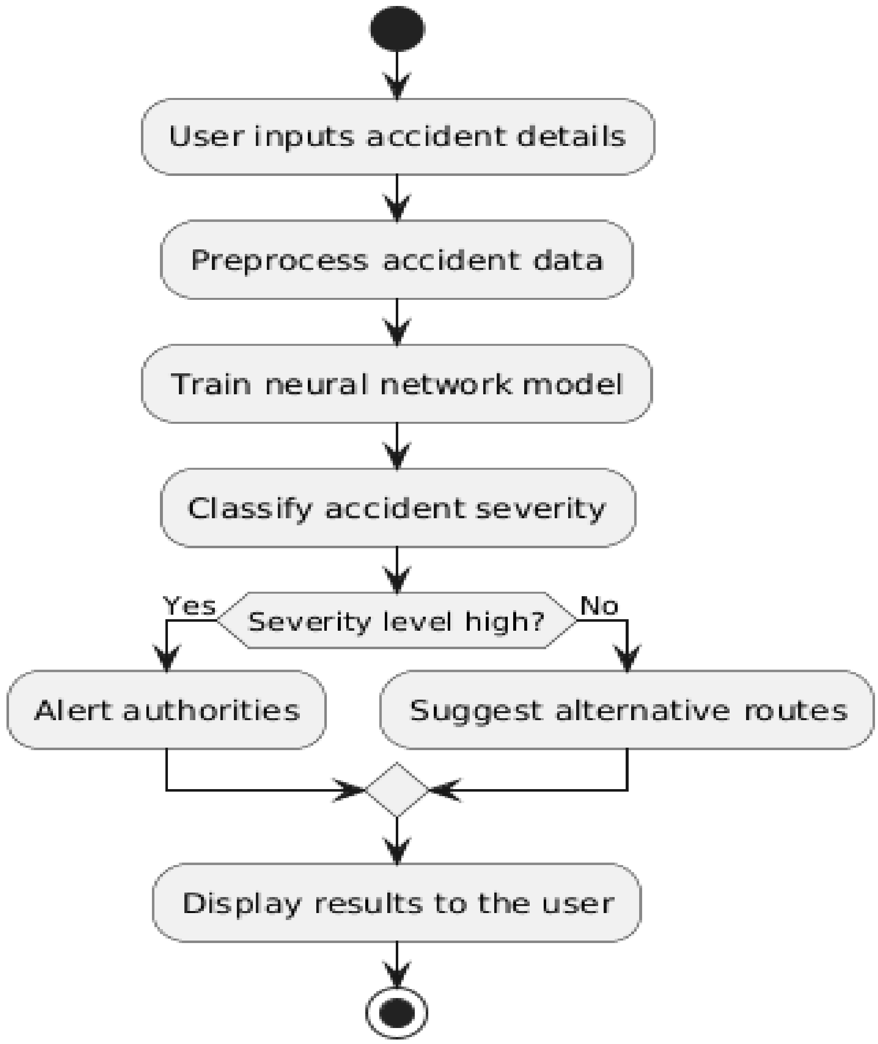
4.2.3 SEQUENCE DIAGRAM

- User enters accident details into the System .
- System sends data to NeuralNetwork for processing.
- NeuralNetwork analyzes and returns the accident severity level.
- System requests a safe route from SpatialNetwork .
- SpatialNetwork computes and returns an optimized route.
- System displays the results to the User .



4.2.4 ACTIVITY DIAGRAM

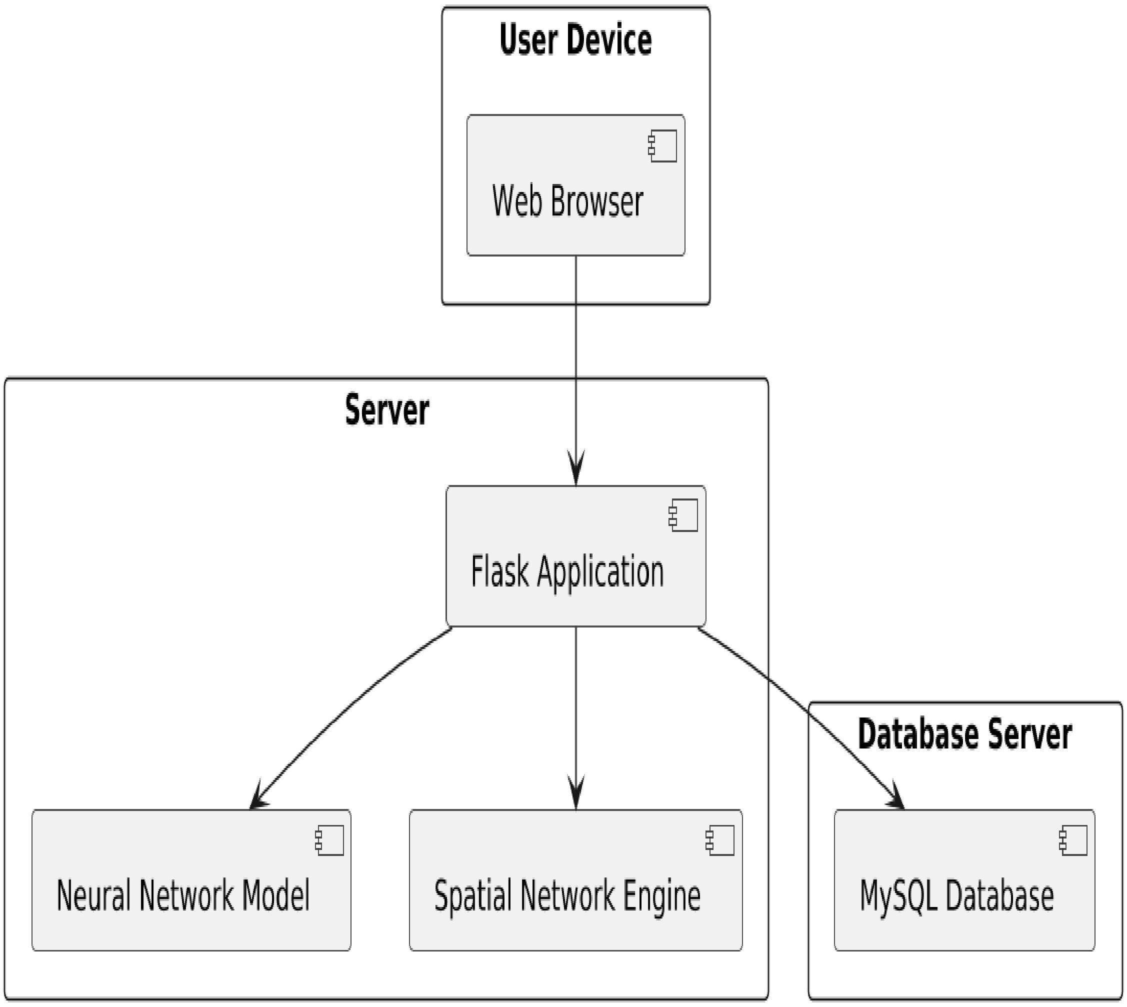
- User inputs accident details into the system.
- System preprocesses accident data for analysis.
- Neural network model is trained using the data.
- System classifies the accident severity based on the trained model.
- Decision Point: Is the severity level high?
  - Yes → Alert authorities for emergency response.
  - No → Suggest alternative safe routes to the user.
- System displays results to the user.



