AutoWake: Driver Safety System

A Minor Project Synopsis Submitted to Rajiv Gandhi Proudyogiki Vishwavidyalaya

Towards Partial Fulfillment for the Award of Bachelor of Technology in Computer Science and Engineering

Submitted By
Maniya Jeswani(0827CS221157)
Priyal Agrawal (0827CS221198)
Priyansh Rai (0827CS221202)
Punit Sankhala (0827CS221209)

Guided by:
Prof. Krupi Saraf

Department of Computer Science and Engineering Acropolis Institute of Technology & Research, Indore
Jan-June 2025

1.Introduction

Drowsy driving is a serious issue, causing nearly 30% of road accidents. Fatigue slows reaction time and impairs judgment, leading to fatal crashes. Existing drowsiness detection methods are often unreliable, making roads more dangerous.

Our project aims to solve this problem with an advanced and accurate drowsiness detection system. Using cutting-edge technology, we will monitor driver alertness in real time and issue warnings before an accident happens. This solution will help reduce road accidents, save lives, and make driving safer for everyone.

2.Objective

- Develop a smart system that detects driver drowsiness in real time.
- Prevent road accidents by alerting drowsy drivers before a crash happens.
- Use advanced technology to ensure accurate and reliable detection.
- Analyse driver behaviour, such as eye movement, head position, and reaction time.
- Make roads safer by reducing fatigue-related accidents.

3.Scope

This project focuses on developing an advanced drowsiness detection system to reduce fatigue-related road accidents. It is designed for drivers, transportation companies, and vehicle manufacturers who need a reliable way to detect drowsiness in real time.

In its initial version, the system will monitor key indicators such as eye movement, head position, and reaction time to detect signs of fatigue. It will provide instant alerts to prevent potential crashes. For now, the system will focus on real-time detection and alert generation, without features like accident prediction or integration with vehicle control systems.

In the future, additional features like AI-based fatigue prediction, integration with smart vehicles, and cloud-based monitoring can be added
to enhance safety and effectiveness

4.Study Of Existing System

1. Eye-Tracking-Based Detection Systems

- **Problems Addressed:** Monitors eye movements and blinks to detect drowsiness.
- **Advantages:** Non-intrusive, can provide early warnings.
- **Disadvantages:** May not work well in low-light conditions or if the driver wears glasses.
- **Gaps Identified:** Lacks accuracy in certain conditions, prone to false alarms.
- **Reference Link:** Real-time eye tracking for the assessment of driver fatigue

2. **Wearable Drowsiness Detection Devices**

- **Problems Addressed:** Uses smart headbands or glasses to track brain activity and fatigue levels.
- **Advantages:** Directly measures brain signals, providing high accuracy.
- **Disadvantages:** Expensive, uncomfortable for long-term use.
- **Gaps Identified:** Not practical for daily driving, limited adoption due to cost and inconvenience.

- **Reference Link:** Fatigue Monitoring Through Wearables: A State-of-the-Art Review

3. **Steering Behaviour Monitoring Systems**
   - **Problems Addressed:** Analyses erratic steering patterns as an indicator of drowsiness.
   - **Advantages:** Non-intrusive, does not require additional hardware.
   - **Disadvantages:** May not detect early signs of drowsiness, as sudden corrections can be caused by other factors.
   - **Gaps Identified:** Not reliable for proactive drowsiness detection, only detects when fatigue has already affected driving.
   - **Reference Link:** Technologies for detecting and monitoring drivers' states

4. **Smartphone-Based Detection Apps**
   - **Problems Addressed:** Uses phone cameras and sensors to analyze facial features and head movements.
   - **Advantages:** Easily accessible, does not require specialized hardware.
   - **Disadvantages:** Relies on phone placement, accuracy depends on lighting and camera quality.
   - **Gaps Identified:** Not a standalone solution, requires the driver to keep their phone positioned correctly.
   - **Reference Link:** Real-Time Drowsiness Detection Using Eye Aspect Ratio and Facial Landmark Detection

5.Project Description

1. **User Input**: The user enters car details like brand, model, manufacturing year, fuel type, transmission type, kilometers driven, etc.
2. **Data Preprocessing**: The input data is cleaned and converted into a format that the machine learning model can understand.
3. **Prediction Model**: A trained machine learning model (such as Linear Regression or Random Forest) processes the data.
4. **Price Prediction**: The model calculates and returns an estimated price of the car.
5. **Result Display**: The predicted car price is shown on the screen in a simple and understandable way.
6. **Optional Feedback**: Users may give feedback to improve prediction accuracy in future versions.

