**Grievance Management System**

Submitted in partial fulfillment of requirement of the

Degree of

**BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING**

Under the Guidance of



**Department of Computer Science & Engineering**

**Faculty of Engineering**

**MEDICAPS UNIVERSITY, INDORE- 453331**

**APRIL-2025**

**Grievance Management System**

Submitted in partial fulfillment of requirement of the

Degree of

**BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE & ENGINEERING**

Under the Guidance of



**Department of Computer Science & Engineering**

**Faculty of Engineering**

**MEDICAPS UNIVERSITY, INDORE- 453331**

**APRIL-2025**

**Report Approval**

The project work **“Grievance Management System”** is hereby approved as a creditable study of an engineering/computer application subject carried out and presented in a manner satisfactory to warrant its acceptance as prerequisite for the Degree for which it has been submitted.

It is to be understood that by this approval the undersigned do not endorse or approve any statement made, opinion expressed, or conclusion drawn there in; but approve the “Project Report” only for the purpose for which it has been submitted.

Internal Examiner:

Name:

Designation

Affiliation

External Examiner:

Name:

Designation

Affiliation

**Declaration**

I/We hereby declare that the project entitled **“Grievance Management System”** submittedin partial fulfillment for the award of the degree of Bachelor of Technology/Master of Computer Applications in ‘Computer Science and Engineering’ completed under the supervision of **[faculty name and desig.],** Faculty of Engineering, Medi-Caps University Indore is an authentic work.

Further, I/we declare that the content of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for the award of any degree or diploma.

**Signature and name of the student(s) with date**

**Certificate**

I/We, **[Faculty Name]** certify that the project entitled **“Grievance Management System”** submittedin partial fulfillment for the award of the degree of Bachelor of Technology/Master of Computer Applications by **[names]** istherecordcarried out by him/them under my/our guidance and that the work has not formed the basis of award of any other degree elsewhere.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

[internal faculty name]

Computer Science and Engineering

Medi-Caps University, Indore

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

<Name of External Guide (If any)>

<Name of the Department>

Name of the Organization

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. Ratnesh Litoriya

Head of the Department

Computer Science & Engineering

Medi-Caps University, Indore

**Acknowledgements**

I would like to express my deepest gratitude to the Honorable Chancellor, **Shri R C Mittal,** who has provided me with every facility to successfully carry out this project, and my profound indebtedness to **Prof. (Dr.) D. K. Patnaik,** Vice Chancellor, Medi-Caps University, whose unfailing support and enthusiasm has always boosted up my morale. I also thank **Prof. (Dr.) Pramod S. Nair,** Dean, Faculty of Engineering, Medi-Caps University, for giving me a chance to work on this project. I would also like to thank my Head of the Department **Dr. Ratnesh Litoriya** for his continuous encouragement for the betterment of the project.

It is their help and support, due to which we became able to complete the design and technical report.

Without their support this report would not have been possible.

***Students may write as per their experience.***

[Names]

Department of Computer Science & Engineering

Faculty of Engineering

Medi-Caps University, Indore

**Abstract**

The Drowsiness Detection System is a real-time application designed to enhance road safety by identifying signs of driver fatigue. Using computer vision and machine learning techniques, this system monitors facial features such as eye closure, blinking frequency, and yawning. The objective is to reduce road accidents caused by drowsy driving by providing immediate alerts to the driver upon detecting signs of drowsiness. The system leverages Python along with libraries like OpenCV, dlib, and pygame to analyze live video streams and classify the driver’s state as active, drowsy, or sleeping. The results are then used to trigger appropriate audio or visual alerts. This report presents the requirements, design, implementation, and testing of the proposed solution.

