Safe Drive

Final Year Project

Session 2021-2024

A project submitted in partial fulfillment of the degree of

BS in Computer Science

Department of Computer Science

Faculty of Computer Science & Information Technology

The Superior University, Lahore

Fall 2024

Type (Nature of project)	[✓] **D**evelopment [] **R**esearch [] **R**&**D**
Area of specialization	Artificial Intelligence & Internet of Things
FYP ID	FYP-BCSM-S24-015

Project Group Members

Sr.#	Reg. #	Student Name	Email ID	*Signature
(i)	Bcsm-F20-428	Taha Ahmad	bcsm-f20-428@superior.edu.pk	
(ii)	Bcsm-F20-320	Talha Saleem	bcsm-f20-320@superior.edu.pk	
(iii)	Bcsm-F20-343	Salman Hashmi	bcsm-f20-343@superior.edu.pk	

*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

Plagiarism Free Certificate

This is to certify that, I Taha Ahmad S/O Muhammad Latif, group leader of FYP under registration no bcsm-f20-428 at Computer Science Department, The Superior University, Lahore. I declare that my FYP report is checked by my supervisor.

Date: ____/_____/_____

Name of Group Leader: Taha Ahmad Signature: _____

Name of Supervisor: Madam Javaria Qadeer

Designation: Lecturer

Signature: _____

HOD: Dr. M. Azam

Signature: _____

Safe Drive

Change Record

Author(s)	Version	Date	Notes	Supervisor's Signature
Taha Ahmad	1.0	5-Feb-2024	Introduction	
Talha Saleem	1.1	5- March-2024	Changes based on Feedback from the supervisor	
Taha Ahmad	1.2	5-April-2024	Completing SRS	
Salman Hashmi	1.3	25-April-2024	Implementing Uml Diagrams	
Talha Saleem	2.1	10-Sep-2024	Prototyping	
Taha Ahmed	2.2	25-sep-2024	Changes based on Feedback from the supervisor	
Salman Hashmi	2.3	7-Oct-2024	Testing Requirements	
Talha Saleem	2.4	7-Nov-2024	Making Conclusion	
Taha Ahmad	3.0	11-Nov-2024	Finalizing	

APPROVAL

PROJECT SUPERVISOR

Comments: _____

_____

Name: _____

Date: _____ Signature: _____

PROJECT MANAGER

Comments: _____

_____

Date: _____ Signature: _____

HEAD OF THE DEPARTMENT

Comments: _____

_____

Date: _____ Signature: _____

Dedication

In dedication to the culmination of our efforts, we extend our deepest gratitude to all those who have contributed to the realization of this project.

To our families, whose unwavering support, encouragement, and understanding have been the cornerstone of our journey. Your belief in our abilities and sacrifices made this endeavor possible.

To our esteemed supervisor, Ma'am Javaria Qadeer, for their invaluable guidance, expertise, and unwavering commitment. Their mentorship and insightful feedback have shaped our project's direction and enriched our learning experience.

To our fellow team members, whose collaboration, dedication, and synergy have been instrumental in overcoming challenges and achieving our goals. Your tireless efforts and shared vision have brought this project to fruition.

To our friends and peers, for their encouragement, understanding, and inspiration throughout this endeavor. Your encouragement and positivity have fueled our determination and perseverance.

Finally, to the pursuit of knowledge, innovation, and excellence. May this project serve as a testament to our commitment to continuous learning and making a meaningful impact in our chosen field.

With sincere appreciation,

Taha Ahmad

Talha Saleem

Salman Hashmi

Acknowledgements

All gestures of recognition are to Almighty Allah, who gave us the qualities, knowledge, and bravery to complete this task. We would like to express our heartfelt appreciation and profound thanks to our Supervisor, Miss Javaria Qadeer for their valuable and constructive suggestions during the planning and development of our project. We are grateful for their patient guidance, enthusiastic encouragement, and insightful critiques of this research work. Javaria Qadeer provided us with invaluable assistance in completing our venture, and we deeply appreciate her generous allocation of time. We would also like to extend our gratitude to the other faculty members who evaluated and provided us with suggestions for improvement in our project. A special thanks goes to Dr. Dr. M. Azam, Head of the Department of Computer Science. We would also like to express our thanks to all our faculty members who have taught us the necessary prerequisites with great care and expertise.

Executive Summary

Road accidents have always been a global concern, and among the many factors that contribute to them drowsy driving is a major one. one among the recent study regards the tired driving where the research revealed that about 21% of the road accidents are due to drowsy driving and this figure is gradually rising each year according to global status report on road safety 2015, which endorsed the information from 180 countries. This in fact draw attention the very fact that across the world the whole lot of road traffic deaths are very high due to drowsiness of drivers. This is because drivers are feeling very much fatigued, other drivers are engaging in drink-and-drive, and carelessness is also being observed. This has now affected many lives and families across mankind in different countries as is evident. This consequently gave rise to the development of Intelligent Driver Assistance Systems; time drowsy driving detection is one of the best possible majors that can be put in place to assist drivers in a manner that would make them aware of drowsy driving situations. The driver behavioral state detection system will be helpful in identifying the motive force drowsy conditions and effectively prevent mishaps. Such fatigue is caused by; heavy traffic means, increase in number of automotive, adverse driving conditions, time tight activities required in commuting to and from work place and also the work loads. Project focuses on developing an intelligent AI system using Raspberry Pi 3 and a camera module to monitor driver drowsiness in real-time. Some of the integrated sub-assemblies include a camera to capture the facial expressions of the driver, Raspberry for processing the information collected, and a speaker for generating an alert. The system uses several supporting libraries like numpy, pandas, TensorFlow, Keras, OpenCV, play sound to identify the features such as eye movement and closure for detecting drowsiness. The main application consists in accurate drowsiness detection and quantification processes. Here we have one problem that is, the size of the eyes differs from each other and some even wear spectacles. This means one has to be very cautious when capturing data on pictures to ensure the right information has been captured. Furthermore, using deep learning methodologies, the framework identifies more precise drowsiness indicators to improve the detection system, and performance assessment and validation prove the system's effectiveness for real-life driving conditions.

Specifically, the utilization of ResNet50 in the CNN model training, which reflects the project's dedication to use only the best and modern methods to detect drowsiness.

Table of Content

List of Figures

List of Tables

Chapter 1

Introduction

Chapter 1: Introduction

Here, in this chapter, we provide an insight into a few important areas of concern as far as the driver drowsiness detection project is concerned. Initially, we investigate on that how much extent it is important to detect drowsy driving and how much it is effective in increasing the rate of accident. We also consider what it takes to produce a reliable and capable detection system specifically on issues of processing and its implementation in the vehicle. Define the primary and secondary objectives of the project, as well as the project aims and objectives, which are to create an AI-based system that may conduct real-time drowsiness detection. This paper also involves a review of literature, although it is done more in order to compare the current methods and potential solutions found in the literature with the method proposed in this paper for making improvements. Here we outline the proposed solution and stress that AI with matching hardware is optimized for detection in real time. Furthermore, there is an explication of the project ideas emphasizing the importance of the project in the vehicular safety application. Lastly, the following project plan is devised in order to be able to infallibly implement the proposed solution by providing an orderly framework: In sum, this chapter provides the necessary groundwork for subsequent analysis and discussion planned in the subsequent chapters.

1.1. Background

Since drowsy driving is one of the leading causes of accidents, detecting driver drowsiness has been of great interest in recent years as it could contribute to reducing the dangerous of drowsy driving. Particularly historically, detection was largely based on subjective measures, such as self-reporting or passenger observation. However, with technology improvements, especially with breakthroughs in Artificial Intelligence (AI) and computer vision, new detection technologies emerged.

Nowadays, AI algorithms have been widely used in the generation of these systems for the detection of driver drowsiness through facial expressions, detection just by eye movements, etc. These systems commonly include cameras or sensors within the car that allow driving states to

be monitored live. When the cameras detect sleep symptoms such as drooping eyelids or yawns, the system warns the driver visually or acoustically by calling on him to take corrective action.

The latest trends in driver drowsiness detection systems are moving towards both the higher detection rate, ultra-low false-positive rate, real-time processability, improving the accuracy at the least cost, as well as helping to improve user experience. This more accurate analysis involves advanced AI models driven by deep learning techniques, in the form of facial muscle features during wakefulness to relaxation just before sleep and during sleep, which are facial patterns associated with profound sleep architectures. In addition, seamless data exchange is made possible through vehicle telematic systems and wearable devices in order to monitor driver alertness more extensively in different driving conditions.

Innovations in hardware, such as low-power computing platforms like Raspberry Pi, and the availability of open-source libraries for AI development have democratized the creation of drowsiness detection systems. This has led to a proliferation of research and development efforts aimed at improving the effectiveness and accessibility of these safety technologies.

In my latest work on driver drowsiness detection, I have focused on leveraging state-of-the-art AI algorithms and hardware platforms to develop a robust and efficient detection system. By combining the latest advancements in computer vision, machine learning, and embedded systems, I aim to create a solution that not only accurately identifies drowsiness but also integrates seamlessly into existing vehicle environments, contributing to safer and more responsible driving practices.

1.2. Motivations and challenges

This section discusses the key reasons for addressing driver drowsiness to improve road safety and outlines the technical and practical difficulties in developing an effective detection system.

1.2.1 Motivations

The main cause of the road accidents throughout the world and the road safety, is driver drowsiness which is known supportively as one of the most important accidents priming factors and, consequently, responsible for creating some tragic mortality and morbidity outcomes. This encompasses the idea of helping to solve this very problem with

the motivation of improving road safety as the main backing. Our goal is to prevent deaths in road accidents through the implementation of a proper detection system for drowsiness.

1.2.2 Challenges

Developing a robust drowsiness detection system presents several challenges. These include accurately identifying drowsiness based on facial expressions and physiological signals, ensuring real-time processing with minimal latency, and integrating the system seamlessly into vehicles without causing distractions to drivers. Additionally, variations in individual behavior and environmental factors pose challenges to the system's reliability and effectiveness.

1.3. Goals and objectives

Some goals and objectives that we need to achieve for complete the project:

1.3.1 Goals:

- Develop AI system to recognize drowsiness on time.
- Integrate the detection system in vehicles to monitor driver alertness at all times
- Provide on-time road safety by alerting the drivers if drowsiness is detected in driver.

1.3.2 Objectives:

1. Gather and analyze data on facial expressions and vital signals of drowsiness
2. Develop and deploy machine learning systems for detecting patterns that suggest sleepiness.
3. Develop a hardware platform, leveraging devices such as Raspberry Pi and cameras, to deploy the detection system in vehicles.
4. Validate the system's performance through rigorous testing under various driving conditions.
5. Evaluate the system's effectiveness in reducing the incidence of drowsy driving accidents through real-world trials.

1.4. Literature review

A literature review is useful in any research where it is conducted as it gives an overview of the current developments within a chosen subject area, including a driver drowsiness detection system. It provides its readers with information about what has been done before and where there are limitations in knowledge about a given topic, which is useful in developing the overall framework for the project. The literature review is important in developing the novel system as it compares previous investigations and solutions as well as facial recognition and EEG monitoring systems for designing the methodology of the new system. It captures the strengths and weaknesses of the current ideas, to emphasize the direction for betterments and advancements. For example, issues with changes in lighting, and the effects of glasses are just some of the areas which can be improved when it comes to recognition. Also, the literature review explains why the project and disposing research area is needed and crucial to bypass some of the questions and inequality and improve recent advances in technology. Finally, it also helps the project progress from where others left off, not duplicate efforts of others while adding value to the effort of enhancing the safety of drivers on the road through appropriate technology. There is the following content:

1.4.1 Implementation of Haar Cascade Classifier and Eye Aspect Ratio for Driver Drowsiness Detection

Using Raspberry Pi:

The study focusses on driver fatigue as one of the highest factors causing road accidents in Malaysia. The authors proposed a detection system employing the image processing methodology implemented on a Raspberry Pi. This system involves Raspberry Pi camera module and Raspberry Pi 3 board and the posing technique involves Haar Cascade Classifier for eyes and face detection and Eye Aspect Ratio (EAR) which whether the eyes of the child are open or closed. Thus, experiments carried on the six subjects show that on the one hand, the system has high accuracy depending on drivers' head position and

their absence of eyeglasses. threshold of hearing, the average EAR values observed fell in the range of 0. 141 for closed eyes to 0.

The percentages ranged between. infields, closed eyes This percentage varied from 141 for closed eyes to 0. 339 for open eyes. The study concludes that it is successful and the implemented algorithms are suitable in Raspberry Pi platform. Other recommendations for future work are to add the Raspberry Pi Touch Screen to the apparatus to make it more space-friendly and include physiological sensors to refine the evaluation.

1.4.2 Drowsy Driver Detection Using Open-cv And Raspberry Pi3:

This paper will focus on explaining the problem of driver drowsiness which is a major cause of traffic hazards and fatalities experienced globally. Lack of sleep, or drowsiness, decreases the ability of an individual to remain wakeful and alert while on the wheel. The paper first elaborates on the problem and importance of recognizing driver fatigue to avoid accidents, then presents an IoT sensor-based approach to identify drowsiness in live environment. The alongside this paper we have designed a system that uses OpenCV and Raspberry Pi 3 for monitoring of the driver's condition and alerting the driver in case of drowsiness. This automated stereo detection system seeks to ensure that drivers are warned in ways that will reduce accidents resulting from fatigue while on the road.