4.2.5 Deployment diagram

Steps for Deployment Diagram

- User Access: The user submits a request via a web browser.
- Flask Processing: The Flask app handles the request, calling the Neural Network Model for severity analysis or the Spatial Network Engine for routing.
- Database Interaction: The Flask app retrieves or updates accident data in MySQL.
- Response: Processed results are sent back to the user.



CHAPTER 5

TECHNOLOGIES

5. TECHNOLOGIES

1. Operating Systems

The project is designed to be versatile and deployable across multiple platforms, including Windows (XP, 7, 8, 10, 11), macOS, Android, and iOS. This multi-platform compatibility ensures that the application can reach a wide user base and function effectively on both desktop and mobile devices.

2. Python

Python serves as the core programming language for the project, powering backend development, data processing, and machine learning functionalities. Its extensive ecosystem of libraries and frameworks enables efficient and effective handling of complex tasks, making it an ideal choice for building robust applications.

3. JavaScript

JavaScript is employed to create dynamic, interactive elements within the application, significantly enhancing the user interface and overall experience. Its integration ensures that the front-end components are responsive and engaging, providing seamless interaction with the underlying system functionalities.

4. Web Framework – Flask

Flask is utilized as the web framework for developing and deploying the web-based interface of the project. This lightweight framework offers simplicity and flexibility, allowing for efficient integration between the frontend and backend, which is critical for building a scalable and responsive web application.

5. Data Processing Libraries

The project leverages data processing libraries such as NumPy and Pandas to efficiently manage and manipulate data. These libraries provide powerful tools for numerical computations and data analysis, forming the backbone for data-driven operations and insights within the application.

6. Machine Learning Libraries

For the machine learning components, Scikit-Learn and TensorFlow are key libraries that enable the development, training, and deployment of predictive models. These libraries offer comprehensive tools and frameworks .

7. Data Visualization

Matplotlib is used for data visualization, allowing the project to generate clear and informative graphical representations of data. This capability is crucial for analyzing patterns, trends, and anomalies, thereby supporting better decision-making and communication of insights.

8. Geospatial and Network Analysis

GeoPandas and NetworkX are incorporated to perform geospatial analysis and network-based computations. These libraries provide specialized tools to handle location-based data and optimize network structures, which are essential for applications that involve mapping, routing, and spatial analytics.

9. Cloud Computing and Storage

The project utilizes cloud services such as Google Colab, AWS, and Google Cloud for scalable computation and storage solutions. These platforms facilitate the handling of large datasets and the execution of resource-intensive tasks, ensuring that the system remains responsive and efficient under varying workloads.

10. HTML

HTML (Hyper Text Markup Language) is used to structure the web pages of the project. It defines the layout and elements of the user interface, ensuring proper organization of content such as headings, forms, buttons, and other essential components. HTML provides the backbone for the visual representation of the application.

11. CSS

CSS (Cascading Style Sheets) is employed to enhance the appearance of the web pages, making them visually appealing and user-friendly. It allows for styling elements, including colors, fonts, layouts, and responsive design, ensuring that the interface is both attractive and adaptable across different devices and screen sizes.

12. Spatial Network

The project incorporates a spatial network approach to enhance safe routing. This technique leverages geospatial data and graph-based algorithms to analyze road networks, identify accident-prone areas, and compute safer travel paths. By integrating spatial analysis into the routing process, the system not only optimizes for efficiency but also prioritizes user safety, offering dynamic route recommendations based on real-time and historical data.

13. Neural Networks

The project utilizes neural networks to classify the severity of road accidents. By processing a multitude of features from historical accident data, these neural models are trained to predict whether an incident is fatal, serious or slight. This classification capability not only enhances the understanding of accident patterns but also facilitates more targeted and effective emergency responses, contributing to overall road safety improvements

CHAPTER 6

IMPLEMENTATION

6. IMPLEMENTATION

1. Requirements Analysis and System Design:

Begin by defining the overall system requirements and architecture. This involves identifying key functionalities, such as accident severity classification using neural networks and safe routing via spatial network analysis. Develop detailed flowcharts and design documents that outline how datasets, machine learning models, and routing algorithms will interact, as well as how the frontend and backend components integrate.