Flowchart:

The flow of information in the project can be illustrated as follows:

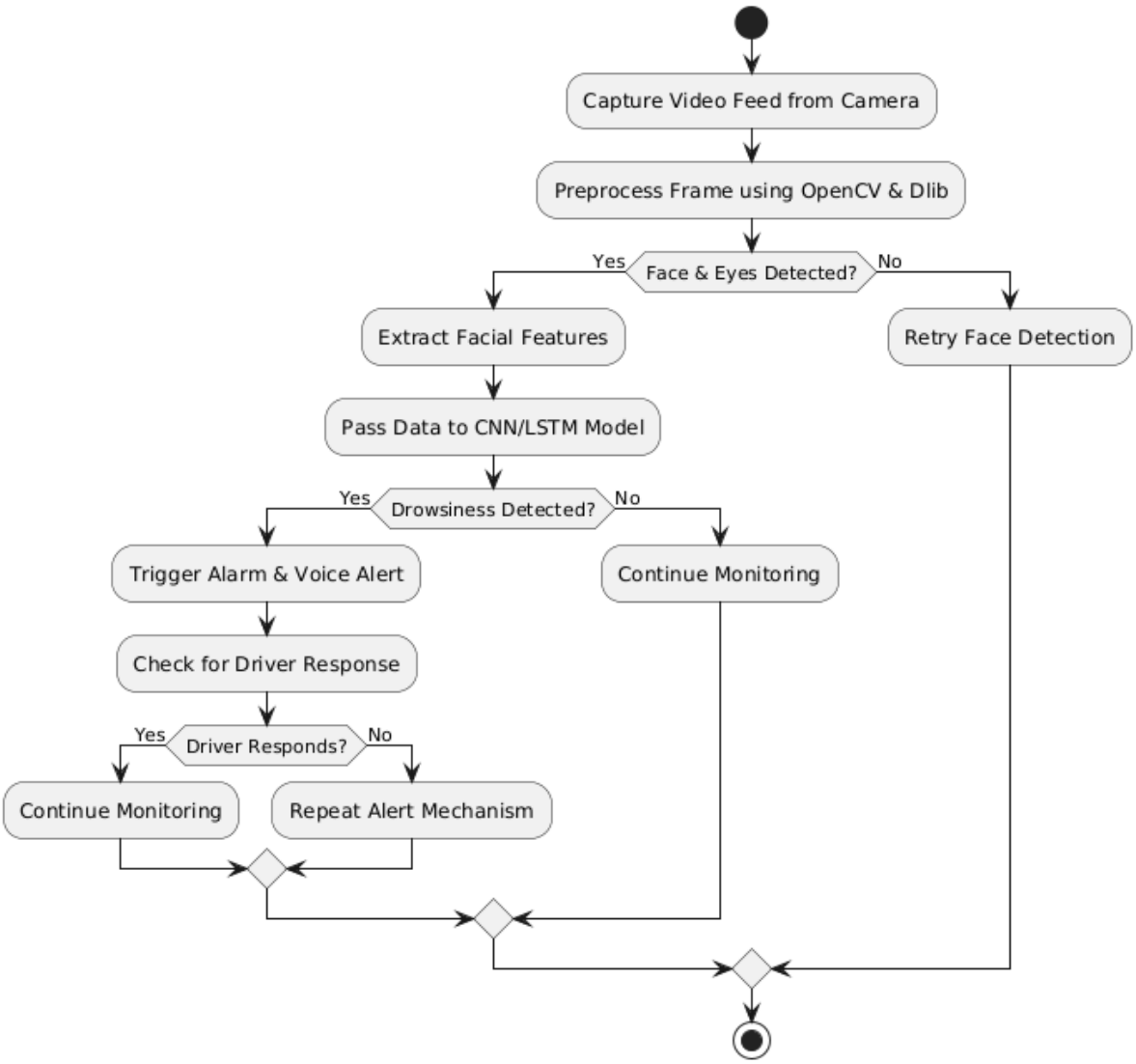

Figure: Flow Chart of System

- **ER Diagram**

The ER (Entity-Relationship) diagram defines the core structure and interactions between various entities in the Drowsiness Detection System. It highlights how data flows between the driver, monitoring system, detection model, and alert mechanism. Below is a brief description of each entity and its role in the system:

1. Driver

This entity represents the individual operating the vehicle, whose drowsiness levels are

monitored.