1.4.3 Existing Solutions:

1. The existing solution of 1st Paper is the first research paper presents an image processing-based driver drowsiness detection system using a Raspberry Pi camera module interfaced with a Raspberry Pi 3 board. The system employs the Haar Cascade Classifier algorithm for detecting eyes and faces, and the Eye Aspect Ratio (EAR) algorithm to detect eye blinks (open and closed states). The study conducted several experiments on six subjects and found that the accuracy of the Haar Cascade classifier depended on the driver's sitting position (head facing the camera) and the absence of glasses or shades. The EAR values for closed and open eyes were identified, demonstrating the system's ability to detect drowsiness. The paper suggests future improvements, including using a Raspberry Pi

Touch Screen and integrating physiological sensors like alcohol and heart rate sensors to enhance detection accuracy and reduce hardware complexity.

2. The existing solution of 2nd Paper is driver drowsiness, for which reliable measurement and detection have remained crucial challenges, is the focus of this research paper given its effect on traffic smash and Figure 1 illustrates some of the real-life fatal traffic accident causes related to driver fatigue, including drowsiness. Sleeping or drowsiness refers to a situation where an individual feels sleepy due to inadequate rest time and can be fatal while driving. The paper starts with the analysis of the issue of the driver fatigue and its connection with car accidents and further introduces an IoT approach to detect fatigue in real time. In the proposed system, OpenCV and Raspberry Pi 3 are used with the purpose of detecting the driver's state, as well as to provide an alert when drowsiness is present. This is an automated detection system that is employed to improve motorist's safety in order to reduce cases of accidents resulting from fatigued driving.

1.4.4 References:

- Sahayadhas, K. Sundaraj, and M. Murugappan, (2012). "Detecting driver drowsiness based on sensors: A review," Sensors (Switzerland), vol. 12, no. 12, pp. 16937–16953, doi:10.3390/s121216937.
- Anil Kumar Biswal, Debabrata Singh, Binod Kumar Pattanayak, Debabrata Samanta, Ming-Hour Yang, "IoT-Based Smart Alert System for Drowsy Driver Detection", Wireless Communications and Mobile Computing, vol. 2021, Article ID 6627217, 13 pages, 2021.

Title	Model	Parameter	Results	Limitation	Gap to fil
Implementation of Haar Cascade Classifier and Eye Aspect Ratio	Haar Cascade Classifier algorithm and ResNet-50	InceptionResNetV2 CNN Optimizer: Adam Batch size and epochs (not specified)	Detect the drowsy driver with 82.2 accuracy of model.	Some inaccurate results in night time detection. Due to multiple features raspberry pi3 is slow.	Train the model on night time data set and also use 4gb ram raspberry pi3 for high-speed performance.
Drowsy Driver Detection Using Open-cv And Raspberry Pi3	ResNet-50	preprocessing augmentation techniques before feeding images into the neural networks.	Perfect detection in day and night.	Device is not Working in Hot Areas.	Use official cooling fan base pi case and install thermoelectric cooling chip (TEC) for enhance cooling system.

1.4.5 Literature Review Table

1.5. Gap Analysis

The Gap that we analyze and cover.

1.5.1 Existing Limitations:

1. Some inaccurate results in night time detection.
2. Due to multiple features raspberry pi3 is slow.
3. Device is not Working in Hot Areas.

1.5.2 Addressing Gaps in the Proposed System:

1. **Improving Data Set:** Train the model on night time data set and also use 4gb ram raspberry pi3 for high-speed performance.
2. **Use Best Cooling System:** Use official cooling fan base pi case and install thermoelectric cooling chip (TEC) for enhance cooling system.

1.6. Proposed Solution

Our solution is a comprehensive drowsiness detection system, designed to protect drivers and passengers from the risks of drowsy driving. Combining hardware components, including a Raspberry Pi 3, small speaker and a high-resolution camera, our system provides robust and reliable monitoring of driver tiredness indicators.

Using advanced machine learning algorithms, our system analyzes real-time facial expressions and eye movements to accurately identify signs of drowsiness, such as yawning and drooping eyelids. This approach enables our device to interfere quickly, issuing noticeable alerts to prompt the driver to take corrective action, such as resting or switching drivers.

With its user-friendly interface and seamless integration into existing vehicle setups, our solution offers unparalleled convenience and accessibility. Whether installed in commercial fleets or personal vehicles, our device serves as a vigilant companion, continuously monitoring driver alertness and ensuring optimal safety on the road.

1.7 Project Plan

We describe complete project plan in form of Work Breakdown Structure, Roles & Reasonability Matrix and Gantt Chart.

1.7.1. Work Breakdown Structure

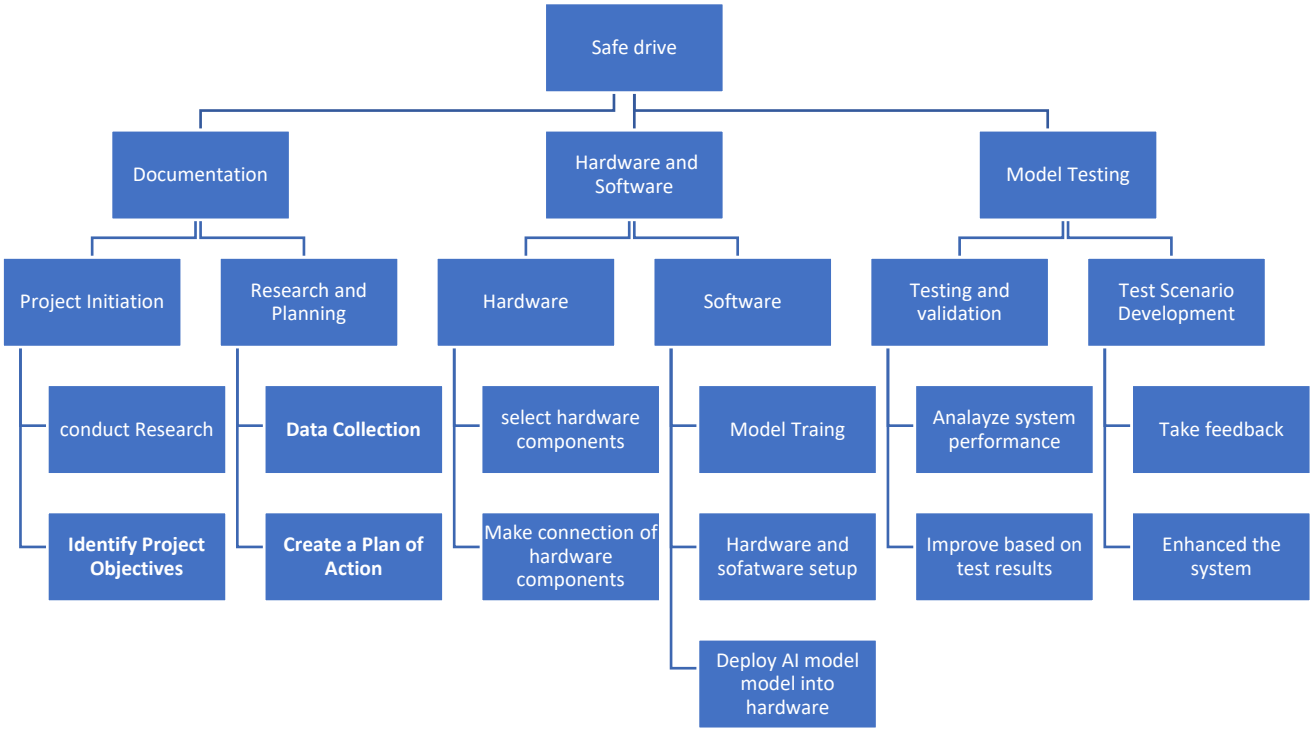


Figure 1 Work Breakdown Structure

1.7.2. Roles & Responsibility Matrix:

WBS #	WBS Deliverable	Activity #	Activity to Complete the Deliverable	Duration (Days)	Responsible Team Member(s) & Role(s)
1	Planning and Preparation	1	Define project scope, objectives, and deliverables	5	In collaboration with all group members
2	Conduct research on drowsiness detection systems	2	Research Team	4	Taha Ahmad
3	Set up project repository and documentation structure	3	Documentation Team		Talha Saleem
4	System Development and Setup	4	Acquire and set up hardware components	4	In collaboration with group
5	Develop drowsiness detection software	5	Software Development	10	Taha Ahmad
6	Integrate hardware and software components	6	Integration Team	5	In collaboration with group
7	Conduct initial testing and debugging	7	Testing Team		In collaboration with group
8	Testing and Validation	8	Design and implement test scenarios	6	Salman Hashmi
9	Collect and analyze data on system performance	9	Data Analysis Team	12	In collaboration with group

WBS #	WBS Deliverable	Activity #	Activity to Complete the Deliverable	Duration (Days)	Responsible Team Member(s) & Role(s)
10	Make improvements based on test results and feedback	10	Development Team	6	
11	Close out project and archive documentation	11	Project Manager, Documentation Team	2	In collaboration with group

1.7.3. Gantt Chart

TASK	START Date	END Date	Duration	Assigned To	Progress
Project Total Time	15-Jan	10-Dec	321		10%
Research on Drowsiness Detection Systems	15-Jan	5-Feb	20	Taha Ahmad	100%
Set up Project Repository and Documentation	6-Feb	20-Feb	15	Talha Saleem	100%
Acquire Hardware Components	21-Feb	5-Mar	15	Team work	80%
Develop drowsiness detection software	6-Mar	10-Apr	35	Taha Ahmad	25%
Data Preprocessing	11-Apr	10-May	30	Taha Ahmad	10%
Algorithm Setup	11-May	5-Jun	25	Taha Ahmad	50%
Testing of Software	6-Jun	10-Jul	35	Salman Hashmi	0%
Integration of Hardware and Software	11-Jul	29-Aug	48	Taha Ahmad	0%

Initial Testing and Debugging	30-Aug	25-Sep	25	Team work	0%
Implement Improvements	26-Sep	26-Oct	30	Team Work	0%
Project Closure and Documentation Archive	27-Oct	10-Dec	43	Talha Saleem	0%

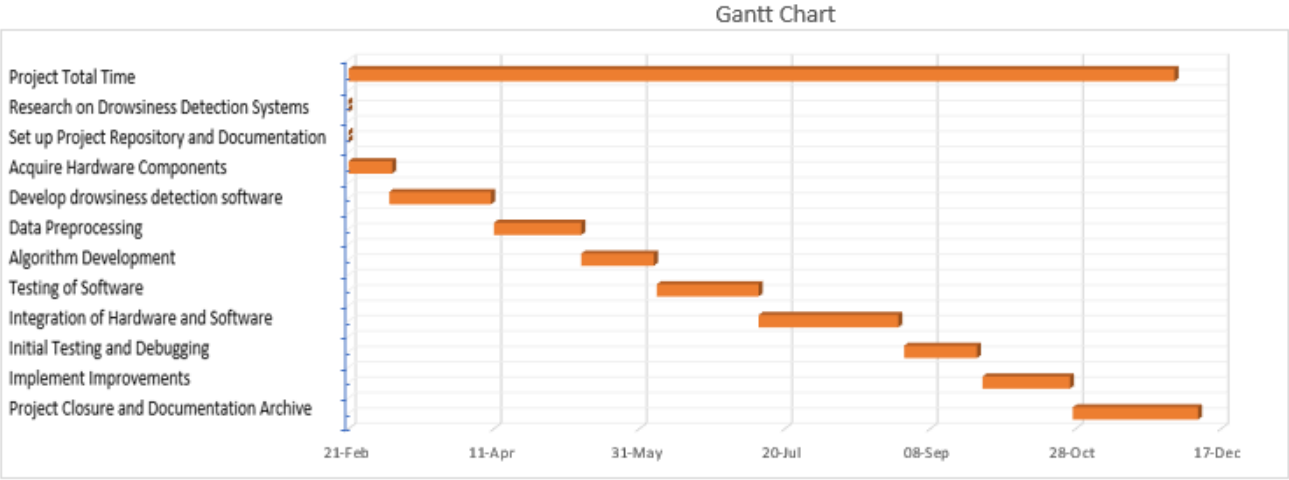


Figure 2 Gantt Chart

1.8. Report Outline

Thinks	Feels	Says
Concerned about safety while driving and avoiding accidents due to drowsiness.	Frustration if system frequently gives false alerts or misses signs of drowsiness.	I need a reliable solution to prevent drowsy driving.
Interested in a system that accurately detects drowsiness without being intrusive.	Anxious about the effectiveness of the system.	I want a user-friendly and non-intrusive system that ensures my safety without being disruptive.
Values a system that adapts to different driving conditions and driving habits.	Confident in the system's ability to work in various driving conditions.	I appreciate a system that adapts to my driving habits and keeps me alert.
Curious about the technology behind the drowsiness detection system.	Motivated to use the system for safer driving.	I am open to using technology to enhance my driving experience.

Chapter 2

Software Requirement Specifications

Chapter 2: Software Requirement Specifications

2.1. Introduction

In this chapter, we delve into the detailed description of the driver drowsiness detection system, encompassing its purpose, document conventions, intended audience, and product scope. Additionally, we explore the system's external interface requirements, system features, nonfunctional requirements, and other critical aspects such as usability, reliability, maintainability/supportability, and portability. This comprehensive overview provides a solid foundation for understanding the design, functionality, and objectives of the driver drowsiness detection system.