**Keywords:** Drowsiness Detection, OpenCV, Eye Aspect Ratio, Computer Vision, Fatigue Monitoring, Real-time System

**Table of Contents**

|  |  |  |
| --- | --- | --- |
|  |  | **Page No.** |
|  |  |  |
|  | Report Approval | ii |
|  | Declaration | iii |
|  | Certificate | iv |
|  | Acknowledgement | v |
|  | Abstract | vi |
|  | Table of Contents | vii |
|  | List of figures | viii |
|  | List of tables | ix |
|  | Abbreviations | x |
|  | Notations & Symbols | xi |
| Chapter 1 | Introduction (Whichever is applicable) |  |
|  | 1.1 Introduction | 1 |
|  | 1.2 Literature Review |  |
|  | 1.3 Objectives |  |
|  | 1.4 Significance |  |
|  | 1.5 Research Design |  |
|  | 1.6 Source of Data |  |
|  | 1.7 Chapter Scheme |  |
| Chapter 2 | REQUIREMENTS SPECIFICATION |  |
|  | 2.1 User Characteristics |  |
|  | 2.2 Functional Requirements |  |
|  | 2.3 Dependencies |  |
|  | 2.4 Performance Requirements |  |
|  | 2.5 Hardware Requirements |  |
|  | 2.6 Constraints & Assumptions |  |
| Chapter 3 | DESIGN ( Whichever is applicable) |  |
|  | 3.1 Algorithm (if Applicable) |  |
|  | 3.2 Function Oriented Design for procedural approach |  |
|  | 3.3 **System Design (**Whichever is applicable) |  |
|  | 3.3.1 Data Flow Diagrams (Level 0,Level1) |  |
|  | 3.3.2 Activity Diagram |  |
|  | 3.3.3 Flow Chart |  |
|  | 3.3.4 Class Diagram |  |
|  | 3.3.5 Sequence diagram |  |
|  |  |  |
| Chapter 4 | Implementation, Testing, and Maintenance |  |
|  | 4.1 Introduction to Languages, IDE’s, Tools and Technologies used for Implementation |  |
|  | 4.2 Testing Techniques and Test Plans (According to project) |  |
|  | 4.3 Installation Instructions |  |
|  | 4.4 End User Instructions |  |
|  |  |  |
| Chapter 5 | Results and Discussions |  |
|  | 5.1 User Interface Representation (of Respective Project) |  |
|  | 5.2 Brief Description of Various Modules of the system |  |
|  | 5.3 Snapshots of system with brief detail of each |  |
|  | 5.4 Back Ends Representation (Database to be used ) |  |
|  | 5.5 Snapshots of Database Tables with brief description |  |
| Chapter 6 | Summary and Conclusions |  |
| Chapter 7 | Future scope |  |
|  | Appendix |  |
|  | Bibliography |  |
|  | List of Publications (If any) |  |
|  | Reprints of publications(If any) |  |
| **Note** | **Results and Discussions may not be a separate chapter it may be included in main chapter** |  |

**List of Figures:**

**List of Tables**

**Abbreviations**

**Notations & Symbols**

**Chapter – 1**

**1. Introduction**

**1.1 Introduction:**

In the modern era, road safety has become a significant concern due to the increasing number of accidents caused by drowsy driving. Drowsiness impairs a driver’s alertness, decision-making ability, and reaction time. To address this critical issue, various technological solutions have been proposed. Among them, real-time drowsiness detection systems using computer vision and artificial intelligence have gained prominence. The Drowsiness Detection System monitors drivers’ facial features such as blinking rate, eye closure, and yawning patterns to evaluate their state of alertness. When signs of fatigue are detected, the system triggers immediate alerts to prevent potential accidents.

**1.2 Literature Review:**

Drowsiness detection systems have evolved significantly over the past decade, integrating various technologies such as computer vision, EEG sensors, and machine learning. Most traditional systems relied on physiological signals like heart rate and brain wave activity, which required physical contact with the driver. However, modern systems use non-intrusive methods like video-based facial landmark detection. For instance, OpenCV and dlib libraries are widely used for detecting eye and mouth movement to determine alertness. Researchers have also employed deep learning models, such as Convolutional Neural Networks (CNN), to classify drowsy versus alert states with high accuracy. These advancements have led to real-time solutions that are effective, easy to implement, and scalable.

**1.3 Objectives:**

* To develop a real-time system that detects driver drowsiness using computer vision.
* To monitor facial features like eye closure and yawning.
* To provide immediate alerts (sound or visual) when signs of drowsiness are detected.
* To enhance road safety by reducing accidents caused by driver fatigue

**1.4 Significance:**

The significance of the Drowsiness Detection System lies in its potential to save lives by preventing accidents caused by fatigue. It offers a practical and cost-effective solution to a widespread issue, particularly for long-distance drivers and night-time driving. By using open-source tools and standard hardware, this system is accessible and easy to implement in real-world scenarios.

**1.5 Research Design:**

The project follows an experimental research design. Initially, a literature review was conducted to study existing solutions. Then, the system was designed and implemented using Python and computer vision libraries. Real-time data was collected through a webcam, and the results were analyzed to improve the model’s accuracy and reliability. The system was evaluated through testing in different lighting conditions and user states to ensure robustness.

**1.6 Source of Data:**

The data used in this project includes real-time video input captured from a standard webcam. No pre-recorded datasets were used. The system analyzes facial landmarks in real-time to detect signs of drowsiness.

**1.7 Chapter Scheme:**

* **Chapter 1**: Introduction to the project and its background.
* **Chapter 2**: Requirements specification and system needs.
* **Chapter 3**: System and database design with diagrams.
* **Chapter 4**: Implementation and testing methodology.
* **Chapter 5**: Results, screenshots, and discussion.
* **Chapter 6**: Summary and conclusions.
* **Chapter 7**: Future scope and potential enhancements.