2. Dataset Collection and Preprocessing:

Gather datasets related to accidents, vehicles, and casualties. Once collected, preprocess the data using Python libraries like Pandas and NumPy. This step includes cleaning the data by handling missing values, normalizing numerical variables, encoding categorical features, and balancing the dataset to ensure that the model training process is effective.

3. Exploratory Data Analysis (EDA) and Feature Engineering:

Conduct an in-depth exploratory analysis of the datasets to uncover underlying patterns and correlations. Use visualization tools like Matplotlib to better understand the data distribution. Based on the insights, perform feature engineering to extract and select the most relevant features for predicting accident severity and enhancing the safe routing mechanism.

4. Neural Network Model Development for Accident Severity Classification:

Design and implement a neural network model using TensorFlow and Scikit-Learn to classify road accident severity into categories (e.g., slight,serious,fatal). Split the preprocessed data into training and testing sets. Train the neural network on the training set while validating its performance with metrics such as accuracy, precision, recall, and F1-score. Optimize the model through hyperparameter tuning to improve its performance.

5. Implementation of the Spatial Network for Safe Routing:

Develop a spatial network model using libraries such as GeoPandas and NetworkX. This model leverages historical accident data to analyze road networks and identify accident-prone areas. The spatial network computes safer alternative routes by integrating real-time factors such as traffic and weather conditions, ensuring that the recommended paths prioritize user safety over just the shortest distance.

6. Frontend and Backend Integration:

Create the web-based user interface using HTML, CSS, and JavaScript to ensure an interactive and responsive experience. Develop the backend using Flask to serve as a bridge between the frontend and the machine learning models. Implement RESTful APIs that handle user inputs, trigger the neural network model for accident classification, and provide safe routing recommendations.

7. Testing, Evaluation, and Deployment:

Thoroughly test each component through unit testing and integration testing to ensure seamless data flow and proper functionality. Evaluate the neural network's performance and the efficiency of the safe routing algorithm using real-world scenarios. Once validated, deploy the application on cloud platforms like AWS or Google Cloud, ensuring scalability and continuous monitoring for ongoing improvements.

CHAPTER 7

TESTING

7. TESTING

7.1 Purpose of testing:

The purpose of testing is to ensure the reliability, accuracy, and efficiency of the system by identifying and fixing errors before deployment. It validates that all components, including the neural network model for accident severity classification, spatial network for safe routing, and web-based interface , function correctly and integrate seamlessly. Testing helps measure performance, security, and scalability, ensuring that the system can handle real-world data and user interactions efficiently. It also verifies that the predictions and recommendations align with actual accident patterns, enhancing road safety. By conducting thorough testing, the system becomes robust, user-friendly, and secure, minimizing failures and improving overall functionality.

7.2 Types of Testing

7.2.1 Unit Testing

Unit testing focuses on testing individual components or functions of the system in isolation. Each module, such as data preprocessing, model training, prediction, and safe route calculation, is tested separately to ensure it functions correctly. Frameworks like PyTest and Unit test are used to automate unit tests.

7.2.2 Integration Testing

Integration testing verifies that different modules work together as expected. It ensures seamless interaction between the Flask backend, neural network model, spatial routing system, and web interface. APIs are tested using Postman to validate data flow and responses.

7.2.3 Performance Testing

This type of testing evaluates the system's ability to handle large datasets and multiple user requests simultaneously. The neural network model's training and inference speed, response time of safe routing, and overall system efficiency are analyzed using tools .

7.2.4 Validation Testing

Validation testing ensures that the system meets the defined requirements and produces accurate results. The accident severity classification model is validated using metrics like accuracy, precision, recall, and F1-score , while the safe routing system is checked against real-world accident data to ensure reliability.

7.2.5 Security Testing

Security testing identifies vulnerabilities in the system, such as SQL injection, unauthorized access, and cross-site scripting (XSS) . API endpoints and user authentication mechanisms are tested using tools like OWASP ZAP to ensure data security and protection against cyber threats.

7.2.6 Regression Testing

Regression testing ensures that new updates or changes do not break existing functionalities. After modifying the neural network model, routing algorithm, or backend logic , previously tested components are re-evaluated to verify they still function as expected.

7.3 Test Strategy and Approach

The testing strategy involves a structured approach to verifying the functionality, performance, security, and usability of the system. Initially, unit testing is conducted to ensure that individual components like neural network classification, spatial routing, and data preprocessing work correctly. This is followed by integration testing , ensuring seamless interaction between the Flask backend, HTML/CSS frontend, and API endpoints . Performance testing is conducted to analyze system behavior under different workloads, while security testing ensures data protection and prevents vulnerabilities.

7.4 Test Objectives

The primary objectives of testing are:

- To verify the accuracy of the accident severity classification using neural networks.

- To ensure the spatial network for safe routing provides optimal and secure routes.

- To test the integration of datasets(accidents, vehicles, casualties) for correct preprocessing and model training.

- To validate the frontend and backend interaction to ensure seamless user experience.

- To evaluate the system's performance under heavy load conditions.

- To identify and fix security vulnerabilities like unauthorized access and SQL injection.

- To confirm the stability and reliability of the system after deployment.

7.5 Features to be Tested

The following key features of the system will be tested:

- **Neural Network Model** – Testing the training process, accuracy, and classification performance for accident severity prediction.
- **Spatial Network Routing** – Verifying the correctness of route suggestions based on accident-prone areas.
- **Data Preprocessing** – Ensuring that datasets (accidents, vehicles, casualties) are cleaned, formatted, and integrated correctly.
- **Frontend Functionality** – Testing user inputs, navigation, and result display using HTML, CSS, and JavaScript .
- **API and Backend Processing** – Ensuring smooth interaction between the Flask API and database.
- **Performance and Load Handling** – Testing how the system behaves under high traffic and large dataset processing.
- **Security and Authentication** – Validating protection against unauthorized access, data leaks, and potential cyber threats.
- **Deployment and Monitoring** – Ensuring the system remains stable and functional after deployment on AWS or Google Cloud .