2.1.1Purpose

Driver drowsiness detection is an idea that has been mooted to develop a smart software tool that is able to detect signs of driver drowsiness in real time so as to minimize on the likelihood of drowsy driving. The principles applying artificial intelligence each will allow the system to quickly and efficiently identify signs or markers more akin to drowsiness including fluctuations in face and eye movements. The first objective is aimed at improving road safety by preventing drowsiness and fatigue among drivers by providing corrective actions; the drivers can take a break or switch with another driver for a short time. In its essence, the project aims to achieve the above objective through the use of audible alerts on specific messages meant to remind drivers about the risks posed by driving while sleepy. In conclusion, the idea would ensure the minimization of risks brought about by drowsy driving and help make the roads safer by promoting attentiveness and accountability.

2.1.2 Document Conventions

The guidelines covering some aspects of this project show the manner in which our work, focusing on the development of the driver drowsiness detection project, will be organized, structured and presented to the different stakeholders for their understanding. These conventions include the coding style and conventions, the relative priority levels between the

different requirements, standards and/or types of typographical conventions that might be needed in the scope of the project. The rules of organization and style are designed to make all the documents well-arranged, and hence easily understandable in the context of providing information regarding the objectives of the project, its specifications, and progress made. Priorities of requirements are useful to make distinctions between different features or functions that must be delivered with the project, to clearly focus on the needs that are more critical for the accomplishment of its objectives and to support decisions about the allocation of resources and time. In addition, particular standards or typographic procedures are used to enhance the work overall, using terminology consistently and adhering to established standards for naming conventions and the structure of documents to minimize confusion and improve integration with other project materials. Through such conventions the flow of the software development and concession is made easy hence enhancing the participation of the parties involved in completing the project goals

2.1.3 Intended Audience and Reading Suggestions

The target audience includes the driving components of the driver drowsiness detection project which includes the developers, project managers, shareholders, and all the stakeholders of the project. Understanding the diverse assignment and commitments of this audience, the materials applied in the frame of this work are expected to enhance their understanding and usefulness for each stakeholder category. Technical design documentation for developers will include extensive lists of technical requirements, source codes, and recommendations on how to accomplish specific tasks. PMs will also value concise status reports, summaries and proposal and resource status plans to effectively manage project and timeframe. Potential users and investors will be given interpretations of the goals, advantages, and achievable results of the project in refined language comprehensible to a layman. Further, every reader is supported in his or her understanding of how to utilize the documentation and offered personal recommendations on the reading, depending on the reader's role within the project. The following approach helps to assess the expectations of all the stakeholders as it provides an understanding of how to meet their documentation needs when managing a particular project.

2.1.4 Product Scope

This is a project focuses on launching driver drowsiness detection system using artificial intelligence. This works in real-time, it just checks and analyzes the driver for signal of tiredness as he is driving. These are some slumber markers such as eye closures, head nodding, and the other related expressions of drowsiness. The main goal is to detect warnings at the early stages and warn the driver before the situation becomes critical or causes an accident. Thus, the goal of the project will be as follows: by preventing drowsiness on the roads, it will be possible to achieve maximal results in the sphere and minimize the number of associated dangers. By integrating up-to-date technology and effective detection methods, the project seeks to achieve a significant reduction in the risk of road accidents due to drowsy driving.

2.1.5 References

The present work relies on references that help in its formulation and the development of the driver drowsiness detection system. These are vast and they include research papers, technical specifications, AI models and all other kinds of materials that can be relevant. Employing references also assist in ascertaining more information for better knowledge concerning other approaches approved for adoption and the methodologies and technologies utilized for similar projects. Through reviewing relevant literature and consulting from domain knowledge, the developer acquires better insights into the challenges of detecting drowsiness and can better approach the creation and implementation of corresponding applications. Moreover, proper use of references affords an opportunity to check the feasibility of the steps taken in this project implicating that the approach being utilized conforms all requisite procedures that should be adopted by any company. There-fore giving the project this knowledge stock will enable it to gain from tested and well researched strategies and findings as a result bringing into a well-researched and reliable driver drowsiness detection system.

2.2. Overall Description

The driver drowsiness detection system integrates into vehicles as a specialized safety subsystem, providing early identification of driver drowsiness to complement existing safety

features. It serves drivers and administrators, ensuring road safety by detecting and alerting drowsiness in real-time, irrespective of lighting or weather conditions.

2.2.1 Product Perspective

The driver drowsiness detection system is an object that is integrated into the vehicle's interior space and works as a separate specialized sub- system aimed at increasing the level of safety of vehicular traffic. Located in the context of the safety system of vehicles, it is currently used alongside others such as collision avoidance systems and lane change warnings. However, its key concern targets the early identification of drivers' drowsiness, thus making it an excellent supplement to the other safety features that offer only response mechanisms. Thus, the system acts as a behavioral feedback that informs drivers about their condition and helps prevent accidents due to fatigue as a result of constantly monitoring driver activity and analyzing indicators of drowsiness in real-time. In this respect, the driver drowsiness detection system being the focus of this paper has the fundamental responsibility of ensuring the protection of the vehicle occupants as well as other users of the roads, which is essential for enhancing the safety and security that surrounds the operation of vehicles.

2.2.2 User Classes and Characteristics

The driver drowsiness detection system serves two distinct user classes: It would not be implementing new policies for only the drivers and the administrators. Consumers are the category that interacts with the system and controls vehicle from their side known as drivers. Their functions and roles chiefly involve the reverberation of alerts that are produced by the system where signs of drowsiness are recognized. Drivers present many different attributes such as the level of experience, difference in gender, and the ability to become drowsy behind the steering wheel. For example, examining the candidate driver population might show that some drivers are more vulnerable to fatigue because of age, health conditions or circadian rhythms. While students are directly in charge of inputting information into the system, administrators are in charge of the overall care, settings, and subsequent calculations of the system. They have skills in system configuration and problem solving that proves beneficial when managing the system

in a way that it runs effectively and remains consistent and precise. Many are the times that the administrators are tasked with the responsibility of checking on the system overseeing their performance and that of the employees, calibrating it and even reviewing the collected data to see whether the system is effective or not. In this respect, the project targets drivers and other road users and administrators in its bottom line of reducing risk associated with drowsiness on the roads.

2.2.3 Operating Environment

The driver drowsiness detection system is designed in a way that does not give power to any kind of light, weather and the type of roads for driving. it must work effectively in all conditions depending on the type of car it is installed on ranging from a small car to a truck and whether is operating in town or out on the highways. This means that it should continue to be effective even when there is too much light around, whether it be during the day or at night, for example during a foggy evening. Regardless of the location where a driver operates a vehicle, which could be in the city or on highways, this device should have the capability of detecting signs of fatigue and alert them. These include handling of different conditions that lead to overcoming the dangers of drowsy driving thus helping keep drivers and passengers safer.

2.2.4 Design and Implementation Constraints

When considering the development and integration of the driving drowsiness detection system, the following constraints are noteworthy. One such interruption is the constraint related to the actual board used in the onboard computing such as the power available for computation and memory for storing data, etc. Because the system is based on real-time data processing gathered from on-board sensory devices such as cameras, it has to be integrated in the context of real time computing limitations of the system. It implies tuning up algorithms and functions on how to work well in terms of how they would utilize the processing power and memory without burdening the performance or introducing errors. Furthermore, commerce with reference to automobile safety standards and laws is an imperative condition that cannot be dropped. Safety measures also need to be taken into consideration to guarantee compatibility of the system with already existing automobile systems and its legal usage on roads. This involves doing a test and

evaluating to ascertain whether the incorporated system has met the standard safety measures and does not bring danger on the vehicle occupants and other road users. Thus, with these considerations addressed, the development and deployment of the driver drowsiness detection system can begin and further be developed in the manner that would ensure both it is effective and safe.

2.2.5 Assumptions and Dependencies

The driver drowsiness detection system employs two basic assumptions, which are to do with willingness of drivers to interact with the drowsiness detection features, and willingness to act appropriately in response to alerts. This automatically assumes that drivers will have to take some level of responsibility as part of accepting that they have to be awake and alert when on the stand. In order for the system to work, drivers should understand how important the alerts are and obedience to the alert by taking the right measures, for instance, exit the road and rest or change drivers if possible. Below these are: requirements of power supplies within the vehicle to constantly power the detection system and Dependencies. Lack of constant power supply to the system could hamper the proper functioning of the system, which essentially relies on a detection of drowsiness, and provide an alert. Additionally, compatibility with car CAN BUS or other proper standards is critical for proper integration with other safety features and constant data transmission. This allows the driver drowsiness detection system to communicate with other technologies that are new in automobiles like collision avoidance systems or the systems that warn you when you are drifting out of your lane or not in your lane making the overall safety higher. In this way, those dependencies and assumptions can be listed so that the development of the system and its implementation would be possible, with regards to the actual real-life driving environment.

2.3. External Interface Requirements

The external interface requirements of the driver drowsiness detection system emphasize user interaction through auditory and visual cues, aiming for simplicity and clarity to assist drivers effectively. Hardware interfaces include components like the camera module and alerting system, while software integration with image processing libraries enhances the system's ability to detect drowsiness accurately in real-time.

2.3.1 User Interfaces

The interaction between the driver and the driver drowsiness detection system is mainly through the interface, the system will use Auditory feedback as the main way to convey information to the driver. Visual cues, like warning lights and authoritative voice that will be applied to check loud and clear to the driver when signs of drowsiness like yawning or closing the eyes are seen. Lastly, we consider using the alert cues to inform the driver of the system's findings, even if there are visual disturbances that can hinder their perception. As for the layout of the user interface, it is optimized for simplicity, which can significantly help the driver to understand everything without the need to struggle with complicated units and measurements. This is to ensure the use of definite and brief tones with the help of which the level of panic is expressed without increased stress or ambiguity. In this way, while the general simplicity of the user interface tries to avoid any possible distraction by acting as an unobtrusive assistant to the driver in his fight against drowsiness, the clarity of the options helps the driver to be better equipped to act according to the information provided and reduce the risks of traffic accidents.

2.3.2 Hardware Interfaces

The hardware interfaces of our system include the following: The main components of our system are Camera module this is an essential component of our system in that it is responsible for capturing the video feed and processing it to detect driver drowsiness. Alerting system this refers to the hardware arrangement through which the system generates alerts to the driver to wake up and continue driving. Integral to this interface is a camera module which once captures the

face of the driver is able to identify expressions and head movement that suggest drowsiness. In parallel with the camera there is a microcontroller or the single board computer like Raspberry Pi functioning as the data processor for performing real time calculations on the data fed by the camera. By incorporating a speaker, it is possible to create an audible signal to wake the driver up, in the instance that the signs of drowsiness are sensed. Moreover, the hardware interface might possess extra sensors, which might be useful for monitoring other signs, such as head position or blinking frequency of the subject, in order to improve the validity of drowsiness evaluation. It also plays an important role where the hardware components need to communicate effectively so as to facilitate the captured data analysis and the resultant generation of the right alert. In this way, the integration and communication of the elements of the hardware necessary for covering real-world driving allows the system to perform its purpose of detecting and preventing the risks related to drowsy driving efficiently.

2.3.3 Software Interfaces

Our system may require integration with pre-existing software components to facilitate data processing, analysis, and alert generation. This integration could encompass interfacing with image processing libraries or AI frameworks to accurately recognize facial expressions indicative of drowsiness. By leveraging these established software tools, our system can enhance its ability to detect subtle signs of fatigue in real-time.

2.4 System Features

Describe what our drowsiness detection system does. We break down each important thing it does, like how it figures out if a driver is getting sleepy and what it does about it. We also explain which things are most important and how the system responds when it notices signs of drowsiness. Additionally, we list out all the specific things each part of the system needs to be able to do to work properly. This section helps guide the development of our system, making sure it does its job well and keeps drivers safe on the road.

2.4.1 System Feature 1

Facial Expression Detection

2.4.1.1 Description and Priority

This include includes identifying facial expressions demonstrative of tiredness, such as yawning or eye closure. It is of tall need due to its basic part in recognizing tired driving behavior.

2.4.1.2. Stimulus/Response Sequences

Stimulus: Capture facial pictures from the camera module.

Response: Analyze facial highlights to distinguish signs of drowsiness.

2.4.1.3 Functional Requirements

REQ-SF1-1: Actualize a facial acknowledgment calculation to recognize key facial landmarks.

REQ-SF1-2: Create a classifier to recognize between ordinary and tired facial expressions.

REQ-SF1-3: Coordinated real-time preparing to guarantee opportune discovery of drowsiness.

2.4.2. Framework Include 2: Sound-related Caution Generation

2.4.3. System Feature 2

Framework Include 2: Sound-related Caution Generation

2.4.2.1. Description and Priority

This include includes producing sound-related cautions to inform the driver

when laziness is identified. It is of medium need as it specifically impacts driver safety.

2.4.2.3. Stimulus/Response Sequences

Stimulus: Discovery of tiredness through facial expression analysis.

Response: Trigger a capable of being heard alert or voice incite to caution the driver.

2.4.2.3. Functional Requirements

REQ-SF2-1: Plan assortment of caution tones to cater to diverse driver preferences.

REQ-SF2-2: Execute volume control to alter the caution concentrated based on surrounding clamor levels.

REQ-SF2-3: Guarantee clear and brief caution messages to encourage quick driver response.

2.5 Other Nonfunctional Requirements

Here, we provide an overview of the four nonfunctional requirements. These four nonfunctional requirements are some of the aspects that are also important for the efficient and effective operation of our driver drowsiness detection system. These include the performance aspects, safety and security, human interface, robustness and dependability, serviceability and modularity, flexibility, and computational effectiveness. Performance requirements describe how effectively the system should operate under given circumstances – to detect drowsiness in real-time properly. Considerations involves reducing overall risk to the car and making sure that the system does not endanger the life of the driver or the passenger. Security issues involve protecting the unauthorized user from accessing improper data and the system in improper methods. The requirements of usability imply that the developed system has to be convenient for both the ordinary drivers and the administrators.