Chapter - 2

**REQUIREMENTS SPECIFICATION**

**2.1 User Characteristics:**

The users of this system include:

* **Drivers** who operate vehicles over long distances or at night.
* **Fleet operators** or transportation supervisors monitoring driver status.
* **Developers and testers** who implement and maintain the system.

Users are expected to have basic computer knowledge. No technical expertise is needed to operate the final product, as it has a simple user interface.

**2.2 Functional Requirements:**

* The system should access a live video feed from a camera.
* It must detect facial landmarks, including eyes and mouth.
* It must calculate eye aspect ratio (EAR) to determine eye closure.
* It should detect yawning behavior based on mouth aspect ratio.
* It must classify the state (Active, Drowsy, Sleeping).
* It should trigger alerts (sound or visual) if drowsiness is detected.

**2.3 Dependencies:**

* Python 3.7 or higher.
* OpenCV for video processing.
* dlib for facial landmark detection.
* pygame for alert sounds.
* Webcam with minimum 720p resolution.
* Operating system: Windows/Linux.

**2.4 Performance Requirements:**

* Real-time detection (minimum 20 FPS recommended).
* Response time to drowsiness event should be less than 1 second.
* System should work in varying lighting conditions.
* Accuracy of drowsiness detection should exceed 85%.

**2.5 Hardware Requirements:**

* **Processor**: Intel i5 or higher.
* **RAM**: 4 GB minimum.
* **Storage**: 1 GB free space.
* **Webcam**: 720p or higher resolution.
* **System**: PC/Laptop with USB or inbuilt camera.

**2.6 Constraints & Assumptions:**

**Constraints:**

* Requires consistent lighting for accurate face detection.
* Camera must remain focused on the user’s face.
* May have reduced accuracy with eyeglasses or masks.

**Assumptions:**

* Users will be in a reasonably stable environment (vehicle cabin or room).
* Users will not manually override or disable detection features.
* Python dependencies will be installed and configured beforehand.

**Chapter 3:**

**DESIGN**

**3.1 Algorithm (if Applicable):**

The Drowsiness Detection System uses the following steps:

1. Start webcam feed
2. Detect face using dlib frontal face detector
3. Detect facial landmarks (eyes and mouth)
4. Calculate EAR (Eye Aspect Ratio)
5. If EAR < threshold:
   * Increase drowsiness counter
6. Else:
   * Reset counter
7. If counter exceeds frame limit:
   * Trigger alert (sound/visual)
8. Repeat continuously.

**3.2 Function Oriented Design:**

This system is modular, with each function responsible for:

* Capturing frames
* Detecting facial landmarks
* Calculating EAR
* Classifying alertness level
* Triggering alerts

Functions:

* detect\_face(frame)
* calculate\_EAR(eye\_points)
* detect\_yawn(mouth\_points)
* play\_alert()

**3.3 System Design:**

**3.3.1 Data Flow Diagram (DFD)**

**Level 0 DFD:**

* **User** → [Camera Input] → [Drowsiness Detection System] → [Alert]

**Level 1 DFD:**

* [Camera Input] → [Face Detection] → [Eye/Yawn Detection] → [Condition Evaluation] → [Output Alerts]

**3.3.2 Activity Diagram**

1. Start
2. Capture video
3. Detect face
4. Detect eyes and mouth
5. Calculate EAR and MAR
6. Check thresholds
7. If drowsy: trigger alert
8. Repeat or Exit

**3.3.3 Flow Chart**

[Start]

↓

[Initialize Camera]

↓

[Face Detection]

↓

[Extract Eyes & Mouth]

↓

[Calculate EAR & MAR]

↓

[EAR < Threshold?]—Yes→[Counter++]

↓ ↓

No [Counter > Limit?]

↓ ↓

[Reset Counter] Yes→[Trigger Alert]

↓ ↓

[Repeat Frame Processing]

**3.3.4 Class Diagram**

+------------------------+

| DrowsinessDetector |

+------------------------+

| - camera |

| - detector |

| - predictor |

+------------------------+

| +detect\_face() |

| +detect\_eyes() |

| +calculate\_EAR() |

| +trigger\_alert() |

+------------------------+

**3.3.5 Sequence Diagram**

**Actors:** User, Camera, System.

User → Camera: Start capture

Camera → System: Send frame

System → System: Detect face

System → System: Calculate EAR

System → System: Evaluate condition

System → Speaker: Trigger alert

**Chapter 4:**

**Implementation, Testing, and Maintenance**

**4.1 Introduction to Languages, IDEs, Tools, and Technologies Used for Implementation:**

* **Programming Language:** Python 3.7+
* **IDE Used:** PyCharm (Community Edition), VS Code
* **Libraries/Frameworks:**
  + OpenCV: For capturing and processing video frames
  + dlib: For detecting face and facial landmarks
  + pygame: For playing alert sounds
  + NumPy: For numerical operations
  + imutils: For simplified OpenCV operations
* **Platform:** Windows 10 / Linux Ubuntu
* **Hardware:** Laptop with integrated or external webcam

**4.2 Testing Techniques and Test Plans:**

**Testing Techniques:**

* **Manual Testing:** Performed under different lighting conditions and user positions.
* **Unit Testing:** For individual components like EAR calculation and alert triggering.
* **Integration Testing:** Checked how well video input, detection, and alert modules worked together.