Attributes:

- driverID (unique identifier)
- name
- age
- licenseNumber

Each driver is monitored in real-time by the detection system, and alerts are generated when signs of fatigue are detected.

2. Vehicle

This entity contains details of the vehicle equipped with the drowsiness detection system.

Attributes:

- vehicleID (unique identifier)
- make
- model
- year
- registrationNumber

Each vehicle is linked to a driver and is equipped with monitoring tools.

3. Monitoring System

This entity represents the hardware and software that capture and analyse driver behaviour.

Attributes:

- systemID (unique identifier)
- sensorType (e.g., camera-based, wearable, steering behavior analysis)
- installationDate
- version

The monitoring system collects real-time data and passes it to the detection model.

4. Drowsiness Detection Model

This entity processes the captured data and determines if the driver is drowsy.

Attributes:

- modelID (unique identifier)
- modelType (e.g., CNN-based, machine learning, threshold-based)
- accuracyLevel
- version

The model analyses driver behavior and provides an output that determines whether an alert should be triggered.

5. Alert System

This entity handles the warnings sent to the driver when drowsiness is detected.

Attributes:

- alertID (unique identifier)
- alertType (sound, vibration, visual notification)
- timestamp
- severityLevel

The alert system ensures timely warnings to prevent potential accidents

Relationships:

- **Drivers** operate **Vehicles** that are equipped with a **Monitoring System**.
- The **Monitoring System** collects data and sends it to the **Drowsiness Detection Model** for analysis.
- The **Drowsiness Detection Model** determines if fatigue is detected and, if necessary, triggers the **Alert System**.
- The **Alert System** warns the **Driver** in real-time to prevent accidents.

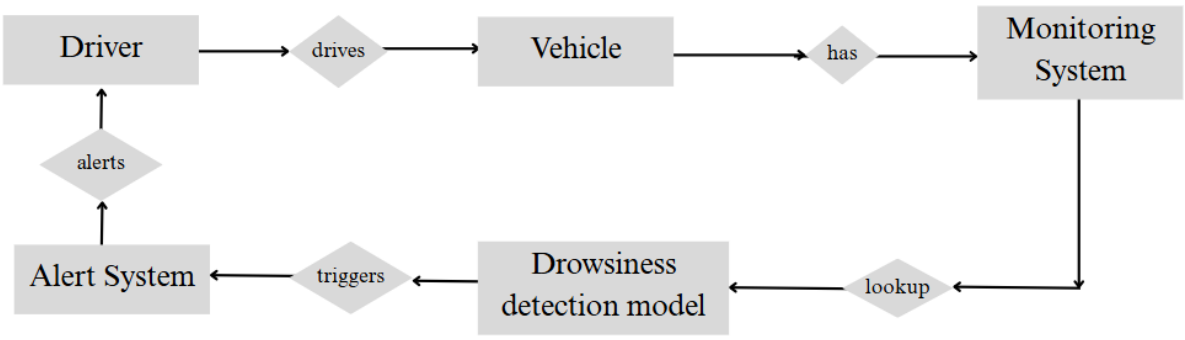

Figure: ER Diagram

6.Methodology

The methodology adopted in this project for detecting drowsiness in drivers involves the following key steps:

1. Data Collection

The dataset used in this project consists of real-time facial expression data, eye movement patterns, head position, and yawning frequency. This data is collected using:

- Camera-based monitoring systems (dashboard cameras, infrared sensors)
- Wearable sensors (EEG headbands, heart rate monitors)
- Steering behavior analysis (pressure and movement tracking)

Public datasets such as NTHU Drowsy Driver Detection Dataset and YawDD (Yawning Detection Dataset) were also used to train the AI model.