2.5.1 Performance Requirements

The evaluation metrics for the driver drowsiness detection system include a maximum of 100 milliseconds of processing delay needed to process the facial images and provide alerts in real-time. This requirement is higher to ensure that any sign of yaws is detected early so that immediate measures can be taken to avert any possible mishaps. Reducing the time of processing delay is of paramount importance, so that the cues for drowsiness can be detected and reacted to, without significantly impacting the driver's experience with the system. Moreover, facial expression recognition accuracy in the system had to be more than 95 percent acceptable even under different lighting conditions. This requirement highlights the need to accurately detect mild facial movements that are characteristic of drivers who are dozing off including eye blinking

or lids drooping at different ambient lighting conditions. Consequently, through increased accuracy in the interpretation of the facial expressions detected on the faces of the drivers, the system also improves its performance in recognizing drowsiness and alerting drivers to become more careful thereby improving road safety.

2.5.2 Safety Requirements

It may therefore be possible to summaries the following key concerns. Safety considerations are also paramount within the driver drowsiness detection system meant for designing any part of the system which might cause degradation of road safety. In particular, the sounds that are produced by the system must play the role of the auditory signals to signal the driver about potential violations; while being loud enough to make the driver look at the spectrum in the car, they cannot be too loud to distract the driver or cause discomfort. In that respect, the system guarantees that the sound of the alerts is audible without being too loud or intrusive and to the extent that the driver can be able to hear them without necessarily being distracted from his or her duties of driving. Furthermore, the system is integrated and structured in such a way, that it complements the driver, without constraining him/her or impeding on his/her capacity to safely drive the vehicle. This can comprise factors such as avoiding any form of interference when it comes to distractions such as visual or auditory prompts, making sure that alert notifications cause no interference to the driver and focusing on the fact that the driver should always remain in full control of the car at all times. Finally, by addressing safety as one of the key aspects of the considered system's design and operation, the driver drowsiness detection system would serve to reduce the risk related to drowsy driving and, therefore, to protect the lives of everyone on the road.

2.5.3 Security Requirements

Still, in our system, PRIAMRY CONCERN had been paid to the issue of privacy and, therefore, the personal information of the users is protected with optimal security measures. Each time that a data has to be transmitted over the internet, then certain coded languages have to be put into practice so that this data is not easily intercepted. Plus, once We receive it, we ensure it is closely

guarded with appropriate data storage techniques such that only appropriate personnel can access it. As for users, only the selected people can enter the system since this or that user needs to confirm his or her identity using password or any other necessary request. By doing all this, we ensure that users' privacy is protected every step of the way, constantly making the user aware that their privacy is secure and their data is safe with our system.

2.6. Other Requirements

The usability, reliability, maintainability/supportability, and portability requirements of our driver drowsiness detection system ensure its effectiveness, trustworthiness, ease of maintenance, and adaptability across diverse vehicle platforms, ultimately enhancing road safety for drivers worldwide. These requirements form the backbone of our system's design, guaranteeing user-friendly operation, consistent performance, easy maintenance, and seamless integration into various vehicle environments.

2.6.1 Usability Requirements

Our approach maintains simplicity in the design of the user interface of our system because this will enhance its functionality by making it easy for drivers to comprehend and act on the alerts as necessitated by the circumstances. Because drivers must respond quickly to these alerts while on the road, especially in situations that the car identifies as potentially soporific, the interface for fatigue recognition should be easy to operate. Through its simple design, which is devoid of any complicated buttons or features that a driver might encounter while operating a vehicle, we help the driver recognize signs of fatigue and do something about it before getting behind the wheel again. Furthermore, the guidelines are straightforward when it comes to installing the system when it comes to system setup, and this makes it easier for users to go through the initial setup without much fuss. These instructions are designed to be uncomplicated and free from confusion during the set-up, so that there would be no significant disappointment among the wider populace that we encourage to use the given system. Thus, it is aimed to develop the

application on the driving side so that drivers can easily understand this fatigue detection system quickly and be able to use it in order to increase road safety for all those related.

2.6.2 Reliability Requirements

We want our system to be super reliable, meaning it works without any problems most of the time. To make sure of this, we regularly check how well it's working and fix any issues we find right away. This helps us catch any potential problems before they become big issues, so our system keeps running smoothly. By doing these checks and fixes regularly, we aim to minimize any downtime or failures, ensuring that our system is always available when users need it. This way, users can trust our system to work reliably, giving them confidence in its effectiveness for keeping them safe on the road.

2.6.3 Maintainability/Supportability Requirements

We are aware that our system must be easy to support and maintain hence the simple mode of operations that will allow the technicians to solve any problem as a technician with ease. This is accompanied by detailed documentation, diagnostic aids, and the ability to connect to the technician’s device and help diagnose issues to minimize the time spent on problem-solving. This eliminates a lot of time that the system would have been closed such that it can support safety of drivers on the roads with maximum efficiency.

2.6.4 Portability Requirements

The hardware requirements of our system are intentionally kept simplistic and flexible so that the system could be made portable and deployable on a diverse range of vehicle hardware platforms. This is done based on the compatibility of the software with Raspberry Pi and camera modules of different models, making it easy to install the software in different vehicle types. Further, our system is designed to work on different operating systems, arising from vehicle initialization and configuration variations. Whether to be used in commercial fleets, or in individual cars, adapting our system to the installed hardware and operating systems will not prove to be a major issue in terms of implementation. This adaptability means that the

functionality of our system is not locked into a particular car model or environment, which eliminates dependency issues and ensures a uniform level of performance and stability. To ensure that the system can be seamlessly implemented in different platforms and devices, we make it a point to focus on portability and compatibility thus increasing road safety for drivers around the world.

Chapter 3

Use Case Analysis

Chapter 3: Use Case Analysis

Centers on sketching out the different scenarios and intuitive between clients and the driver tiredness location framework. It differentiates and describes characteristic use cases to get it how clients will be associated with the framework and what functionalities are required to meet their needs viably. This chapter serves as an outline for the advancement group, giving a clear understanding of client prerequisites and framework behavior.

3.1. Use Case Model

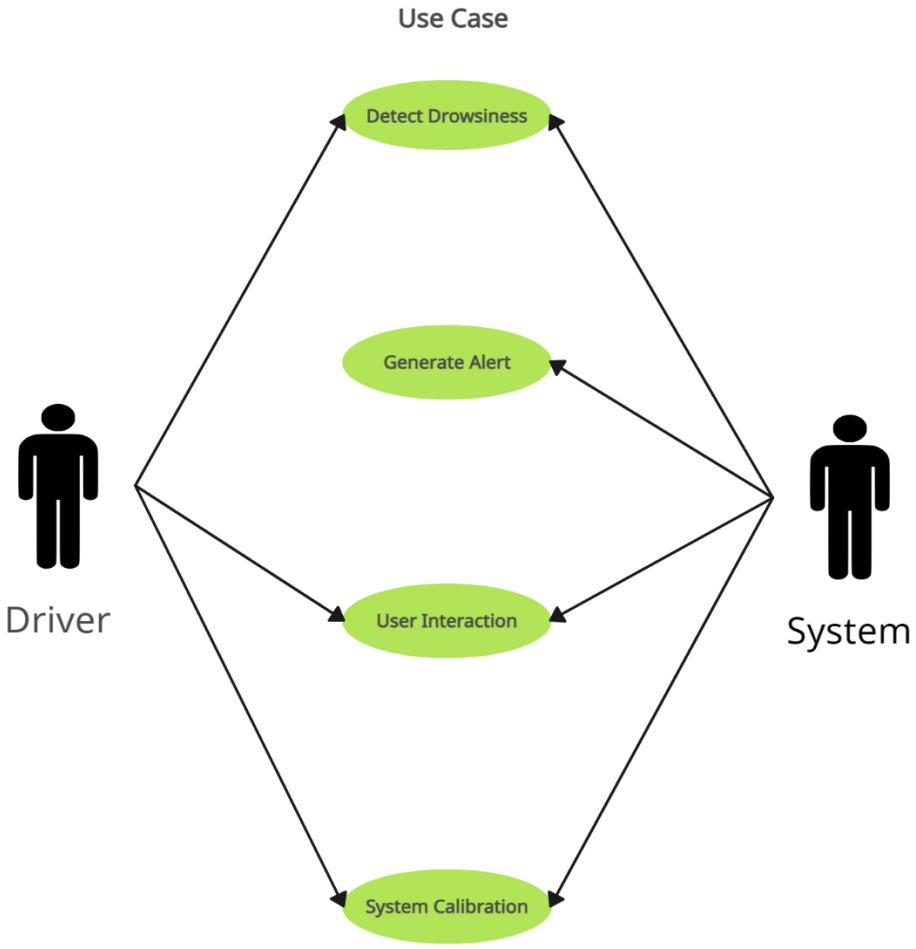


Figure 3 Use Case Model

3.2. Use Cases Description

Use Case#	1
Use Case Name	Detect Drowsiness
Actor	Driver, System
Description	The framework ceaselessly screens the driver's facial expressions to identify signs of laziness, such as yawning or hanging eyelids.
Preconditions	The framework is operational, and the driver is in control of the vehicle.
Flow	1. The framework captures the driver's facial picture utilizing the onboard camera. 2. Facial picture preparing calculations analyze the picture to identify signs of drowsiness. 3. If laziness is identified, the framework triggers an alert.
Postconditions	A caution is produced if laziness is detected.

Use Case#	2
Use Case Name	Generate Alert
Actor	System
Description	The framework produces a caution to inform the driver of identified drowsiness.
Preconditions	Drowsiness is detected.
Flow	1. The framework triggers a sound-related alert to caution the driver. 2. Visual markers may too be enacted, such as blazing lights or a caution message on a display. 3. The alarm continues until the framework recognizes that the driver is caution and responsive.
Postconditions	The driver is cautioned to the nearness of drowsiness.

Use Case#	3
Use Case Name	User Interaction
Actor	Driver, System
Description	Permits the driver to connected with the framework, such as altering caution settings or recognizing alerts.
Preconditions	The framework is operational, and the driver is present.
Flow	1. The driver gets to the framework interface, which may be through physical controls or a touchscreen interface. 2. The driver can alter caution edges, customize alarm inclinations, or recognize alerts.
Postconditions	The driver's intelligent are reflected in the system's settings or responses.

Use Case#	4
Use Case Name	System Calibration
Actor	Driver, System
Description	Empowers the framework to calibrate its location calculations based on client inclinations and natural factors.
Preconditions	The framework is operational, and the driver starts calibration.
Flow	1. The framework prompts the driver to perform calibration errands, such as yawning or closing their eyes briefly. 2. Based on the driver's activities, the framework alters its location edges and calculations accordingly.
Postconditions	The framework is calibrated to the driver's inclinations and natural conditions.

Chapter 4

System Design

Chapter 4: System Design

The driver drowsiness detection system uses a camera module to monitor the driver's face in real-time, focusing on eye movements and head position to identify signs of fatigue. The captured images are processed on a Raspberry Pi, where a trained deep learning model detects drowsiness patterns, such as prolonged eye closure. If drowsiness is detected, an alert is immediately generated through a speaker or buzzer to prompt the driver to stay alert. The system leverages machine learning libraries like OpenCV and TensorFlow to ensure accurate and efficient processing in various lighting conditions, making it suitable for real-world driving scenarios.

4.1. Architecture Diagram

This overview of driver drowsiness detection system can be obtained as a structural layout showing the components and how they interface.

For the driver drowsiness detection system, presented in the architecture diagram illustrates the usage of the camera module and Raspberry Pi, as well as speaker, and the AI model responsible for the facial expression. It depicts the streaming of data from the camera to the AI model for processing which then passes data to the speaker to produce alert. Also, it points out how these components are integrated to enhance the overall capabilities of the system so as to enable the early detection of drowsiness or issuance of alerts as required.

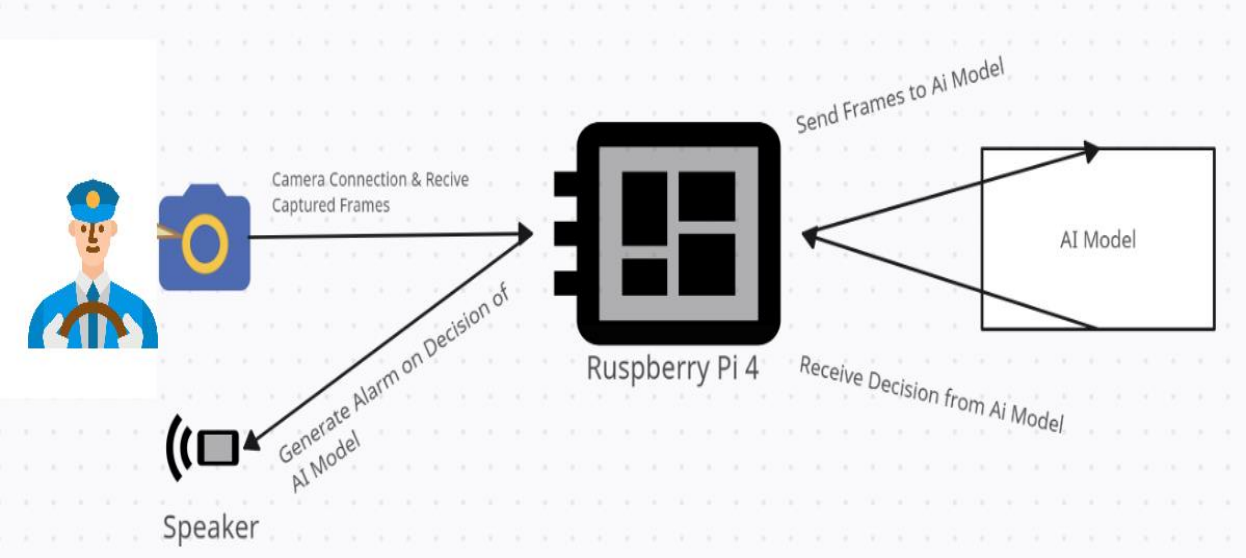


Figure 4 Architecture Diagram

4.2. Entity Relationship Diagram with data dictionary

The drivers' drowsiness detection system Entity Relationship Diagram (ERD) illustrates the entities and connections within the system.