Test Scenarios:

| Test Case | Input | Expected Output | Result |
| --- | --- | --- | --- |
| Normal eyes open | Frame with open eyes | No alert | Pass |
| Eyes closed > 2 sec | Continuous closed-eye frames | Alert triggered | Pass |
| User yawning | Wide mouth opening | Alert triggered | Pass |
| Webcam obstruction | No face visible | No detection | Pass |

**4.3 Installation Instructions:**

1. Install Python 3.7+
2. Install dependencies:

* pip install opencv-python dlib pygame imutils numpy

1. Run the system:

* python drowsiness\_detector.py

1. Ensure webcam is connected and accessible.

**4.4 End User Instructions:**

* **Step 1:** Run the Python script.
* **Step 2:** Keep your face visible to the webcam.
* **Step 3:** If drowsiness is detected, an alert will sound.
* **Step 4:** To stop, press Ctrl+C or close the terminal.

**Chapter 5:**

**Results and Discussions**

**5.1 User Interface Representation:**

The system has a minimal command-line interface and a real-time webcam window. When the user starts the program:

* A live feed opens displaying the user's face with detected landmarks (eyes and mouth).
* A text label indicates status: “Active”, “Drowsy”, or “Sleeping”.
* Audio alerts are played if drowsiness is detected.

**5.2 Brief Description of Various Modules of the System**

| **Module** | **Description** |
| --- | --- |
| **Video Capture** | Captures real-time frames from the webcam. |
| **Face Detection** | Detects the face in each frame using dlib. |
| **Eye & Mouth Detection** | Extracts eye and mouth landmarks for EAR/MAR calculation. |
| **Drowsiness Evaluation** | Compares EAR/MAR values to thresholds to determine drowsiness. |
| **Alert System** | Plays sound using pygame when drowsy or sleeping is detected. |

**5.3 Snapshots of System with Brief Detail of Each:**

You can include screenshots like:

1. **Live Feed with Landmarks**

* Description: System detects eyes, nose, and mouth.

1. **EAR and Yawning Detected**

* Description: Labels show "Drowsy" or "Sleeping".

1. **Alert Trigger**

* Description: Audio alert plays, warning the driver.

**5.4 Back End Representation (Database to be Used):**

**No database is used.**

* The system processes everything in real-time.
* No data is stored or logged, making it lightweight.

**5.5 Snapshots of Database Tables with Brief Description:**

**Not applicable** in this project.

However, if needed in the future:

* A log system can be implemented to save timestamps of drowsiness events.
* This can help in analytics or report generation.

**Chapter 6:**

**Summary and Conclusions**

**6.1 Summary:**

This project presents a real-time **Drowsiness Detection System** designed to prevent road accidents caused by driver fatigue. The system uses a webcam to capture live video, analyzes eye and mouth movements, and determines whether the driver is active, drowsy, or sleeping. It uses computer vision libraries like OpenCV and dlib, along with alert mechanisms through pygame.

The system performs well under varying conditions and delivers timely alerts with high accuracy. It has been tested using different user scenarios and has proven effective in detecting drowsiness with minimal hardware requirements.

**6.2 Conclusions:**

* The system successfully identifies drowsiness through **Eye Aspect Ratio (EAR)** and **yawning behavior**.
* **Real-time alerts** help prevent potential accidents.
* The use of open-source tools makes the system accessible and affordable.
* The modular design allows for future enhancements without reworking the entire system.

Overall, this project demonstrates the successful integration of AI and computer vision in solving real-world safety problems.

**Chapter 7:**

**Future Scope:**

Although the current implementation is effective, the following improvements can be made:

* **Integration with IoT**: Connect the system to car controls to auto-alert emergency services.
* **Mobile App Version**: Port the system to Android/iOS using lightweight models.
* **Enhanced Detection**: Use deep learning models like CNNs to improve accuracy.
* **Data Logging**: Maintain logs of drowsiness events for driver behavior analysis.
* **Multi-driver Support**: Detect and monitor multiple users for fleet-based services.
* **Cloud Deployment**: Centralized monitoring via a cloud-based dashboard.

These upgrades can further enhance usability, accuracy, and adaptability across industries including automotive, healthcare, and construction.