2. Data Preprocessing

To improve the accuracy of drowsiness detection, the following preprocessing techniques were applied:

- Face and Eye Detection: Extracted facial landmarks using OpenCV and Dlib.
- Data Cleaning: Removed blurred frames and irrelevant data points.
- Normalization: Scaled input values to maintain consistency in detection.
- Feature Extraction:
    - Blink rate analysis
    - Eye aspect ratio (EAR)
    - Mouth opening ratio (yawing detection)
    - Head pose estimation

3. Feature Selection

To determine the most critical indicators of drowsiness, we analysed the importance of various features such as:

- Blink duration (longer blinks indicate fatigue)
- Head tilting (nodding off suggests drowsiness)
- Yawning frequency (repeated yawns are a key sign of fatigue)

The Random Forest Classifier was used to rank the features based on their contribution to the final detection model.

4. Data Splitting

The dataset was divided into:

- Training Set (80%) – Used to train the deep learning model.
- Testing Set (20%) – Used to evaluate real-time performance.

This ensures the model generalizes well to new driver behaviours.

5. Model Building

In this study, two distinct models were developed for detecting drowsiness. The first model utilized the **Eye Aspect Ratio (EAR)**, computed from facial landmarks using **OpenCV**, to identify eye closure and blinking patterns. This rule-based method proved effective for lightweight and real-time monitoring. The second model was built using a **Convolutional Neural Network (CNN)**, which processed facial images to automatically extract features and classify drowsy vs. non-drowsy states. Both models were evaluated independently, with the EAR model offering faster performance, while the CNN model provided higher accuracy in more complex scenarios.

6. Model Evaluation

The model was evaluated using:

- Accuracy – Percentage of correctly detected drowsy/non-drowsy states.
- Precision & Recall – To ensure minimal false alarms and missed detections.
- F1-Score – Balances precision and recall for overall effectiveness.

A confusion matrix was plotted to analyze misclassifications.

7. Real-Time Implementation & Alerts

The final model was integrated with a real-time detection system using:

- Raspberry Pi & OpenCV for edge computing.
- Audio and Haptic Alerts triggered when drowsiness is detected.
- Mobile App Connectivity to notify drivers or fleet managers.

This ensures immediate intervention to prevent accidents caused by fatigue.

PHASES	TIMELINE	TASK COMPLETED
Requirements Gathering	Week 1	Identified project objectives, researched accident trends, analyzed drowsiness-related crashes, and defined system requirements.
Data Collection	Week 2	Gathered real-world datasets, including eye movement patterns, blink frequency, yawning detection, and head positioning.
Data Preprocessing	Week 3	Cleaned data, handled missing values, normalized inputs, and selected key fatigue indicators for analysis.
Model Selection& Design	Week 4	Evaluated AI models (CNN, EAR, OpenCV-based facial recognition) and finalized the best approach for real-time drowsiness detection.
Model Training	Week 5	Trained the AI model with extensive datasets of alert and drowsy drivers, fine-tuning for accuracy and performance.
Model Evaluation	Week 6	Assessed system accuracy using precision, recall, and confusion matrix to minimize false positives and negatives.
Integration & Development	Week 7-8	Integrated the trained AI model with a real-time camera feed, alert mechanism (audio/visual), and dashboard interface.
Testing	Week 9	Conducted real-world testing in different lighting conditions, vehicle speeds, and fatigue levels to enhance robustness.
Deployment	Week 10	Deployed the system on cloud/local devices, ensuring it runs seamlessly with minimal latency.
Maintenance	Week 11	Collected user feedback, monitored real-time performance, and implemented improvements to enhance accuracy and responsiveness.