CHAPTER 8

RESULTS

CHAPTER 8

RESULTS

User Interface:

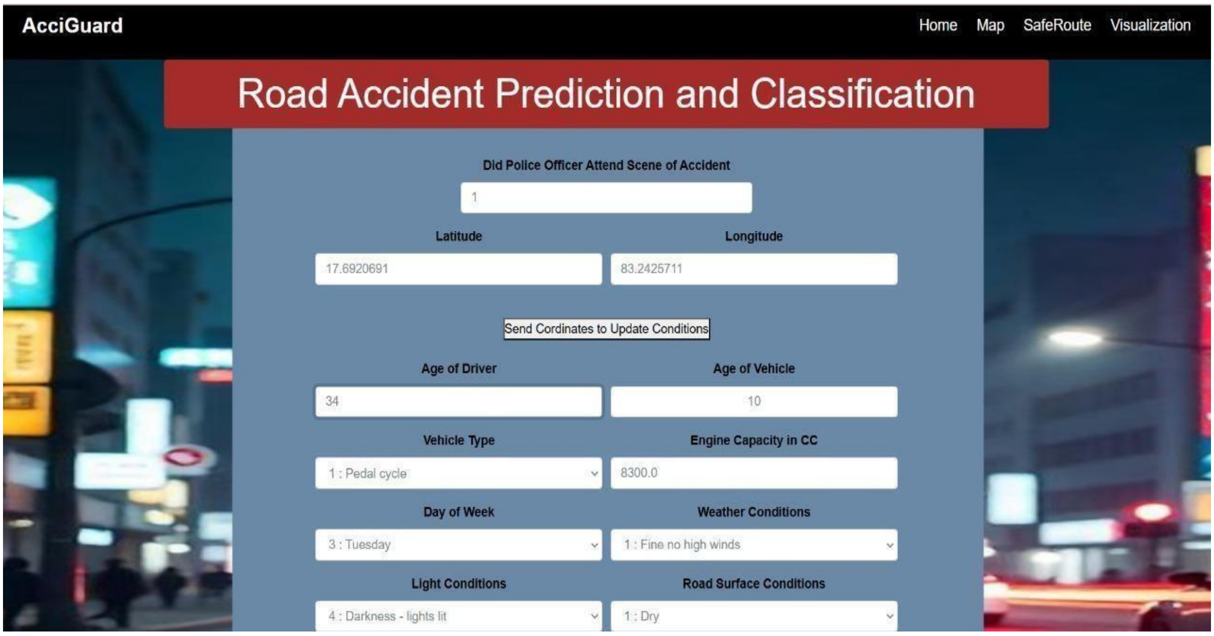


Fig 8.1

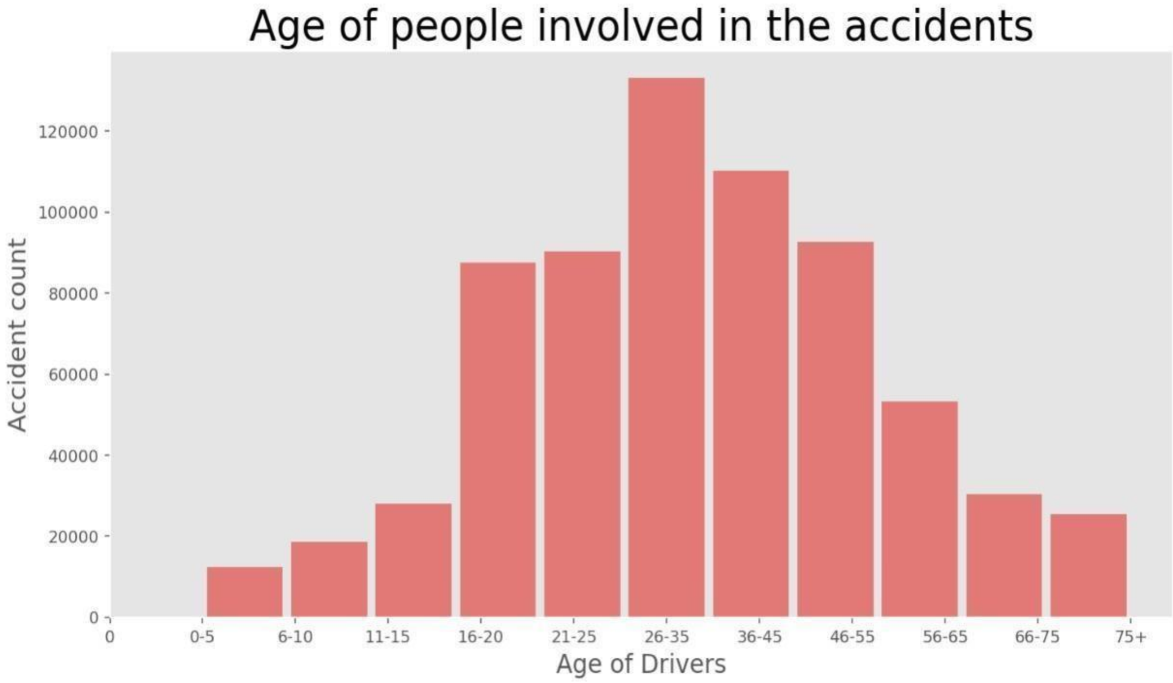
This screenshot taking the values from the user to predict the severity of the patient.

Predicting the output

This screenshot shows output predicted by the model.