Some of the entities likely to be on the ERD for the driver drowsiness detection system are the Facial Expression Frame and Facial Expression Recognition. The Facial Expression Frame entity is the process of capturing facial expression frames from the camera module, and the Facial Expression Recognition entity harnesses frames to identify expressions as well as drowsiness. The dependency of these entities is one to one meaning that each Facial Expression Frame is owned by one Facial Expression Recognition instance. The uses of this diagram include providing a clear illustration of the flow and connectivity of data alongside the various components in the compiling scheme, making it easy to design and implement the database accurately to meet its intended purpose.

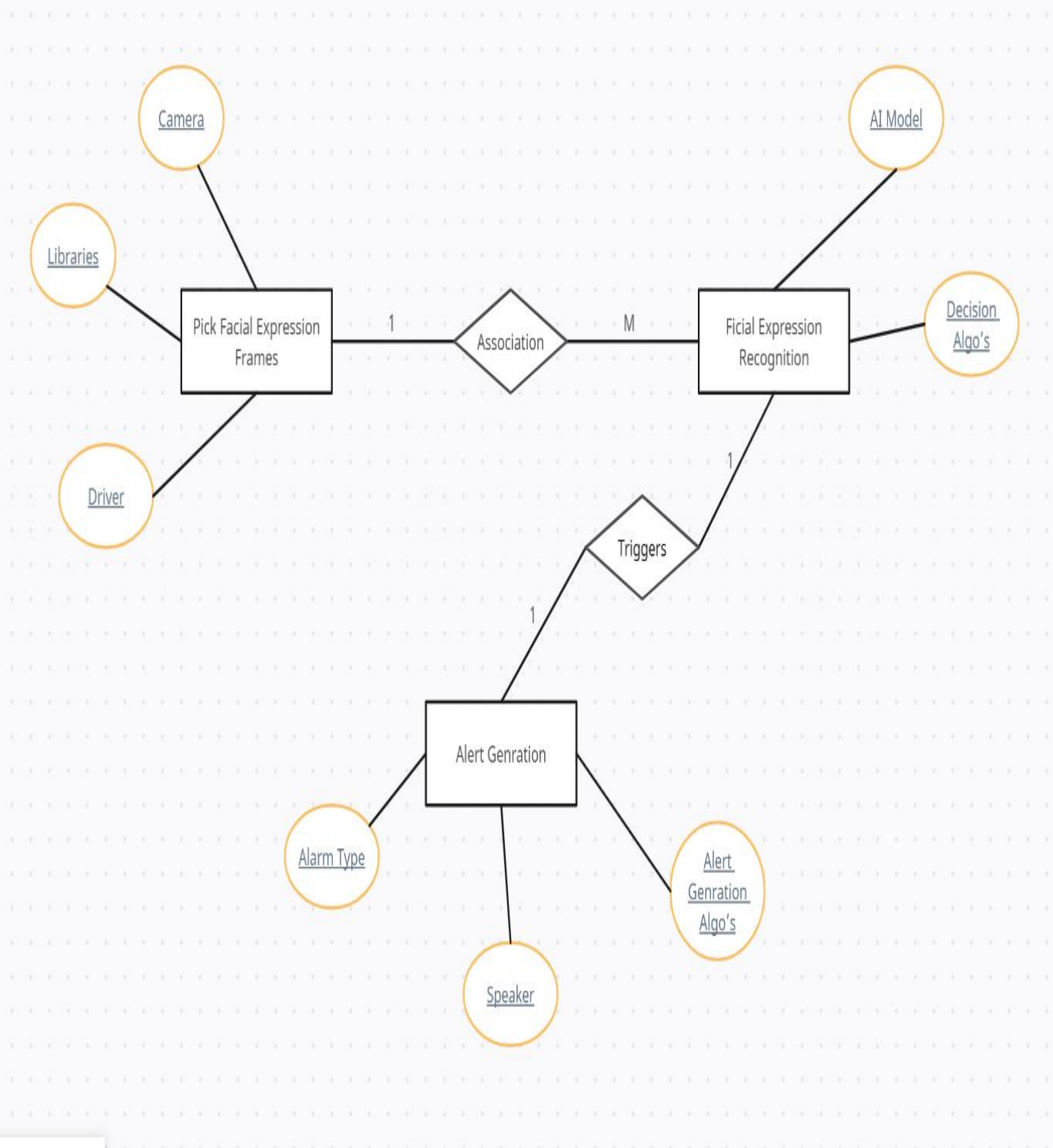


Figure 5 Entity Relationship Diagram

4.3. Class Diagram

The class diagram delineates the system's classes, attributes, methods, and associations, offering insights into the system's structure and behavior.

Within the class diagram for the driver drowsiness detection system, classes such as Driver, Facial Expression Detector, Alert Generator, and Calibration Manager are depicted, each encapsulating specific functionality. The attributes and methods associated with these classes outline their properties and behaviors, while the associations between classes illustrate the relationships and dependencies within the system architecture. This diagram provides a detailed representation of the system's class structure and interactions, facilitating effective system design and implementation.

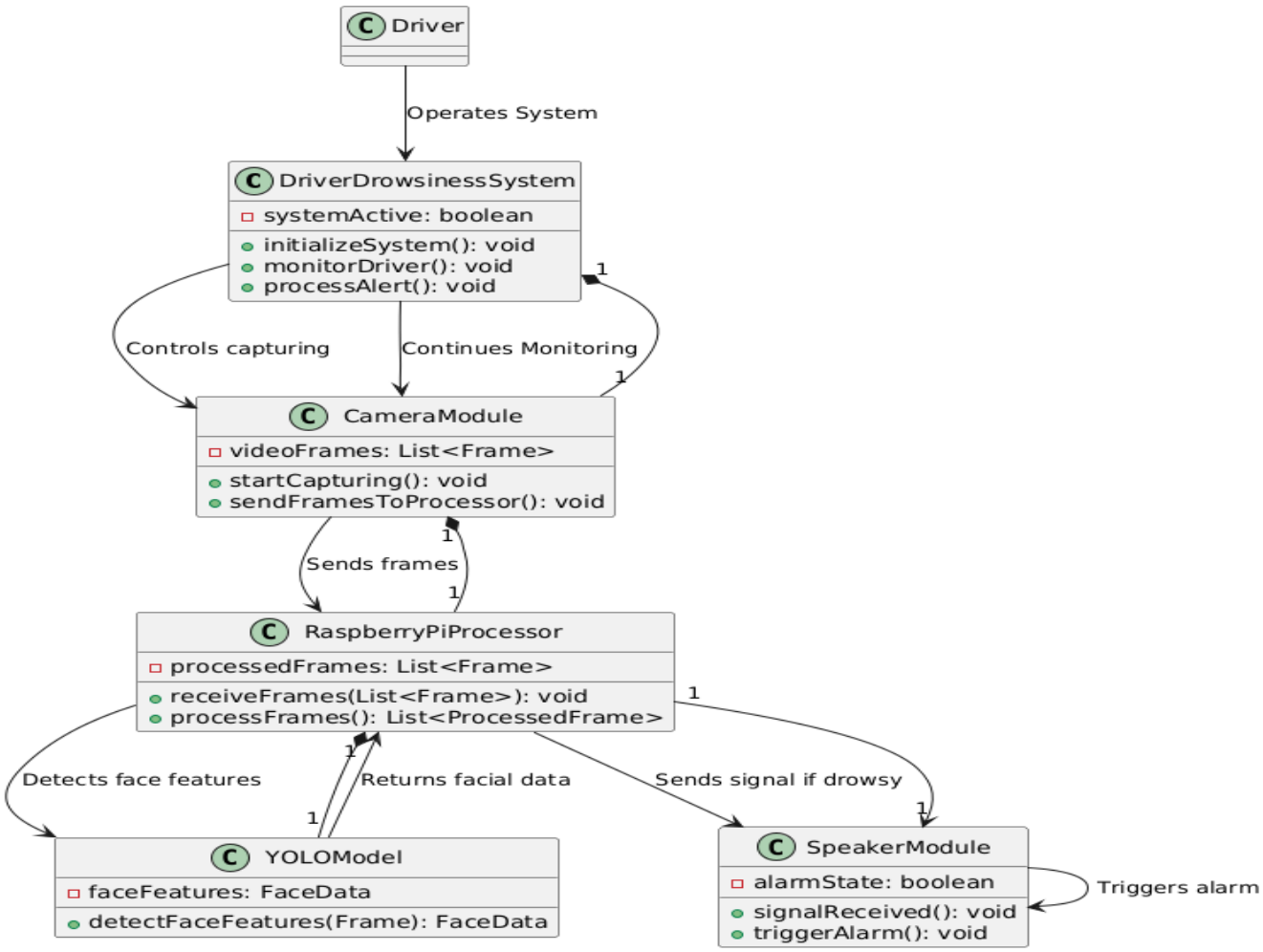


Figure 6 Class Diagram

4.4. Sequence / Collaboration Diagram

The sequence or collaboration diagram illustrates the dynamic interactions and message exchanges between system components during runtime.

In the sequence or collaboration diagram for the driver drowsiness detection system, the sequential flow of actions between the Driver, System, and Alert Generator is depicted. It showcases how the system processes facial expression data captured by the camera, analyzes it using the AI model, and triggers alerts if drowsiness is detected. The diagram highlights the communication and coordination between system components, providing a visual representation of the system's runtime behavior and message exchanges.

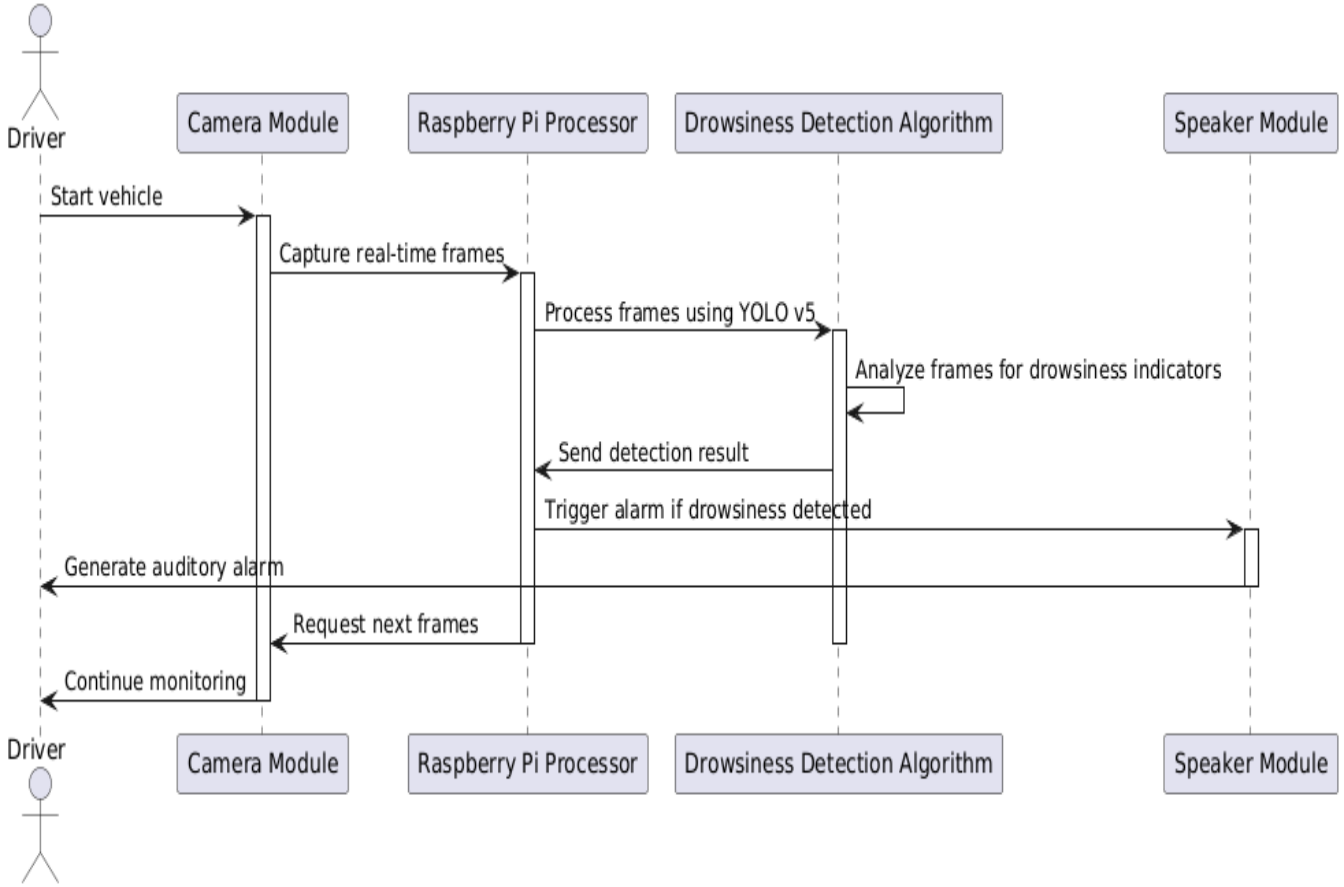


Figure 7 Sequence / Collaboration Diagram

4.5. Deployment Diagram

Description: The Deployment Diagram shows to which physical nodes the hardware and software of the components in the driver drowsiness detection system should be deployed.