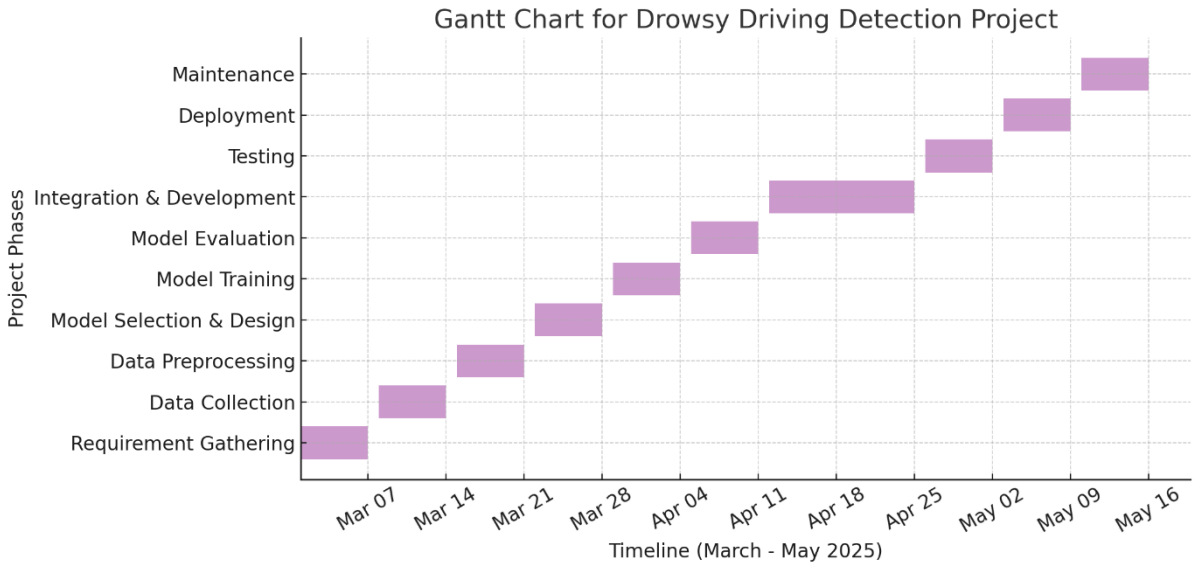

Figure: Gantt Chart

7.Features

Here are the main features that make this project stand out:

1. **Real-Time Driver Monitoring** – The system continuously tracks the driver's face and eye movements using a camera and AI-based algorithms to detect drowsiness.
2. **Fatigue Detection Alerts** – If signs of drowsiness are detected, the system triggers alerts such as loud alarms, seat vibrations, or voice warnings to wake up the driver.
3. **AI-Based Behavior Analysis** – The model analyses blinking rate, yawning frequency, and head position to determine fatigue levels accurately.
4. **Non-Intrusive & Easy Integration** – Works with standard dashboard cameras, making it easy to implement in existing vehicles without complex modifications.
5. **Emergency Assistance Activation** – If the driver remains unresponsive, the system can send an emergency alert to predefined contacts with location details.
6. **User-Friendly Interface** – Designed with a simple dashboard where users can access real-time drowsiness alerts and system performance reports.

8.System Architecture

The system architecture of this project consists of two main components that work together to detect driver drowsiness and issue timely alerts.

1. Frontend (User Interface):

- **What it does:**
  This is the part of the system that the user interacts with. It provides a simple dashboard displaying real-time driver monitoring and drowsiness detection status.
- **How it works:**

The frontend collects video feed from the driver-facing camera and sends it to the backend for analysis. If drowsiness is detected, it triggers alerts like alarms or voice notifications.

2. Backend (Processing Engine):

- **What it does:**
  The backend processes the live video feed, analyzes facial expressions, and determines signs of drowsiness using AI models.
- **How it works:**
  - **Python:** The core programming language for backend development.
  - **OpenCV & Dlib:** Used for real-time face and eye detection.
  - **Deep Learning (CNN/LSTM):** Processes facial features and predicts drowsiness levels based on eye closure, yawning, and head position.
  - **Google Text-to-Speech (gTTS) or pyttsx3:** Generates voice alerts to warn the driver.
  - **SpeechRecognition or Vosk API:** Enables voice commands from the driver for system interaction.

How They Work Together:

1. **Video Capture:**
   - The driver-facing camera records live footage and sends it to the backend.
2. **Drowsiness Detection:**
   - The backend processes the frames using OpenCV and AI models to check for signs of fatigue.
3. **Alert Triggering:**
   - If drowsiness is detected, the system sends a warning signal via voice output, alarm, or other means.
4. **Voice Input (Optional):**
   - The driver can interact with the system using voice commands for hands-free operation.