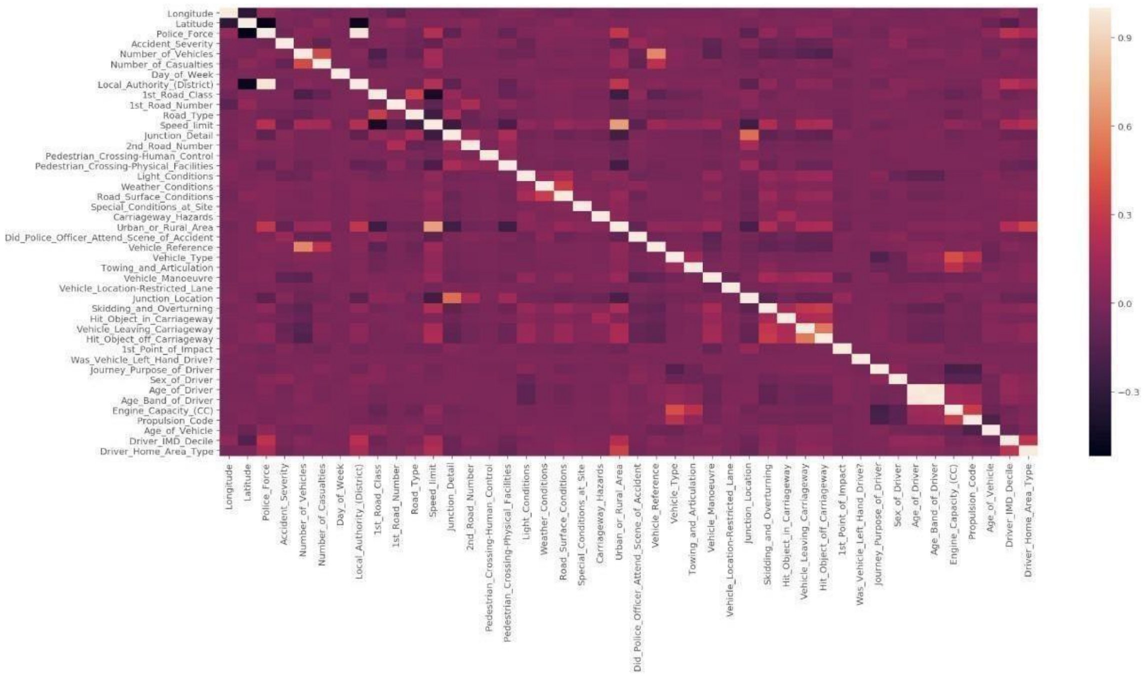
Fig.8.2

Visualized Result by the model



The above screenshot shows that how the age of the person influence the accident.

Fig.8.3



This Screenshot shows the factors those influence the accident

Fig.8.3

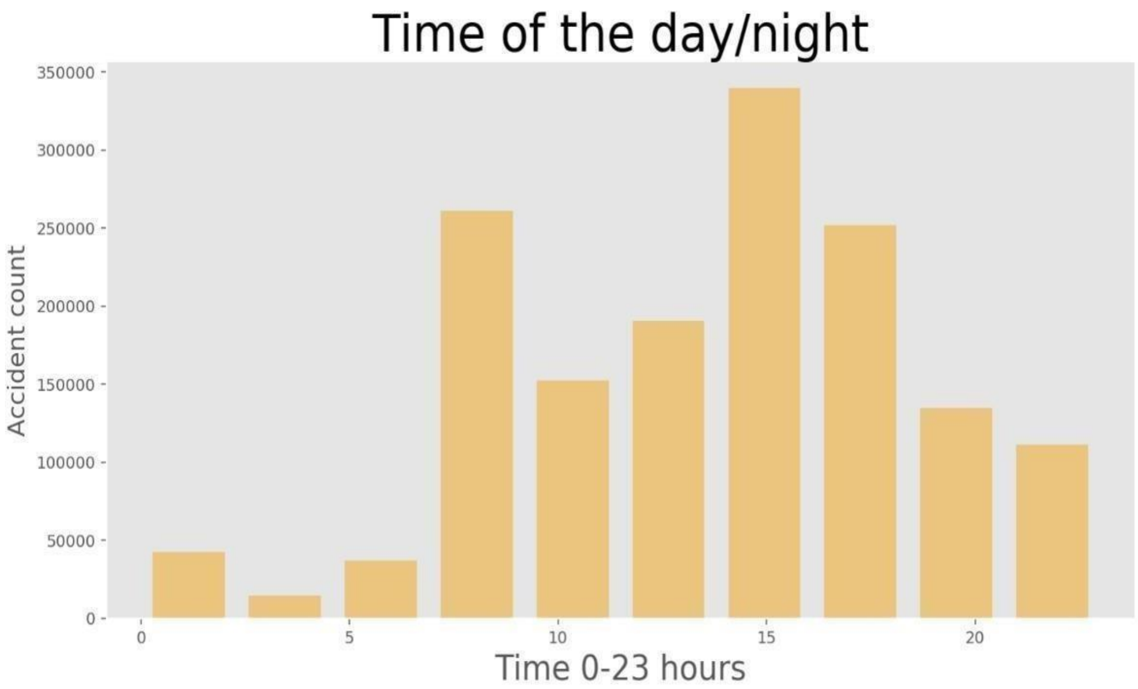
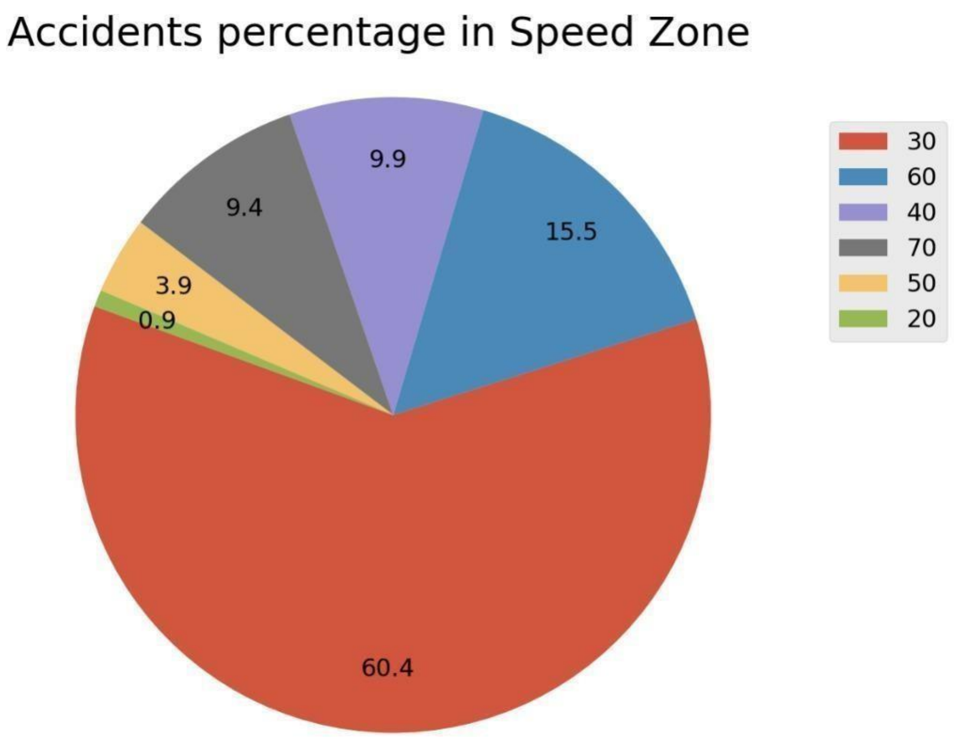