Topographically, the deployment diagram below shows the nodes of the system which include the camera module, Raspberry Pi, and speaker used for the driver drowsiness detection system. The silicon-level facial expression recognition is implemented in the software component in the figure ensconced within the Raspberry Pi. Relations are to define the linkage between the nodes which in this case refers to the cables or else connection media which link the pieces of hardware. This diagram assists an analyst in understanding the system architecture and the physical distribution of the components by illustrating how the system has been mapped across different physical nodes.

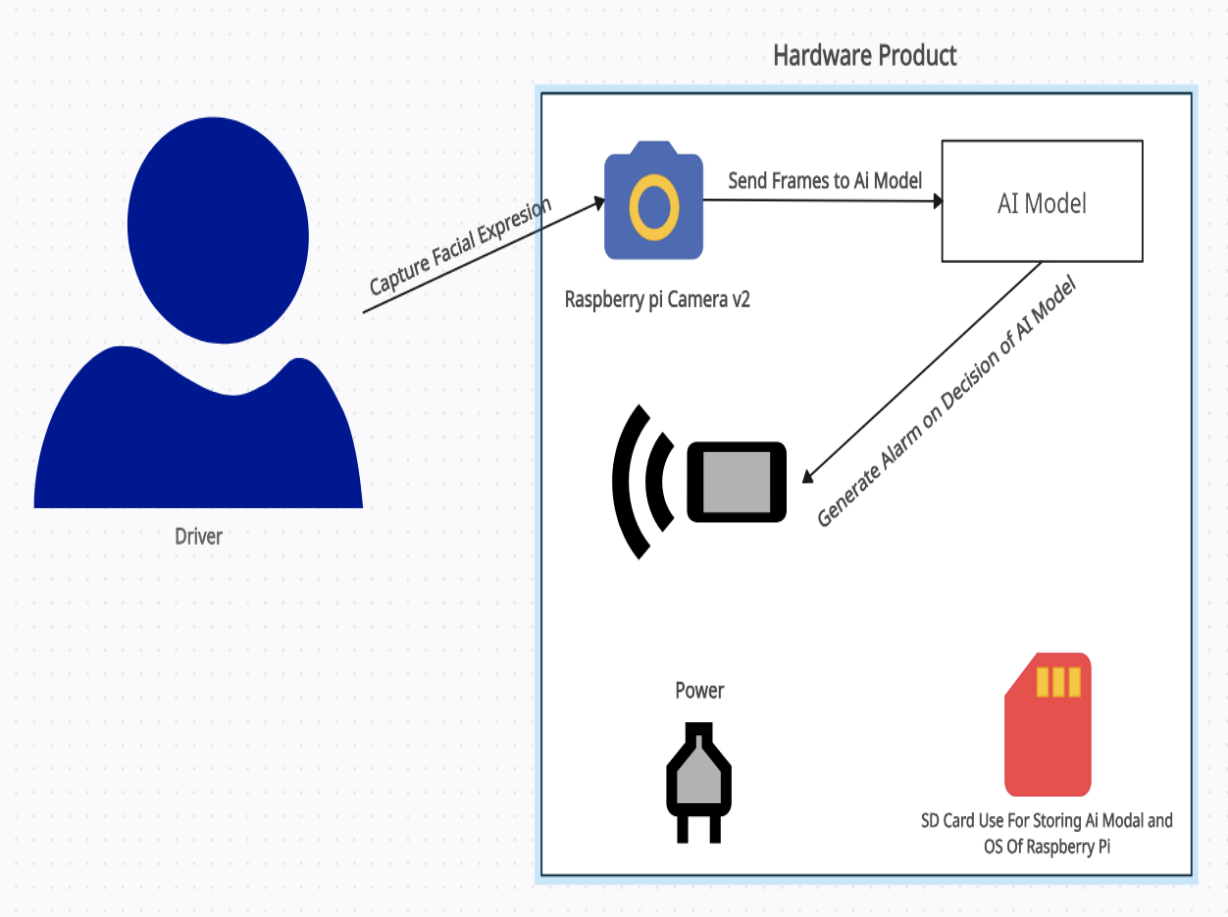


Figure 8 Deployment Diagram

Chapter 5

Implementation

Chapter 5: Implementation

5.1. Important Flow Control/Pseudo codes

Imports for Train Model:

```
from tensorflow.keras.layers import Input, Lambda, Dense, Flatten, Conv2D, MaxPooling2D, Dropout
from tensorflow.keras.models import Model
from tensorflow.keras.models import Sequential
from keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf
import numpy as np
import pandas as pd
import os
import cv2
```

Take only face(For yawn and not_yawn):

```
def face_for_yawn(direc=r"archive\train",
face_cas_path=r"archive(1)\haarcascade_frontalface_default.xml"):
    yaw_no = []
    IMG_SIZE = 145
    categories = ["yawn", "no_yawn"]
    for category in categories:
        path_link = os.path.join(direc, category)
        class_num1 = categories.index(category)
        print(class_num1)
        for image in os.listdir(path_link):
            image_array = cv2.imread(os.path.join(path_link, image), cv2.IMREAD_COLOR)
            face_cascade = cv2.CascadeClassifier(face_cas_path)
            faces = face_cascade.detectMultiScale(image_array, 1.3, 5)
            for (x, y, w, h) in faces:
                img = cv2.rectangle(image_array, (x, y), (x+w, y+h), (0, 255, 0), 2)
                roi_color = img[y:y+h, x:x+w]
                resized_array = cv2.resize(roi_color, (IMG_SIZE, IMG_SIZE))
                yaw_no.append([resized_array, class_num1])
    return yaw_no

yawn_no_yawn = face_for_yawn()
```

For Open and Closed Eye:

```
def get_data(dir_path=r"archive\train", face_cas=r"archive(1)\haarcascade_frontalface_default.xml",
eye_cas=r"archive(1)\haarcascade.xml"):
    labels = ['Closed', 'Open']
    IMG_SIZE = 145
    data = []
    for label in labels:
        path = os.path.join(dir_path, label)
        class_num = labels.index(label)
        class_num +=2
        print(class_num)
        for img in os.listdir(path):
            try:
                img_array = cv2.imread(os.path.join(path, img), cv2.IMREAD_COLOR)
                resized_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))
                data.append([resized_array, class_num])
            except Exception as e:
                print(e)
    return data
```

Data Augmentation:

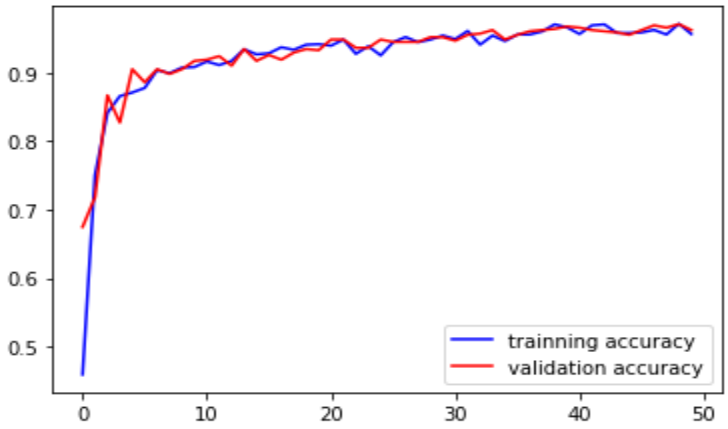
```
train_generator = ImageDataGenerator(rescale=1/255, zoom_range=0.2, horizontal_flip=True,
rotation_range=30)
test_generator = ImageDataGenerator(rescale=1/255)


#train_generator = tf.data.Dataset.from_tensor_slices((X_train, y_train))
#test_generator = tf.data.Dataset.from_tensor_slices((X_test, y_test))

train_generator = train_generator.flow(np.array(X_train), y_train, shuffle=False)
test_generator = test_generator.flow(np.array(X_test), y_test, shuffle=False)
```

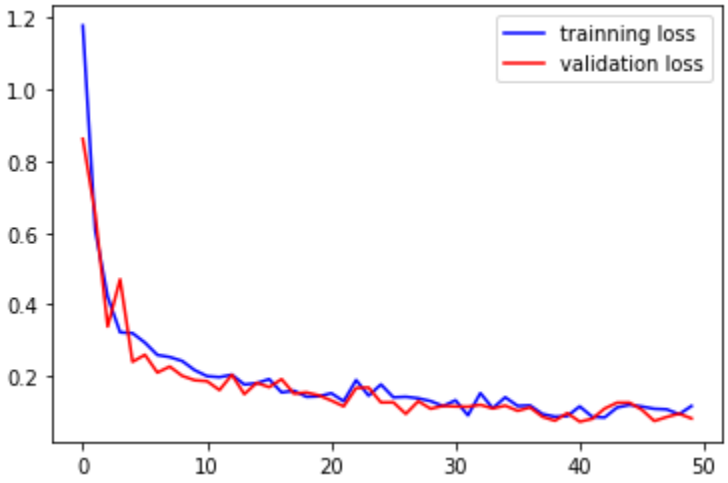
Model:

```
model = Sequential()

model.add(Conv2D(256, (3, 3), activation="relu", input_shape=(145,145,3)))
model.add(MaxPooling2D(2, 2))

model.add(Conv2D(128, (3, 3), activation="relu"))
model.add(MaxPooling2D(2, 2))

model.add(Conv2D(64, (3, 3), activation="relu"))
model.add(MaxPooling2D(2, 2))

model.add(Conv2D(32, (3, 3), activation="relu"))
model.add(MaxPooling2D(2, 2))

model.add(Flatten())
model.add(Dropout(0.5))

model.add(Dense(64, activation="relu"))
model.add(Dense(4, activation="softmax"))

model.compile(loss="categorical_crossentropy", metrics=["accuracy"], optimizer="adam")

model.summary()
```

History:

```
accuracy = history.history['accuracy']
val_accuracy = history.history['val_accuracy']
loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = range(len(accuracy))

plt.plot(epochs, accuracy, "b", label="trainning accuracy")
plt.plot(epochs, val_accuracy, "r", label="validation accuracy")
plt.legend()
plt.show()

plt.plot(epochs, loss, "b", label="trainning loss")
plt.plot(epochs, val_loss, "r", label="validation loss")
plt.legend()
plt.show()
```


0.9
0.8
0.7
0.6
0.5
0
10
20
30
40
50
trainning accuracy
validation accuracy


1.2
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40
50
trainning loss
validation loss

Test Drowsiness:

Imports, Load Model and Load Sound:

```
import cv2
import numpy as np

from keras.models import load_model
from keras.preprocessing.image import img_to_array
from playsound import playsound
from threading import Thread

def start_alarm(sound):
    """Play the alarm sound"""
    playsound('data/alarm.mp3')

classes = ['Closed', 'Open']
face_cascade = cv2.CascadeClassifier("data/haarcascade_frontalface_default.xml")
left_eye_cascade = cv2.CascadeClassifier("data/haarcascade_lefteye_2splits.xml")
right_eye_cascade = cv2.CascadeClassifier("data/haarcascade_righteye_2splits.xml")
cap = cv2.VideoCapture(0)
model = load_model("drowiness_new7.h5")
count = 0
alarm_on = False
alarm_sound = "data/alarm.mp3"
status1 = ''
status2 = ''
```

Capture Frames, Detect Drowsiness and Take Decision:

```
while True:
    _, frame = cap.read()
    height = frame.shape[0]
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 1)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = frame[y:y+h, x:x+w]
        left_eye = left_eye_cascade.detectMultiScale(roi_gray)
        right_eye = right_eye_cascade.detectMultiScale(roi_gray)
        for (x1, y1, w1, h1) in left_eye:
            cv2.rectangle(roi_color, (x1, y1), (x1 + w1, y1 + h1), (0, 255, 0), 1)
            eye1 = roi_color[y1:y1+h1, x1:x1+w1]
            eye1 = cv2.resize(eye1, (145, 145))
            eye1 = eye1.astype('float') / 255.0
            eye1 = img_to_array(eye1)
            eye1 = np.expand_dims(eye1, axis=0)
            pred1 = model.predict(eye1)
            status1=np.argmax(pred1)
            #print(status1)
            #status1 = classes[pred1.argmax(axis=-1)[0]]
            break
```

```
for (x2, y2, w2, h2) in right_eye:
        cv2.rectangle(roi_color, (x2, y2), (x2 + w2, y2 + h2), (0, 255, 0), 1)
        eye2 = roi_color[y2:y2 + h2, x2:x2 + w2]
        eye2 = cv2.resize(eye2, (145, 145))
        eye2 = eye2.astype('float') / 255.0
        eye2 = img_to_array(eye2)
        eye2 = np.expand_dims(eye2, axis=0)
        pred2 = model.predict(eye2)
        status2=np.argmax(pred2)
        #print(status2)
        #status2 = classes[pred2.argmax(axis=-1)[0]]
        break

    # If the eyes are closed, start counting
    if status1 == 2 and status2 == 2:
    #if pred1 == 2 and pred2 == 2:
        count += 1
        cv2.putText(frame, "Eyes Closed, Frame count: " + str(count), (10, 30),
cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 255), 1)
        # if eyes are closed for 10 consecutive frames, start the alarm
        if count >= 5:
            cv2.putText(frame, "Drowsiness Alert!!!", (100, height-20),
cv2.FONT_HERSHEY_COMPLEX, 1, (0, 0, 255), 2)
            if not alarm_on:
                alarm_on = True
                # play the alarm sound in a new thread
                t = Thread(target=start_alarm, args=(alarm_sound,))
                t.daemon = True
                t.start()
    else:
        cv2.putText(frame, "Eyes Open", (10, 30), cv2.FONT_HERSHEY_COMPLEX, 1, (0, 255,
0), 1)
        count = 0
        alarm_on = False

  cv2.imshow("Drowsiness Detector", frame)

  if cv2.waitKey(1) & 0xFF == ord('q'):
    break
```