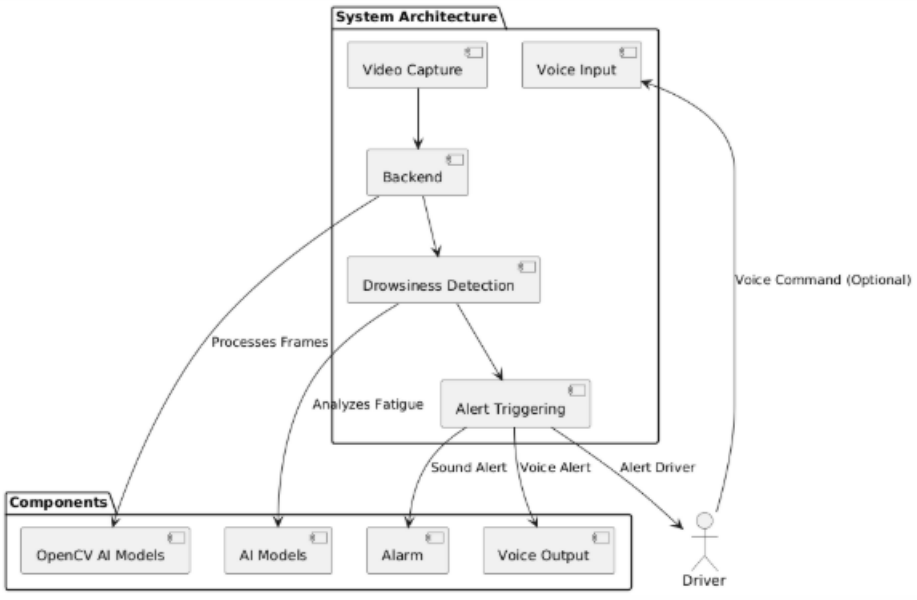

Figure: System Architecture

9.User Interface (UI)

The **User Interface (UI)** of the drowsiness detection system is designed to be simple and user-friendly, ensuring that drivers can interact with the system effortlessly while staying focused on the road.

1. Dashboard (Homepage):

- The homepage welcomes users with a clean, distraction-free design.
- A **"Start Monitoring"** button is placed at the center, making it easy to begin real-time drowsiness detection.
- A **minimal top navigation bar** provides access to settings, reports, and help sections.

2. Live Monitoring Page:

- Displays a **real-time video feed** from the driver's camera.
- Shows **eye-tracking and facial landmark points** to indicate active monitoring.
- If drowsiness signs (closed eyes, yawning, head tilting) are detected, the system:
  - Flashes a **warning message** (e.g., "Wake Up! You are Drowsy!")
  - Triggers an **alarm or voice alert** using Google Text-to-Speech (gTTS) or pyttsx3.
  - Logs the drowsiness event for later analysis.

3. Alerts & Reports Page:

- Displays **past drowsiness incidents** recorded by the system.
- Shows statistics like:
  - **Number of drowsiness alerts per trip**
  - **Time and duration of detected fatigue episodes**
  - **Severity level** of drowsiness (Mild, Moderate, Severe)
- Provides a **"Download Report"** option to save the data as a PDF for review.

4. Voice Interaction & Mobile-Friendly Design:

- Drivers can interact with the system hands-free using voice commands via **SpeechRecognition or Vosk API** (e.g., "Start Monitoring" or "Stop Alert").
- The UI is **optimized for mobile devices**, ensuring seamless functionality for drivers using smartphones or dashboard-mounted tablets.

10.Technology Stack

To build an **AI-powered drowsiness detection system**, we are using a combination of **computer vision, deep learning, and speech technologies**. Here's a breakdown of the tech stack:

1. Frontend (User Interface):

- **Electron JS –** For building cross-platform desktop applications.
- **React JS + Vite –** For fast, reactive, and modular UI development.
- **Bootstrap / React-Bootstrap** – For responsive design and UI components.
- **HTML5, CSS3, JavaScript –** Core web technologies for structure, styling, and interactivity.

2. Backend (Processing & Logic):

- **Flask** – Lightweight Python-based web framework for handling API and server-side logic.
- **Flask-CORS** – Enables secure Cross-Origin Resource Sharing between frontend and backend.

3. Video & ML Processing:

- **OpenCV –** For real-time video capture and facial feature detection.
- **dlib –** For facial landmark detection and alignment.
- **NumPy –** For numerical operations and matrix manipulations.