Fig.8.4



The above screenshot showing the accidents percentage in speed zone

Fig.8.5

Safe Routing

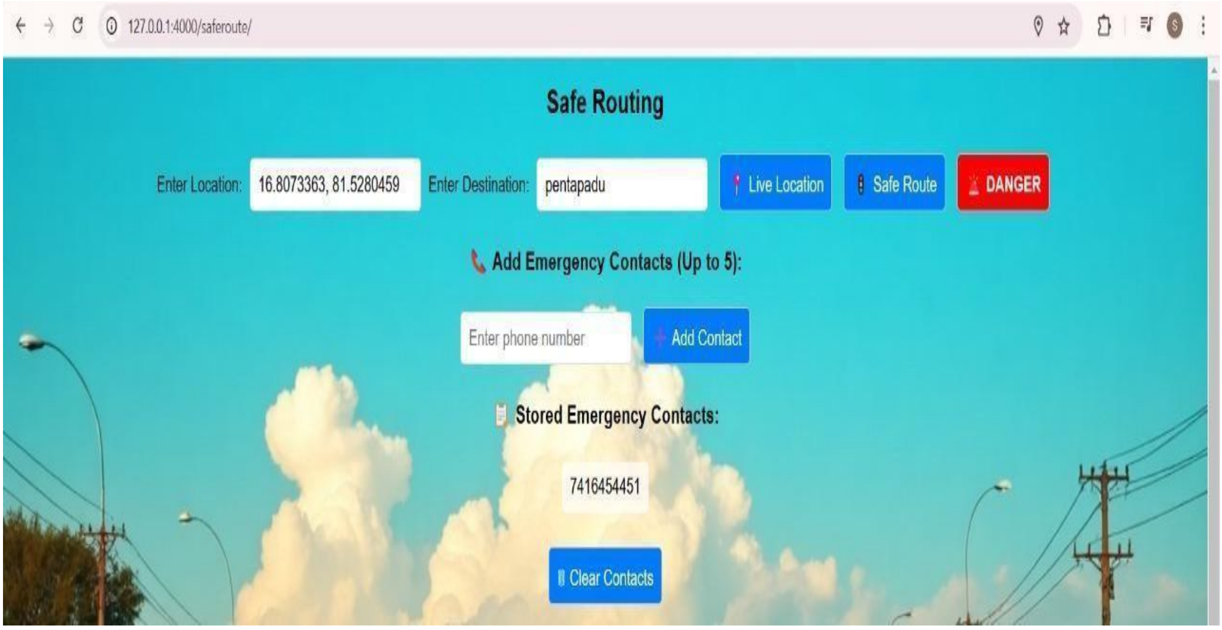


Fig.8.6

This screenshot shows the interface where the user enter the values.

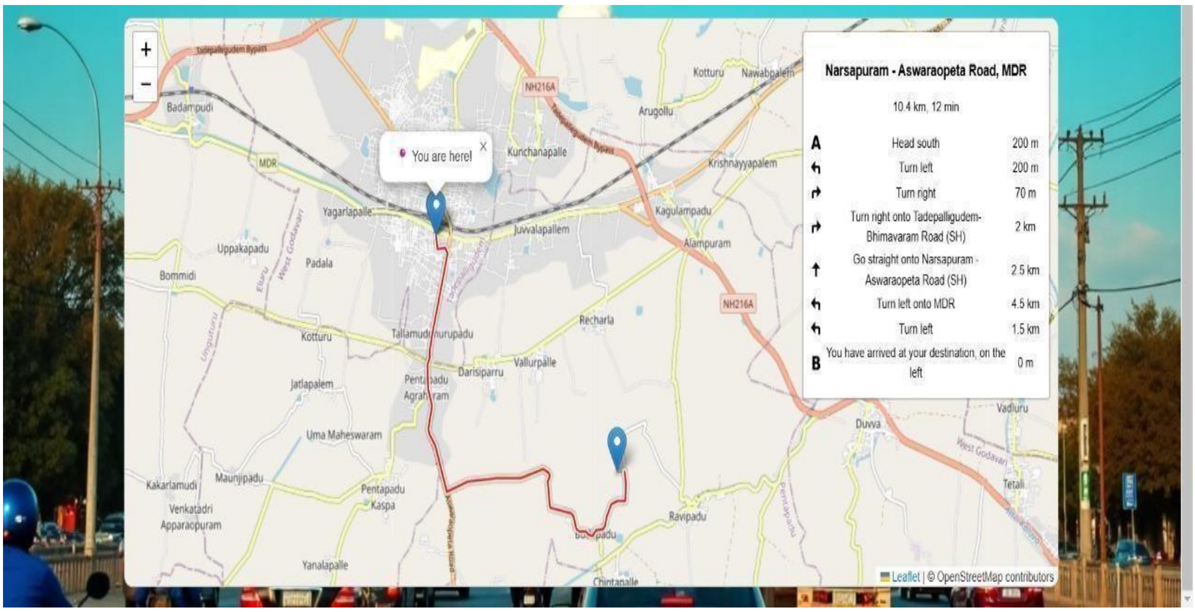


Fig.8.6

This screenshot shows safe route from source to destination.