5.2. Components, Libraries

- numpy
- pandas
- tenserflow
- tenserflow keras
- Open CV
- playsound
- Camera
- Raspberry Pi
- AI Model
- Speaker
- OS

5.3. Deployment Environment

- Vehicle Integration
- Onboard Hardware
- Embedded Operating System
- Power Supply
- Integration with Vehicle Systems
- Communication Interfaces
- Data Storage and Logging

5.4. Tools and Techniques

- Programming Languages (Python, Scripts of Raspberry pi 3 OS)

- Machine Learning Frameworks (ML Model for facial expression recognition, Tensor Flow, Keras, Open CV)
- Raspberry Pi Development Environment
- Integrated Development Environment (Pycharm, Google Colab)
- Data Collection Tools
- Training and Testing Framework
- Real-Time Monitoring and Alert Systems
- Deployment and Containerization Tools

5.5. Version Control

- Github

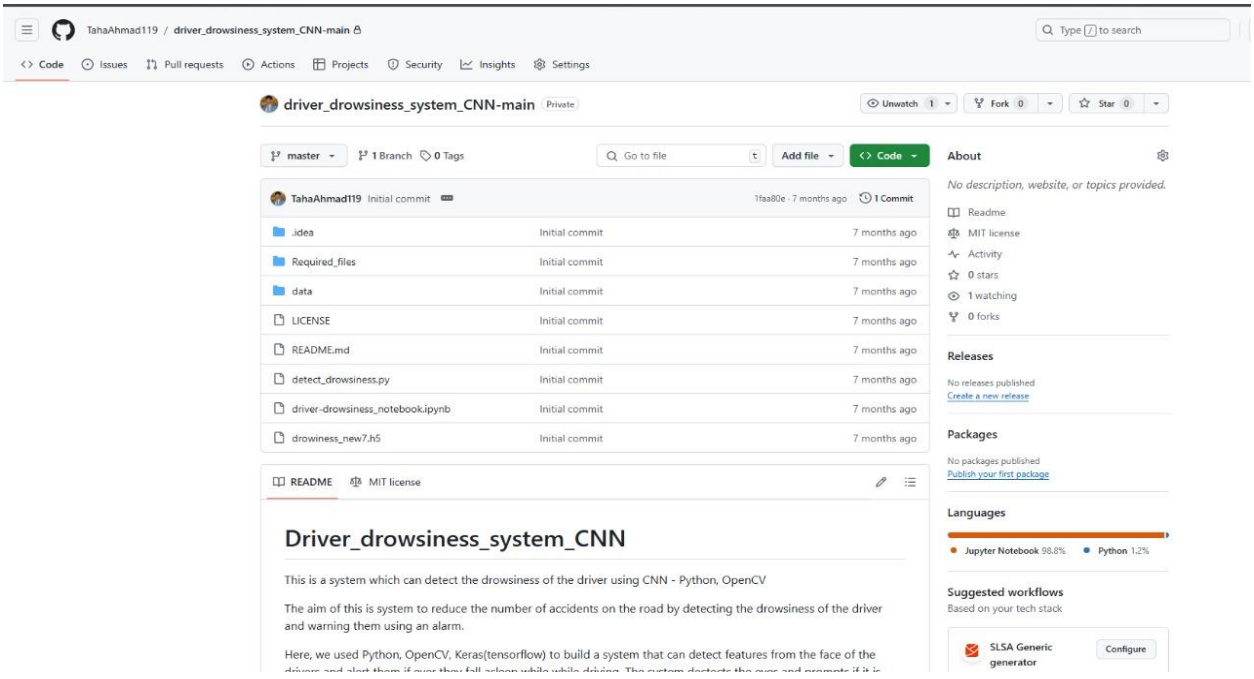


Figure 9 Version Control 1

```
import cv2
import numpy as np
from keras.models import load_model
from keras.preprocessing.image import img_to_array
from playsound import playsound
from threading import Thread

def start_alarm(sound):
    """Play the alarm sound"""
    playsound('data/alarm.mp3')

classes = ['Closed', 'Open']
face_cascade = cv2.CascadeClassifier("data/haarcascade_frontalface_default.xml")
left_eye_cascade = cv2.CascadeClassifier("data/haarcascade_lefteye_2splits.xml")
right_eye_cascade = cv2.CascadeClassifier("data/haarcascade_righteye_2splits.xml")
cap = cv2.VideoCapture(0)
model = load_model("drowiness_new7.h5")
count = 0
alarm_on = False
alarm_sound = "data/alarm.mp3"
status1 = ''
status2 = ''

while True:
    _, frame = cap.read()
    height = frame.shape[0]
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 1)
        roi_gray = gray[y:y+h, x:x+w]
        roi_color = frame[y:y+h, x:x+w]
        left_eye = left_eye_cascade.detectMultiScale(roi_gray)
        right_eye = right_eye_cascade.detectMultiScale(roi_gray)
        for (x1, y1, w1, h1) in left_eye:
            cv2.rectangle(roi_color, (x1, y1), (x1 + w1, y1 + h1), (0, 255, 0), 1)
            eye1 = roi_color[y1:y1+h1, x1:x1+w1]
```

Figure 10 Version Control 2

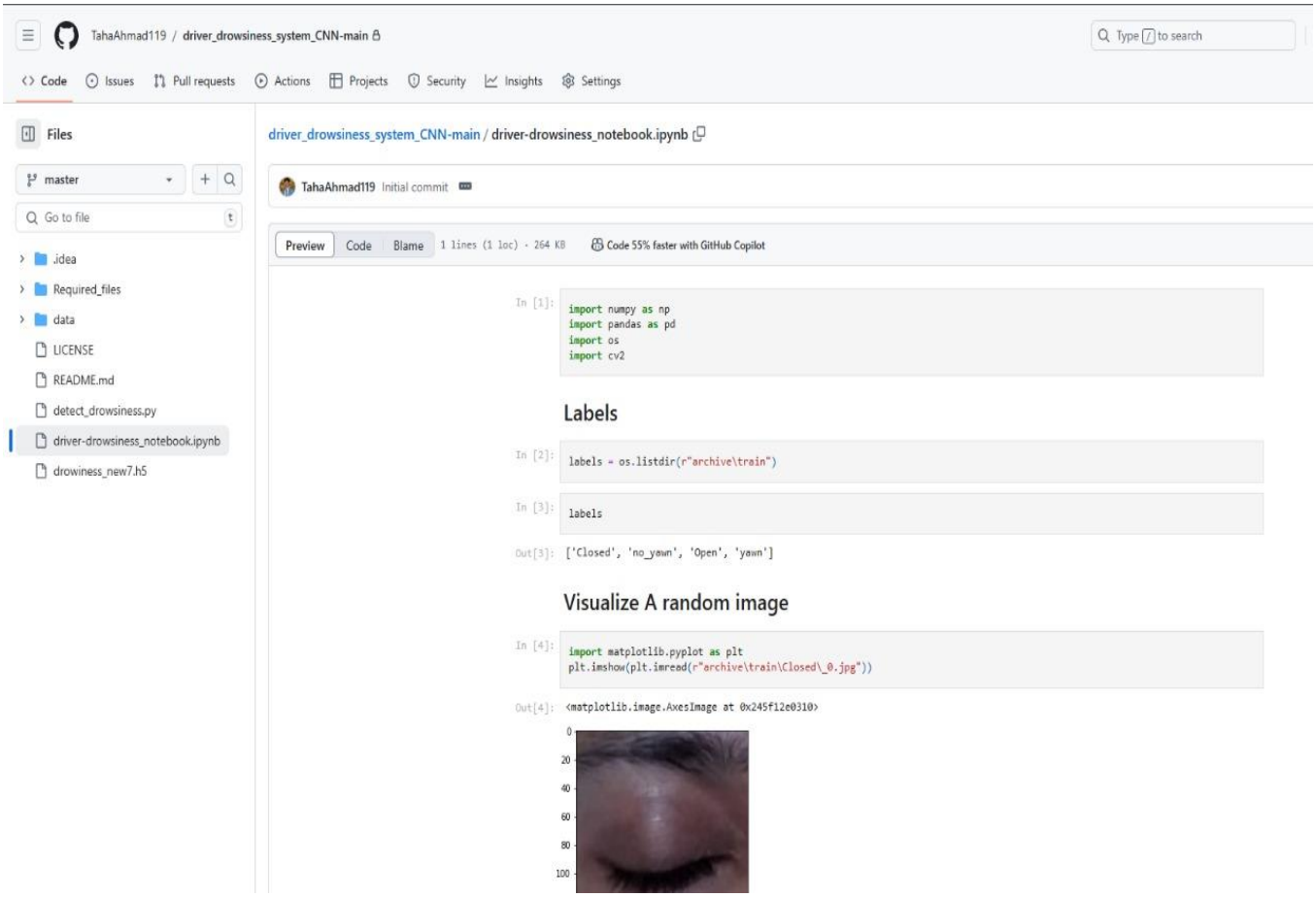


Figure 91 Version Control 3

Chapter 6

Testing and Evaluation

Chapter 6: Testing and Evaluation

Testing Overview

The testing process involves verifying individual system components, integration between components, and the overall system performance. The main objectives are to:

- Ensure accurate drowsiness detection.
- Validate reliable alert generation.
- Assess real-time processing efficiency.
- Verify user interactions and customizable settings.
- Ensure system stability and error handling.

6.1. Use Case Testing

TC ID	Scenario to Test	Pre-requisite	Steps	Expected Results	Actual Result	Pass/Fail
TC1	Detect Drowsiness	System is operational and driver is present.	1. Start the vehicle. 2. Position the camera. 3. Monitor facial expressions.	Captures image and detects signs of drowsiness (e.g., yawning, closed eyes).	The system captured the facial image and detected yawning.	Pass
TC2	Generate Alarm	Drowsiness has been detected.	1. Wait for drowsiness detection. 2. Observe alarm response.	Triggers auditory and visual alerts for drowsiness.	The auditory alarm triggered and visual alerts activated.	Pass
TC3	User Interaction	System is operational and driver is present	1. Access interface. 2. Adjust alert settings. 3. Acknowledge alert.	Successfully adjusts settings and acknowledges alerts.	The driver adjusted settings and acknowledged alerts without issues.	Pass

TC4	System Calibration	System is operational.	1. Start calibration mode. 2. Perform prompts (e.g., yawn). 3. Confirm calibration	Calibrates based on actions and updates thresholds.	The system calibrated successfully based on the driver's actions.	Pass

6.2. Equivalence partitioning

For the Driver Drowsiness Detection System, equivalence partitioning is applied to categorize input scenarios related to facial expression detection and alert settings. The following partitions are identified:

Equivalence Class	Description	Expected Outcome
Valid Facial Expressions	Yawning, eyes closed, drowsy face	System should detect drowsiness
Invalid Facial Expressions	Smiling, looking away, neutral expression	System should not detect drowsiness
Valid Alert Settings	Low, medium, high sensitivity for alerts	Alerts should function based on set sensitivity
Invalid Alert Settings	Non-numeric values, extreme values	System should reject the input and maintain default settings

6.3. Boundary value analysis

Boundary value analysis is utilized in the Driver Drowsiness Detection System to assess the limits of input scenarios related to drowsiness detection. This approach focuses on testing threshold

values, such as the duration of yawning and closed eyes, to ensure accurate detection and alert responses. By evaluating edge cases, the system's reliability is validated at its operational limits. This analysis helps identify potential issues in detection and alert mechanisms, ensuring effective driver monitoring.

Parameter	Boundary Values	Expected Outcome
Yawning Duration	0 seconds (not detected), 1 second (just detected), 5 seconds (alert triggered)	System should trigger an alert when yawning lasts 5 seconds.
Eyes Closed Duration	0 seconds (not detected), 1 second (just detected), 3 seconds (alert triggered)	System should trigger an alert when eyes are closed for 3 seconds.
Sensitivity Level	Low (1), Medium (2), High (3)	System should function correctly at each sensitivity level.
Calibration Tasks	0 tasks performed (no calibration), 1 task (minimum), 5 tasks (maximum)	System should calibrate successfully with 1 to 5 tasks performed.

6.4. Data flow testing

In the Driver Drowsiness Detection System, data flow testing examines the movement of data between key components:

- **Data Capture**: The camera module captures real-time facial data of the driver, focusing on key indicators like eye closure and yawning frequency.
- **Data Processing**: This facial data is processed by the AI model, which analyzes indicators of drowsiness.
- **Alert Generation**: If drowsiness indicators are confirmed, the processed data triggers both auditory and visual alerts to warn the driver.