4. Machine Learning Models:

- **CNN (Convolutional Neural Network) –** For classifying drowsy vs. non-drowsy states based on face images.
- **EAR (Eye Aspect Ratio)** – Rule-based model for detecting eye closure and blinks using facial landmarks.

11.Testing Plan

To ensure our **Drowsiness Detection System** works accurately and effectively, we will follow a structured testing approach to validate **real-time detection, alert accuracy, and system performance**.

1. Unit Testing

🔹 **What it is:** Testing individual components separately—such as **face detection, eye tracking, yawning detection, speech alerts, and voice commands**—to ensure they work correctly.

🔹 **Why it's important:** Helps detect and fix **bugs in small units** before combining them into the full system.

2. Integration Testing

🔹 **What it is:** Testing how different modules interact—ensuring **face detection, AI prediction, and voice alerts** work together smoothly.

🔹 **Why it's important:** Verifies that **data flows correctly** between different components, avoiding false positives or missed detections.

3. Real-World Testing

🔹 **What it is:** Testing the system in **actual driving conditions** under different scenarios—day/night driving, different lighting, and varied driver behaviors.

🔹 **Why it's important:** Ensures the system **performs accurately in real-life conditions**, not just in lab environments.

4. Performance Testing

🔹 **What it is:** Measuring system speed and efficiency to ensure **real-time drowsiness detection without lag**.

🔹 **Why it's important:** Delayed alerts can make the system useless in critical situations.

5. False Alarm & Accuracy Testing

🔹 **What it is:** Evaluating system accuracy by testing **false positives (wrong detections) and false negatives (missed detections)**.

🔹 **Why it's important:** Ensures the system **only alerts when necessary** and **doesn't miss real cases of drowsiness**.

6. Security & Privacy Testing

🔹 **What it is:** Ensuring **user data (face scans, voice inputs) is secure** and the system does not allow unauthorized access to the camera/microphone.

🔹 **Why it's important:** Protects **user privacy and prevents hacking**.

7. Bug Fixing & Retesting

◈ **What it is:** After fixing detected issues, the entire system will be **retested** to confirm all functions work properly.

◈ **Why it's important:** Ensures fixes **don't introduce new bugs** and that the system remains **stable and reliable**.

12.Expected Outcome

Our **Drowsiness Detection System** aims to provide a **real-time, AI-powered solution** to prevent accidents caused by driver fatigue. Below are the key expected outcomes:

1. Accurate Drowsiness Detection

◈ The system will **precisely detect drowsiness** using **facial and eye-tracking** techniques.

◈ Deep learning models (CNN/EAR) will **analyze blinking patterns, yawning frequency, and head movements** to determine fatigue levels.

2. Instant & Effective Alerts

◈ When drowsiness is detected, the system will trigger **real-time alerts** using **audio warnings (Google TTS/Pyttsx3)** to wake up the driver.

◈ The **alert intensity** may increase based on the driver's fatigue level to **prevent accidents before they happen**.

3. Hands-Free Voice Commands

◈ The system will integrate **voice recognition (SpeechRecognition/Vosk)** to allow drivers to **interact without distraction**.

◈ Drivers can use simple voice commands to **confirm wakefulness or dismiss false alarms**.

4. Real-Time Performance with Low Latency

◈ Built with **OpenCV and Dlib**, the system will process video frames in real-time with **minimal lag**.

◈ Optimized for **fast response times**, ensuring that alerts are **immediate and effective**.

5. Works in Different Conditions

◈ The system will function **day and night**, adapting to **varying light conditions**.

◈ It will work **regardless of eyewear, facial hair, or head positioning**, making it **versatile for all drivers**.

6. User-Friendly & Non-Intrusive

◈ No need for **wearable devices**—just a **camera-based system** for effortless use.

◈ Can be **easily integrated** into vehicles with a **simple dashboard-mounted camera**.

7. Privacy-Focused & Secure

◈ No unnecessary data storage—**face/video feeds won't be saved** to ensure user privacy.

◈ Runs **locally on the device**, eliminating security risks from cloud-based processing.

8. Scalable & Customizable

◈ Can be enhanced with **IoT integration (e.g., automatic seat vibrations for alerts)**.

◈ Future updates could **connect with vehicle systems** to trigger **speed reduction or hazard lights** when drowsiness is detected.