CHAPTER 9

CONCLUSION AND FUTURE WORK

9.CONCLUSION AND FUTURE WORK

9.1 Conclusion

The implementation of a road accident severity classification system using neural networks and a spatial network for safe routing provides a data-driven approach to enhancing road safety. By preprocessing accident, vehicle, and casualty datasets, the model effectively classifies accident severity, helping authorities and users make informed decisions. The integration of Flask for backend processing, HTML/CSS for the user interface, and API-based spatial routing ensures a seamless and interactive experience. Comprehensive testing, including unit, integration, performance, and security testing , has validated the system’s accuracy, efficiency, and reliability. Overall, this project demonstrates the potential of AI and spatial networks in mitigating accident risks and improving transportation safety.

9.2 Future Work

- **Real-time Accident Prediction** – Integrate live traffic data, weather conditions, and sensor inputs for real-time severity analysis.

- **Enhanced Neural Network Model** – Utilize advanced deep learning techniques such as CNNs or LSTMs to improve classification accuracy.

- **Voice-Based Assistance** – Implement a voice-enabled feature to guide users with accident severity insights and safe route suggestions.

- **Multi-Language Support** – Expand the system to support multiple languages for better accessibility.

- **IoT-Based Vehicle Sensors** – Integrate IoT sensors in vehicles for real-time accident detection and automated reporting.

-**Mobile Application Development** – Develop a mobile version of the system to provide on-the-go access to accident severity classification and safe rout

CHAPTER 10

REFERNCES

10.REFERENCES

[1] Y. Gu, D. Liu, R. Arvin, A.J. Khattak, L.D. Han, Predicting intersection crash frequency using connected vehicle data: a framework for geographical random forest, Accid. Anal. Prev. 179 (2023) 106880. https://doi.org/10.1016/j.aap.2022.106880

[2] Y. Ali, F. Hussain, M.M. Haque, Advances, challenges, and future research needs in machine learning-based crash prediction models, Accid. Anal. Prev. 194 (2024) 107378. https://doi.org/10.1016/j.aap.2023.107378

[3] S. Ahmed, M.A. Hossain, S.K. Ray, M.M.I. Bhuiyan, S.R. Sabuj, A study on road accident prediction and contributing factors using explainable machine learning models, Transp. Res. Interdiscip. Perspect. 19 (2023) 100814. https://doi.org/10.1016/j.trip.2023.100814

[4] J. Wang, T. Luo, T. Fu, Crash prediction based on traffic platoon characteristics using floating car trajectory data and the machine learning approach, Accid. Anal. Prev. 133 (2019) 105320. https://doi.org/10.1016/j.aap.2019.105320

[5] M.F. Labib, A.S. Rifat, M.M. Hossain, A.K. Das, F. Nawrine, Road accident analysis and prediction of accident severity by using machine learning in Bangladesh, 2019 Int. Conf. Smart Comput. Commun. (ICSCC), pp. 1–5. https://doi.org/10.1109/ICSCC.2019.8843640

[6] P. Gorzelanczyk, Using neural networks to forecast the number of road accidents in Poland taking into account weather conditions, Results Eng. 17 (2023) 100981. https://doi.org/10.1016/j.rineng.2023.100981

[7] B.D.S.P. Amorim, A.A. Firmino, C.D.S. Baptista, G.B. Júnior, A.C.D. Paiva, F.E.D.A. Júnior, A machine learning approach for classifying road accident hotspots, ISPRS Int. J. Geo-Inf. 12 (6) (2023) 227. https://doi.org/10.3390/ijgi12060227

[8] M. Kanevski, A. Pozdnukhov, V. Timonin, Machine learning algorithms for geospatial data, Applications and Software Tools, Int. Congr. Environ. Model. Softw. (2008). https://scholarsarchive.byu.edu/iemssconference/2008/all/53

[9] S. Olugbade, S. Ojo, A.L. Imoize, J. Isabona, M.O. Alaba, A review of artificial intelligence and machine learning for incident detectors in road transport systems, Math. Comput. Appl. 27 (5) (2022) 77. https://doi.org/10.3390/mca27050077

[10] E.A. Atumo, T. Fang, X. Jiang, Spatial statistics and random forest approaches for traffic crash hot spot identification and prediction, Int. J. Inj. Control Saf. Promot. 29 (2) (2022) 207–216. https://doi.org/10.1080/17457300.2021.1983844

[11] Z. Pu, Z. Li, R. Ke, X. Hua, Y. Wang, Evaluating the nonlinear correlation between vertical curve features and crash frequency on highways using random forests, J. Transp. Eng. Part A: Syst.

