**Phase-2 Submission Template Student Name:** JEBARAJ C

**Register Number:** 510623104037

**Institution:** C.ABDUL HAKEEM COLLEGE OF ENGINEERING AND TECHONOLOGY **Department:** COMPUTER SCIENCE AND ENGINEERING  
 **Date of Submission:** 08-05-2025

**Github Repository Link:**

<https://github.com/Jeba-c08/krayees001-Enhancing-road-safety-with-AI-driven-traffic-accident-analysis-and-prediction>

# Problem Statement

**Road accidents are a significant cause of fatalities and injuries worldwide. Many of these accidents are preventable with timely analysis and prediction. Traditional methods often lack real-time insights and predictive power**.  
  
 **This project aims to leverage AI techniques to analyze traffic accident data and predict high-risk zones and accident probabilities, thereby enhancing road safety and aiding decision-making for traffic authorities and urban planners.  
  
  
1.Rising Road Accidents:**  
Road traffic accidents are a major global issue, leading to significant loss of life, injuries, and property damage.

**2.Inefficiency of Traditional Methods:**  
Conventional approaches rely heavily on historical accident reports and static risk assessments, which are reactive and lack predictive capabilities.

**3.Fragmented Data Sources:**  
Critical traffic data is often scattered across different sources—such as police reports, GPS systems, weather services, and road sensors—making it difficult to analyze comprehensively.

**4.** **Lack of Real-Time Insights:**  
Current systems rarely provide real-time identification of accident-prone zones or immediate risk alerts to drivers and authorities.

**5.Underutilization of AI/ML:**  
Despite advancements in AI and machine learning, these technologies are not widely adopted in the domain of traffic safety prediction and proactive accident prevention.

# Project Objectives

1. **Collect real-world traffic accident data from reliable sources such as the UK Road Safety Open Data portal.**
2. **Clean and preprocess the dataset to ensure data quality and remove inconsistencies.**
3. **Perform analysis to identify patterns, trends, and hotspots of accident occurrence.**
4. **Engineer meaningful features from raw data that influence accident severity (e.g., time of day, road surface).**
5. **Train and evaluate machine learning models like Random Forest and XGBoost to predict accident likelihood and severity.**
6. **Fine-tune model parameters for improved performance using cross-validation and grid search techniques.**
7. **Simulate real-time scenarios to demonstrate how the model behaves with dynamic inputs.**
8. **Visualize results using charts, heatmaps, and graphs to communicate findings effectively.**
9. **Incorporate additional data sources, like weather conditions, for enhanced model accuracy.**
10. **Provide actionable insights that can inform policies and road safety improvements.**.

key points **Analyze Historical Accident Data  
 Examine past traffic accident records to identify recurring patterns, contributing factors, and high-risk scenarios.**

* **Build Predictive Models  
  Develop machine learning models to predict accident likelihood based on variables such as location, time, weather, and road conditions.**
* **Identify and Visualize High-Risk Zones  
  Use geospatial mapping to highlight accident-prone areas, aiding urban planners and traffic authorities in hotspot identification.**
* **Provide Actionable Insights  
  Generate meaningful insights that support data-driven decisions for implementing safety measures and infrastructure improvements.**
* **Develop a Real-Time Dashboard  
  Create an interactive web-based dashboard for monitoring predictions, visualizations, and traffic safety alerts in real time.**
* **Improve Road Safety through AI  
  Leverage artificial intelligence to shift from reactive to proactive accident prevention, ultimately reducing injuries and fatalities**

# Flowchart of the Project Workflow

* **Data Collection**
* Gather accident data from public sources, traffic APIs, and weather datasets.
* Include location, time, road type, weather, and vehicle data.
* **Data Preprocessing**
* Clean missing or inconsistent entries.
* Extract relevant features (e.g., time of day, speed limit zone).
* Normalize and encode data for model training.
* **Exploratory Data Analysis (EDA)**
* Identify patterns in accident occurrence.
* Visualize accident hotspots using charts and geospatial maps.
* **Feature Engineering**
* Create meaningful inputs like risk scores, peak traffic indicators.
* Reduce irrelevant features to improve model accuracy.
* **Model Development**
* Apply ML algorithms (e.g., Random Forest, XGBoost).
* Train models on labeled accident datasets.
* **Model Evaluation**
* Measure performance using metrics: accuracy, precision, recall, F1-score.
* Tune hyperparameters for optimal results.
* **Visualization & Dashboard**
* Create heatmaps of high-risk zones.
* Build a dashboard for users to view predictions interactively.
* **Deployment**
* Develop APIs to serve model predictions (Flask or FastAPI).
* Deploy on cloud (e.g., Heroku, AWS, or GCP) for real-time use.

# Data Description

**Source**: Kaggle Traffic Accident Datasets, US DOT, or local government datasets.  
  
 **Attributes may include**:  
  
**- Date and time of accident  
  
- Location (latitude, longitude, city/state)  
  
- Weather conditions  
  
- Road type and traffic signal information  
  
- Number and severity of injuries/fatalities  
  
- Vehicle types involved- Cause of accident**

**5) Data Preprocessing (Expanded)**

1. Handle Missing Values: Drop or impute nulls to maintain data integrity.
2. Time Feature Conversion: Extract hour/day/month from date-time fields.
3. Encode Categorical Data: Apply label or one-hot encoding as appropriate.
4. Normalization: Standardize numerical features where necessary.
5. Incident Filtering: Focus only on valid road transport incidents.
6. Column Cleanup: Remove columns with too many nulls or irrelevant info.
7. Outlier Detection: Identify and handle anomalies in the data.
8. Feature Merging: Combine similar features for clearer signals.
9. Target Separation: Isolate the label from input features.
10. Dataset Splitting: Use an 80:20 split for training and evaluation.