- **User Interaction**: The driver can interact with alert settings, which are updated in the system and reflected in future alert responses.

Data flow testing in this context ensures that each component processes data efficiently and reliably, maintaining the system's accuracy in detecting and responding to drowsiness in real-time.

6.5. Unit testing

In the Driver Drowsiness Detection System, unit testing ensures that individual components work correctly in isolation:

- **Camera Module Test**: Verifies that the camera accurately captures clear images in various lighting conditions.
- **Drowsiness Detection Algorithm Test**: Checks if the AI model can correctly identify signs of drowsiness, such as eye closure or yawning, from captured images.
- **Alert System Test**: Confirms that auditory and visual alerts trigger accurately when drowsiness is detected.
- **User Settings Interaction Test**: Ensures the system correctly applies driver adjustments to alert settings and responds accordingly.

These unit tests ensure each component functions independently, providing reliable drowsiness detection and alert generation.

TC ID	Component	Scenario to Test	Expected Result	Actual Result
TC1	Camera Module	Image capture under various lighting	Camera captures clear images regardless of lighting.	Camera successfully captured clear images.

TC2	Drowsiness Detection Algorithm	Detection of drowsiness indicators	System detects closed eyes or yawning accurately.	System detected closed eyes and yawning correctly.
TC3	Alert System	Alarm response upon detecting drowsiness	Auditory and visual alerts activate upon drowsiness detection.	System produced auditory and visual alerts as expected.
TC4	User Settings Interaction	Adjustment of alert settings	System saves and applies new alert settings effectively.	System successfully saved and applied new settings.

6.6. Integration testing

Integration testing is the process of testing the interactions and data flow between different components of a system to ensure they work together as intended. In the Driver Drowsiness Detection System, this involves testing how the facial recognition module integrates with the alert system and the user interface. It aims to identify any issues that arise when combining components, such as data miscommunication or failures in alert generation. This phase is crucial for ensuring that the overall system functions correctly when all parts are integrated.

TC ID	Component	Scenario to Test	Expected Result	Actual Result
TC1	Camera Module & Drowsiness Detection Algorithm	Detecting drowsiness from captured image	System detects drowsiness indicators from real-time images.	System accurately detected drowsiness in real-time images.
TC2	Drowsiness Detection & Alert System	Triggering alarm upon drowsiness detection	System triggers auditory and visual alerts upon detecting drowsiness.	System produced both alerts upon detection.
TC3	User Settings & Alert System	Modifying alert settings and triggering alerts	System applies modified settings and adjusts alarm behavior.	Alerts triggered based on modified settings
TC4	System Calibration & Detection Algorithm	Calibrating detection based on user input	System adjusts sensitivity to detection based on calibration inputs.	Sensitivity adjusted according to calibration inputs.

6.7. Performance testing

1. **Response Time:** Measure the time taken by the system to detect drowsiness after capturing the driver's facial image. The goal is to ensure the detection occurs within a predefined threshold (e.g., less than 2 seconds).

2. **Accuracy of Detection:** Test the system's accuracy in identifying drowsiness indicators (e.g., yawning, closed eyes) under various lighting conditions and angles. A target accuracy of above 90% should be established.

3. **Scalability:** Assess how the system performs when multiple drivers are being monitored in a simulated environment. The system should maintain performance without lag when handling multiple input streams.

4. **Alert Trigger Time:** Measure the time taken to trigger auditory and visual alerts once drowsiness is detected. This should also fall within a predefined threshold (e.g., within 1 second of detection).

5. **Resource Utilization:** Monitor CPU and memory usage during operation to ensure that the system does not exceed resource limits, particularly on the Raspberry Pi. This includes evaluating system performance under varying loads.

6. **User Interaction Latency:** Evaluate the time taken for the system to respond to user inputs when adjusting settings or acknowledging alerts. Aim for a response time of under 1 second for a smooth user experience.

6.8. Stress Testing

Stress Testing Results

High Load Simulation: Test the system’s performance under maximum expected load conditions, such as monitoring multiple drivers simultaneously (if applicable) or running the drowsiness detection algorithm continuously for an extended period (e.g., 24 hours) to assess stability.

Input Variability: Evaluate how the system handles unexpected or extreme input conditions, such as rapid changes in lighting (e.g., transitioning from bright sunlight to darkness), different facial angles, or unexpected facial expressions to determine if it still accurately detects drowsiness.

Resource Exhaustion: Intentionally simulate low memory or CPU availability on the Raspberry Pi to observe how the system reacts. This includes running other resource-intensive applications concurrently to assess if the system maintains its functionality.

Extended Operation Time: Run the system for prolonged periods (e.g., several days) without restarting to test for memory leaks or performance degradation. Monitor if the system maintains its detection accuracy and responsiveness over time.

Rapid Alert Triggering: Simulate a scenario where drowsiness is detected repeatedly within short intervals to test if the alert system can handle rapid firing of alarms without failures or lag.

Network Failure: If applicable, assess how the system behaves when it loses network connectivity (if dependent on external servers or cloud services) during operation. Check if it can continue functioning and provide alerts locally.

Stress on Calibration Functionality: Test the calibration feature by rapidly changing calibration parameters to see if the system can handle dynamic adjustments without crashing or producing erroneous behavior.

Hardware Limitations: Stress test by overloading peripheral components (e.g., by using an external camera with higher resolution) to observe if the system can adapt without failures or significant performance drops.

Chapter 7

Summary, Conclusion and Future Enhancements

Chapter 7: Summary, Conclusion & Future Enhancements

7.1. **Project Summary**

Upon project completion, we successfully developed a real-time drowsiness detection system using Raspberry Pi 3, a camera module, and deep learning techniques to monitor drivers' alertness. The system efficiently detects signs of drowsiness, such as prolonged eye closure and reduced eye movement, leveraging a Convolutional Neural Network (CNN) model built on ResNet50 for high accuracy. Using libraries like OpenCV, TensorFlow, and Keras, the system identifies fatigue indicators in various conditions, including variations in eye sizes and drivers wearing glasses. Alerts are triggered through a speaker when drowsiness is detected, providing immediate feedback to drivers and helping prevent accidents caused by fatigue. Performance tests confirmed the system's reliability in diverse driving environments, showcasing its potential as a valuable addition to Intelligent Driver Assistance Systems to enhance road safety.

7.2. **Achievements and Improvements**

The project successfully developed a real-time driver drowsiness detection system using Raspberry Pi 3, a camera module, and deep learning algorithms. The system accurately identifies drowsiness indicators such as prolonged eye closure and eye movement, achieving reliable detection in varied lighting and driving conditions. Integration of the ResNet50-based CNN model ensured high precision, with minimal false alarms. Real-time alerts through the speaker provide immediate feedback to drivers, enhancing road safety. The system demonstrates robustness across different driver profiles, including those with glasses.

Future improvements include optimizing the model to reduce computational load, allowing for smoother performance on smaller hardware. Expanding the detection capabilities to include yawning or head position tracking could increase accuracy in fatigue detection. Implementing adaptive brightness control would help enhance detection in low-light conditions. The system could also benefit from multi-sensor fusion, integrating data from other vehicle sensors for a holistic safety solution. Lastly, a larger, more diverse training dataset could further improve detection accuracy across varied demographics.

7.3. Critical Review

While the drowsiness detection project achieved its primary goal of identifying driver fatigue, there are several areas that require critical evaluation. First, although the ResNet50 model provides high accuracy, its computational demands can strain the Raspberry Pi hardware, occasionally impacting real-time performance. Additionally, the system relies heavily on facial feature detection, meaning drivers with certain physical attributes, like small eye shapes or glasses, may not always be accurately monitored. Environmental factors such as poor lighting or sudden glares can also reduce detection reliability, posing challenges in real-world applications. Furthermore, limited testing conditions may impact the system's generalizability to a wide range of drivers and driving environments. Finally, while effective for drowsiness, the system could benefit from a broader scope, such as identifying other risky behaviors (e.g., distraction), to provide more comprehensive driver monitoring.

7.4. Lessons Learnt

Developing the drowsiness detection system provided valuable insights into both the technical and practical aspects of implementing real-time driver monitoring. One key lesson was the importance of optimizing machine learning models to run on limited hardware like the Raspberry Pi, as high-computation models can hinder performance without careful adjustments. Additionally, handling diverse driver characteristics, such as variations in eye shape and eyewear, underscored the need for more adaptable detection algorithms. Working with environmental variables, like lighting changes, highlighted the complexity of designing a system suitable for real-world conditions. The project also reinforced the value of extensive testing across different driving scenarios to ensure system reliability and generalizability. Overall, the experience emphasized the necessity of balancing accuracy, performance, and adaptability when creating driver assistance technologies for practical applications.

7.5. Future Enhancements/Recommendations

To further improve the drowsiness detection system, several enhancements can be considered. First, optimizing the deep learning model to reduce its computational load would allow it to operate more efficiently on devices like the Raspberry Pi. Additionally, integrating head position and yawning detection alongside eye movement analysis could provide a more comprehensive assessment of driver fatigue. Improving the system's adaptability to diverse lighting conditions through automatic brightness adjustment would increase its effectiveness in low-light and high-glare scenarios.

Another recommendation is to expand the model's dataset to include a wider range of driver profiles, covering various ethnicities, eye shapes, and glasses types for improved accuracy. The inclusion of multi-sensor fusion, such as heart rate or steering behavior data, could provide a more holistic approach to detecting drowsiness and other risky behaviors. Adding a cloud-based data storage component would allow for continuous learning and model improvement through feedback from real-world usage.

Finally, expanding the system to detect distracted driving behaviors, such as looking away from the road or using a mobile device, could broaden its functionality as a driver safety tool. Regular updates based on advancements in AI and image processing could ensure the system remains state-of-the-art, meeting evolving safety standards in intelligent transportation systems.

Appendices

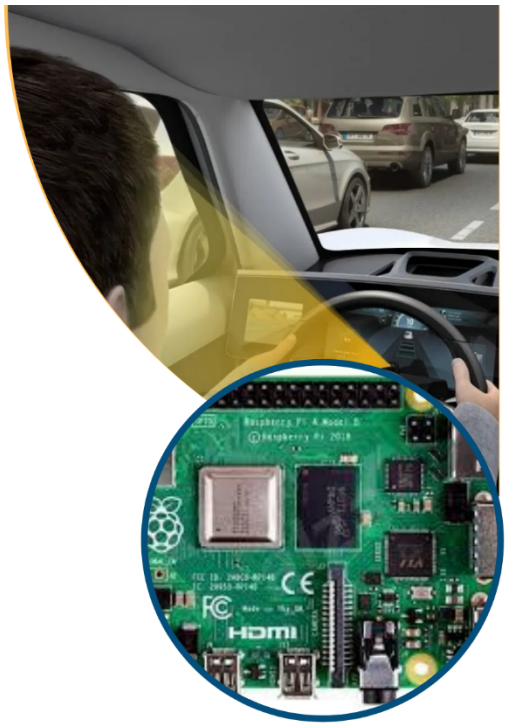
Appendix A: Information / Promotional Material

This appendix provides the promotional materials developed to raise awareness about the Driver Drowsiness Detection System. The materials include brochures, flyers, and banners aimed at informing potential users about the system benefits and features.

A.1. Broacher

A.2. Flyer

SAFE DRIVE



"STAY ALERT, DRIVE SAFE WITH AI."

"Alert Guardian" combines AI with real-time monitoring to detect driver drowsiness early, helping ensure safer journeys and proactive accident prevention.

ARTIFICIAL INTELLIGENCE

INTERNET OF THINGS

COMPUTER VISION

SUPERIOR UNIVERSITY

CMACED

9 INDUSTRY, INNOVATION AND INFRASTRUCTURE

CONTACT US

+923227149659

A.3. Standee


Eyes Direction
Eyelid Opening
SAFE DRIVE
"STAY ALERT, DRIVE SAFE WITH AI."
About the Alert Guardian:
"Alert Guardian" combines AI with real-time monitoring to detect driver drowsiness early, helping ensure safer journeys and proactive accident prevention.
Your Safety Our Priority
Python
AI
Computer Vision
IOT
COMPONENTS
Supervisor
Madam Javaria Qadeer
Fyp ID
FYP-BCSM-S24-015
• Raspberry Pi 3
• Speaker
• Camera
• SD Card
• Type-C Power Cable
Group Members
Taha Ahmad Bcsm-f20-328
Talha Saleem Bcsm-f20-320
Salman Hashmi Bcsm-f20-343
CONTACT US
+923227149659
CMACED
SUPERIOR UNIVERSITY
9 INDUSTRY, INNOVATION AND INFRASTRUCTURE

A.4. Banner


"STAY ALERT, DRIVE SAFE WITH AI."
SAFE DRIVE
Subtitle: Driver Drowsiness Detection System
"Alert Guardian" combines AI with real-time monitoring to detect driver drowsiness early, helping ensure safer journeys and proactive accident prevention.
Python
AI
Computer Vision
IOT

Reference and Bibliography

Reference and Bibliography

[1] Y. Dong, Z. Hu, K. Uchimura, and N. Murayama, “Driver inattention monitoring system for intelligent vehicles: A review,” IEEE Trans. Transp. Syst., vol. 12, no. 2, pp. 596–614, Jun. 2020.

[2] C. Bila, F. Sivrikaya, M. A. Khan, and S. Albayrak, “Vehicles of the future: A survey of research on safety issues,” IEEE Trans. Intell. Transp. Syst., vol. 18, no. 5, pp. 1046–1065, 2020.

Index

Index