146 (10) (2020). https://doi.org/10.1061/JTEPBS.0000402

[12] B.W. Al-Mistarehi, A.H. Alomari, R. Imam, M. Mashaqba, Using machine learning models to forecast severity level of traffic crashes by R studio and ArcGIS, Front. Built Environ. 8 (2022) 860805. https://doi.org/10.3389/fbuil.2022.860805

[13] T. Bokaba, W. Doorsamy, B.S. Paul, Comparative study of machine learning classifiers for modelling road traffic accidents, Appl. Sci. 12 (2022) 828. https://doi.org/10.3390/app12020828

[14] S.P. Ardakani, X. Liang, K.T. Mengistu, R.S. So, X. Wei, B. He, A. Cheshmehzangi, Road car accident prediction using a machine-learning-enabled data analysis, Sustainability 15 (7) (2023) 5939. https://doi.org/10.3390/su15075939

[15] S. Shafiei, Z. Gu, M. Saberi, Calibration and validation of a simulation-based dynamic traffic assignment model for a large-scale congested network, Simul. Model. Pract. Theory 86 (2018) 169–186. https://doi.org/10.1016/j.simpat.2018.04.006

[16] Kumar, R., Sharma, S., & Singh, V., Accident Severity Classification Using Machine Learning Techniques, Int. J. Comput. Appl. (2021).

[17] Singh, A., & Patel, M., Road Accident Severity Prediction Using Ensemble Machine Learning Models, J. Intell. Transp. Syst. (2022).

[18] Lee, J., & Park, S., A Graph-Based Safe Routing Model for Urban Traffic Management, Transp. Res. Part C Emerg. Technol. (2019).

[19] Lu Wenqi, Luo Dongyu, and Yan Menghua, A model for predicting traffic accident severity using Convolutional Neural Networks (CNN), Proc. Conf. Intell. Transp. Syst. (2017).

[20] Fu Huilin and Zhou Yucai, The Traffic Accident Prediction Based on Neural Network, Proc. Int. Conf. Smart City Emerg. Technol. (2017).

CHAPTER 11

SOURCE CODE

11. SOURCE CODE

11.1 Integrating with Flask

```
from flask import Flask, render_template, request, jsonify
import osmnx as ox
import networkx as nx
import folium
import joblib
import numpy as np
import urllib.request
import urllib.parse
from opencage.geocoder import OpenCageGeocode
app = Flask(__name__)
# Load ML Model
model = joblib.load('litemodel.sav')
# OpenCage API Key (Replace with your key)
API_KEY = "a51b7df648f24977a2f403014782bb4d"
geocoder = OpenCageGeocode(API_KEY)
# Load road network
city = "Bangalore, India"
graph = ox.graph_from_place(city, network_type="drive")
# Assign safety weights
for u, v, data in graph.edges(data=True):
    data["weight"] = data.get("length", 1)  # Default weight is road length
    if "fatalities" in data:
        data["weight"] *= (1 + data["fatalities"] * 0.1)  # Penalize dangerous roads
# 🔹 Function: Get Coordinates from Location Name
def get_coordinates(location_name):
    result = geocoder.geocode(location_name)
    if result:
        return result[0]['geometry']['lat'], result[0]['geometry']['lng']
    return None, None

# 🔹 Function: Send SMS Alerts
def sendSMS(apikey, numbers, sender, message):
    data = urllib.parse.urlencode({'apikey': apikey, 'numbers': numbers, 'message': message, 'sender':
sender})
    data = data.encode('utf-8')
    request = urllib.request.Request("https://api.textlocal.in/send/?")
    f = urllib.request.urlopen(request, data)
    return f.read()

# 🔹 Route Calculation API
@app.route('/safe_route', methods=['POST'])
def safe_route():
    data = request.json
    start_name = data.get("start")
    dest_name = data.get("destination")
```

```
    start_lat, start_lon = get_coordinates(start_name)
    dest_lat, dest_lon = get_coordinates(dest_name)

    if not start_lat or not dest_lat:
        return jsonify({"error": "Invalid locations"}), 400

    orig = ox.distance.nearest_nodes(graph, start_lon, start_lat)
    dest = ox.distance.nearest_nodes(graph, dest_lon, dest_lat)

    route = nx.shortest_path(graph, orig, dest, weight="weight")
    route_coords = [(graph.nodes[node]["y"], graph.nodes[node]["x"]) for node in route]

    return jsonify({"route": route_coords})

# 🔹 Accident Prediction API
def cal(ip):
    input = dict(ip)
    Did_Police_Officer_Attend = input['Did_Police_Officer_Attend'][0]
    age_of_driver = input['age_of_driver'][0]
    vehicle_type = input['vehicle_type'][0]
    age_of_vehicle = input['age_of_vehicle'][0]
    engine_cc = input['engine_cc'][0]
    day = input['day'][0]
    weather = input['weather'][0]
    light = input['light'][0]
    roadsc = input['roadsc'][0]
    gender = input['gender'][0]
    speedl = input['speedl'][0]

    data = np.array([Did_Police_Officer_Attend, age_of_driver, vehicle_type, age_of_vehicle,
engine_cc, day, weather, roadsc, light, gender, speedl])

    print("logging",data)
    data = data.astype(float)
    data = data.reshape(1, -1)

    try: result = model.predict(data)
    except Exception as e: result = str(e)

    return str(result[0])

# 🔹 SMS Alert API
@app.route('/sms_alert', methods=['POST'])
def sms_alert():
    try:
        response = sendSMS('UwYs16dD3zM-DKuzZKQYolAJkoba1j0BmRGompsNRs', '9618205648',
'TXTLCL', 'Severe accident detected!')
        return jsonify({"status": "Alert Sent!", "response": response.decode()})
    except Exception as e:
        return jsonify({"error": str(e)})

@app.route('/', methods=['GET'])
def index():
    return render_template('index.html')
```

```
@app.route('/', methods=['POST'])
def get():
    return cal(request.form)

@app.route('/visual/')
def visual():
    return render_template('visual.html')


@app.route('/export1/')
def export1():
    return render_template('export1.html')

@app.route('/saferoute/')
def saferoute():
    return render_template('saferoute.html')

# 🔹 Run Flask App
if __name__== '__main__':
    app.run(host='0.0.0.0', debug=True, port=4000)
```