***6) Exploratory Data Analysis (Expanded)***

* ***Time Trends: Visualize accidents by hour, weekday, and month.***
* ***Severity Distribution: Plot how many accidents fall into each severity class.***
* ***Location Insights: Generate heatmaps of high-frequency accident areas.***
* ***Weather Correlation: Check how adverse weather affects accident severity.***
* ***Vehicle Type Analysis: Examine which vehicle categories are more accident-prone.***
* ***Road and Light Conditions: Investigate their impact on accident severity.***
* ***Weekday vs Weekend: Compare accident counts and severity.***
* ***Regional Distribution: Show accident rates across different areas.***
* ***Temporal Trends: Track accident data over multiple years.***
* ***Visualization Tools: Use Seaborn and Matplotlib for all plots.***

**7) Feature Engineering (Expanded)**

* Rush Hour Detection: Flag hours with increased traffic (e.g., 8–10 AM, 5–7 PM).
* Weather Risk Score: Quantify the severity of weather into a numerical score.
* Day Type: Differentiate between weekdays and weekends.
* Urban vs Rural: Add a binary indicator based on location type.
* Risk Aggregation: Group similar road surfaces by accident risk.
* Location Clustering: Optionally apply KMeans for hotspot detection.
* Interaction Features: Combine road and weather info for better insights.
* Encoding Techniques: Apply label encoding or one-hot based on model compatibility.
* Lag Features: Use past values if considering time-series modeling.
* Feature Pruning: Drop irrelevant or low-information features.  
    
    
  **8) Model Building (Expanded)**
* **Model Selection: Begin with Random Forest, XGBoost, and Logistic Regression.**
* **Training: Fit the model using the training dataset.**
* **Cross-Validation: Improve robustness and avoid overfitting.**
* **Hyperparameter Tuning: Use GridSearchCV to find optimal model parameters.**
* **Evaluation Metrics: Measure accuracy, precision, recall, and F1-score.**
* **Confusion Matrix: Visualize performance for each class.**
* **Prediction: Generate predictions on the test set.**
* **Feature Importance: Identify which features impact severity prediction.**
* **Ensemble Learning: Combine models to boost accuracy (if needed).**
* **Simulations: Test how well the model generalizes to new data.**  
    
    
    
    
    
  **9) Visualization of Results and Model Insights (Expanded)**
* **Confusion Matrix: Heatmap showing prediction performance across classes.**
* **Feature Importances: Horizontal bar chart of top predictors.**
* **Severity Pie Chart: Visual distribution of predicted classes.**
* **Accident Maps: Plot accident density by GPS coordinates.**
* **Time Series: Track accident counts over time.**
* **Prediction Comparison: Actual vs predicted outcomes.**
* **Regional Risk Map: Visual heatmaps by geography.**
* **Correlation Matrix: Show interdependencies between features.**
* **EDA Charts: Use bar plots, histograms, boxplots to analyze trends.**
* **Report Output: Save visualizations and results as CSV or HTML.**  
    
  **10)Tools and Technologies Used (Expanded)**
* Google Colab – Cloud-based development and testing platform.
* Python 3.8+ – Main programming language for data science.
* Pandas – Library for data manipulation and cleaning.
* NumPy – For efficient numerical computation.
* Seaborn & Matplotlib – For creating professional visualizations.
* Scikit-learn – Core library for building and evaluating ML models.
* XGBoost – Gradient boosting framework for powerful tree-based models.
* KMeans (optional) – Unsupervised clustering for hotspot detection.
* OpenCV or PIL – For optional image processing (e.g., visual mapping).
* CSV Files – Standard format for dataset storage and sharing.  
    
    
    
    
  **11) Team Members and Roles:**  
    
   This project was collaboratively developed by a dedicated team of five members. Each team member was assigned specific roles and responsibilities based on their individual strengths and interests, ensuring a smooth and efficient workflow throughout the project.  
    
    
   **1. MOHAMED RAYEES.K**  
   **Role:** Team Lead & Model Developer  
   **Responsibilities:**   
    
   • Led the overall project planning and execution.  
   • Designed and implemented the core Convolutional Neural Network (CNN)  
   • Conducted hyperparameter tuning and model optimization.  
   • Coordinated meetings and integration tasks among all team members  
    
   **2. LOGITH .S.T**  
   **Role:** Data Engineer & Preprocessing Specialist   
  **Responsibilities:**  
   • Handled dataset acquisition and formatting.  
   • Performed image normalization, reshaping, and augmentation.   
  • Ensured data quality and consistency across training and testing phases.  
   • Assisted in EDA (Exploratory Data Analysis) and dataset visualization.  
    
  **3. JAMAL.S**   
  **Role:** Visualization & Evaluation Analyst   
  **Responsibilities:**  
   • Created training vs. validation accuracy/loss plots  
   • Built and interpreted confusion matrices.  
   • Performed statistical analysis of performance metrics like precision, recall, and F1-score.  
   • Helped assess model robustness and performance.   
    
   **4. DHUSHYANDH.N**  
   **Role:** UI/UX & Deployment Developer (Optional Streamlit Interface) **Responsibilities:**  
    
   • Developed an interactive web-based interface using Streamlit for real-time digit prediction.  
  • Integrated the trained model into the user interface.   
  • Ensured usability and responsiveness of the application.  
    
   **5. JEBARAJ.C**  
   **Role:** Documentation & Report Writer

**Responsibilities:**  
  
• Compiled and wrote detailed sections for the project report (problem statement, methodology, results, etc.).  
• Handled citation formatting and references.   
• Prepared visual content (charts, diagrams, sample images) for documentation.   
• Managed the final submission materials (PDF/DOCX report formatti

# 