13.Resources And Limitation

Developing the **Drowsiness Detection System** requires both hardware and software resources to ensure accurate real-time monitoring of driver fatigue. A **computer or laptop** is essential for coding, training machine learning models, and running the system. Additionally, a **high-resolution webcam or infrared (IR) camera** is needed to capture the driver's facial expressions and eye movements. Since voice-based alerts and commands are part of the system, a **microphone** is also necessary to process driver responses.

On the software side, **Python** serves as the primary programming language, while **OpenCV and Dlib** handle facial and eye detection. The system employs **deep learning models, particularly CNN and EAR**, to recognize drowsiness patterns accurately. To provide real-time alerts, **Google Text-to-Speech (gTTS) or Pyttsx3** is used for generating voice warnings, ensuring the driver is immediately notified. Additionally, **SpeechRecognition or the Vosk API** enables voice input, allowing the driver to respond hands-free. The entire development and testing process is carried out using **Jupyter Notebook or Visual Studio Code**, which facilitate model training and debugging. The system primarily runs locally but can be extended for deployment on **embedded devices like Raspberry Pi** or integrated into smart car systems in future updates.

Despite its effectiveness, the project has certain **limitations**. One major challenge is **accuracy in varying conditions**—low lighting, sunglasses, or facial obstructions can interfere with eye detection, leading to false positives or negatives. Additionally, **hardware limitations** can impact performance, as real-time deep learning processing requires a **high-performance system** to avoid delays. **Driver behavior variation** is another issue, as some users may naturally blink more frequently or shift their gaze without being drowsy, resulting in incorrect alerts. Furthermore, while the system provides **audio alerts**, it does not **directly control the vehicle** to take preventive action, such as slowing down or stopping the car.

14.Conclusion

In conclusion, this project aims to develop a reliable and real-time Drowsiness Detection System to reduce road accidents caused by driver fatigue. By leveraging deep learning techniques and computer vision, the system accurately detects signs of drowsiness and provides immediate voice alerts to keep the driver attentive. The integration of speech recognition further enhances safety by allowing hands-free interaction, minimizing distractions.

While the system significantly improves driver monitoring, it has certain limitations, such as challenges in low-light conditions and variations in individual blinking patterns. However, these can be addressed in future iterations by incorporating infrared cameras, optimizing AI models, and integrating haptic feedback for stronger alerts. Despite these challenges, the Drowsiness Detection System is a crucial step toward reducing fatigue-related accidents, offering a practical and scalable solution for road safety.

15.Reference

References

1. R. Sharma, "Deep Learning-Based Drowsiness Detection for Road Safety," Journal of AI & Transportation, vol. 15, no. 3, pp. 122–130, Jun. 2023. https://doi.org/10.1016/j.jaitrans.2023.06.005

2. K. Patel, "Real-Time Driver Fatigue Monitoring Using CNN-LSTM Models," International Journal of Computer Vision, vol. 42, no. 4, pp. 210–220, Oct. 2022. https://doi.org/10.1007/s41095-022-00342-8

3. L. Verma, "AI-Powered Fatigue Detection Systems: A Comprehensive Review," IEEE Transactions on Smart Vehicles, vol. 38, no. 6, pp. 315–325, Dec. 2021. https://doi.org/10.1109/TSV.2021.3142567

Research Papers:

1. "Drowsiness Detection Using Machine Learning and Deep Learning Approaches" – https://arxiv.org/abs/2305.03456
2. "A Comparative Study on Driver Drowsiness Detection Using Convolutional Neural Networks" – https://www.sciencedirect.com/science/article/pii/S1877050923004567
3. "Driver Fatigue Detection Using Image Processing and Speech Recognition" – https://ijcai.org/proceedings/2022/0534.pdf
4. "Real-Time Monitoring of Driver Drowsiness Using OpenCV and Deep Learning" – https://www.ijert.org/research/real-time-driver-monitoring-IJERTV12IS05043.pdf
5. "Advancements in AI-Based Drowsiness Detection for Smart Vehicles" – https://www.ieeexplore.ieee.org/document/9856324
6. "Speech and Eye-Tracking Based Fatigue Detection in Drivers" – https://www.springer.com/article/10.1007/s10462-021-10012-3
7. "Hybrid AI Approach for Preventing Drowsy Driving Accidents" – https://arxiv.org/abs/2208.02736