A Summer Internship Report

On

SAFE DRIVE- ML BASED DISTRACTED MONITORING SYSTEM

Submitted for partial fulfillment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

in

INFORMATION TECHNOLOGY

by

G. Rashmitha	2451-21-737-002
D. Poojitha	2451-21-737-005
D. Sai Kiran Kumar	2451-21-737-055

Carried a Virtual Internship
From
AICTE Eduskills Academy

MATURI VENKATA SUBBA RAO (M.V.S.R) ENGINEERING COLLEGE

(An Autonomous Institution)

Department of Information Technology

(Affiliated to Osmania University & Recognized by AICTE)
Nadergul, SaroorNagar Mandal, Hyderabad – 501 510

AY: 2024-25

CERTIFICATE

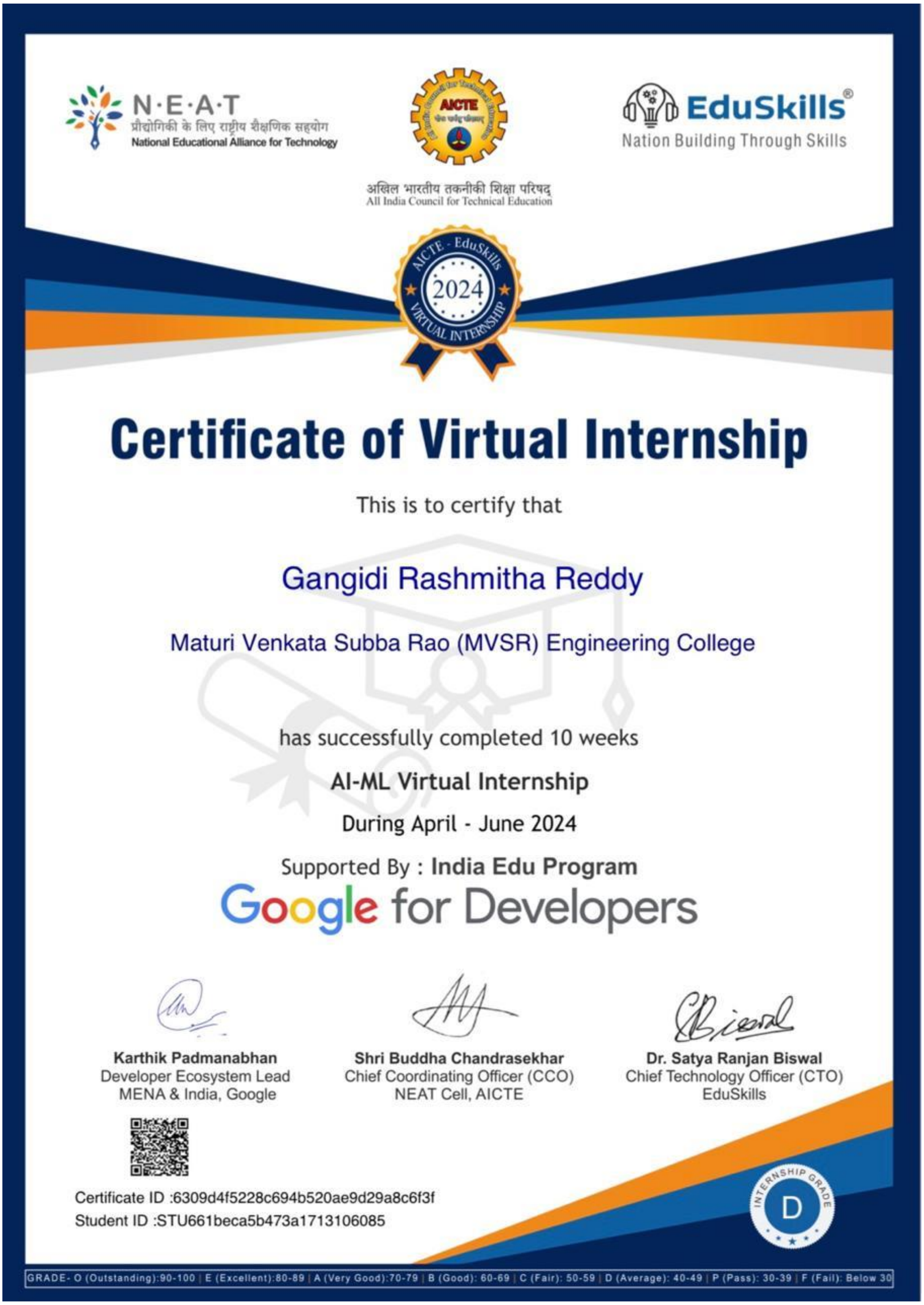
This is to certify that the Summer Internship case study entitled " ***SafeDrive – ML Based Distraction Monitoring System*** *" is a bonafide work carried out by Ms.* G. Rashmitha*(2451-21-737-002), Ms.* D. Poojitha*(2451-21-737-005) and Mr.* D. Sai Kiran Kumar *(2451-21-737-055) in partial fulfillment of the requirements for the award of the degree of Bachelor of Engineering in Information Technology from Maturi Venkata Subba Rao (M.V.S.R.) Engineering College, an Autonomous Institution, affiliated to Osmania University Hyderabad, during the Academic Year 2024-25 through virtual mode from AICTE – Eduskills in AIML.*

The results embodied in this report have not been submitted to any other university or institute for the award of any degree or diploma.

Summer Internship Coordinator
Dr. H. Jayasree
Professor, ITD

HOD
Dr. K. Venugopal Rao
Dean – Academics,
Professor & Head - ITD

External Examiner



N·E·A·T
प्रौद्योगिकी के लिए राष्ट्रीय शैक्षणिक सहयोग
National Educational Alliance for Technology

AICTE
अखिल भारतीय तकनीकी शिक्षा परिषद्
All India Council for Technical Education

EduSkills®
Nation Building Through Skills

AICTE - EduSkills
2024
VIRTUAL INTERNSHIP

Certificate of Virtual Internship

This is to certify that

Gangidi Rashmitha Reddy

Maturi Venkata Subba Rao (MVSR) Engineering College

has successfully completed 10 weeks

AI-ML Virtual Internship

During April - June 2024

Supported By : **India Edu Program**

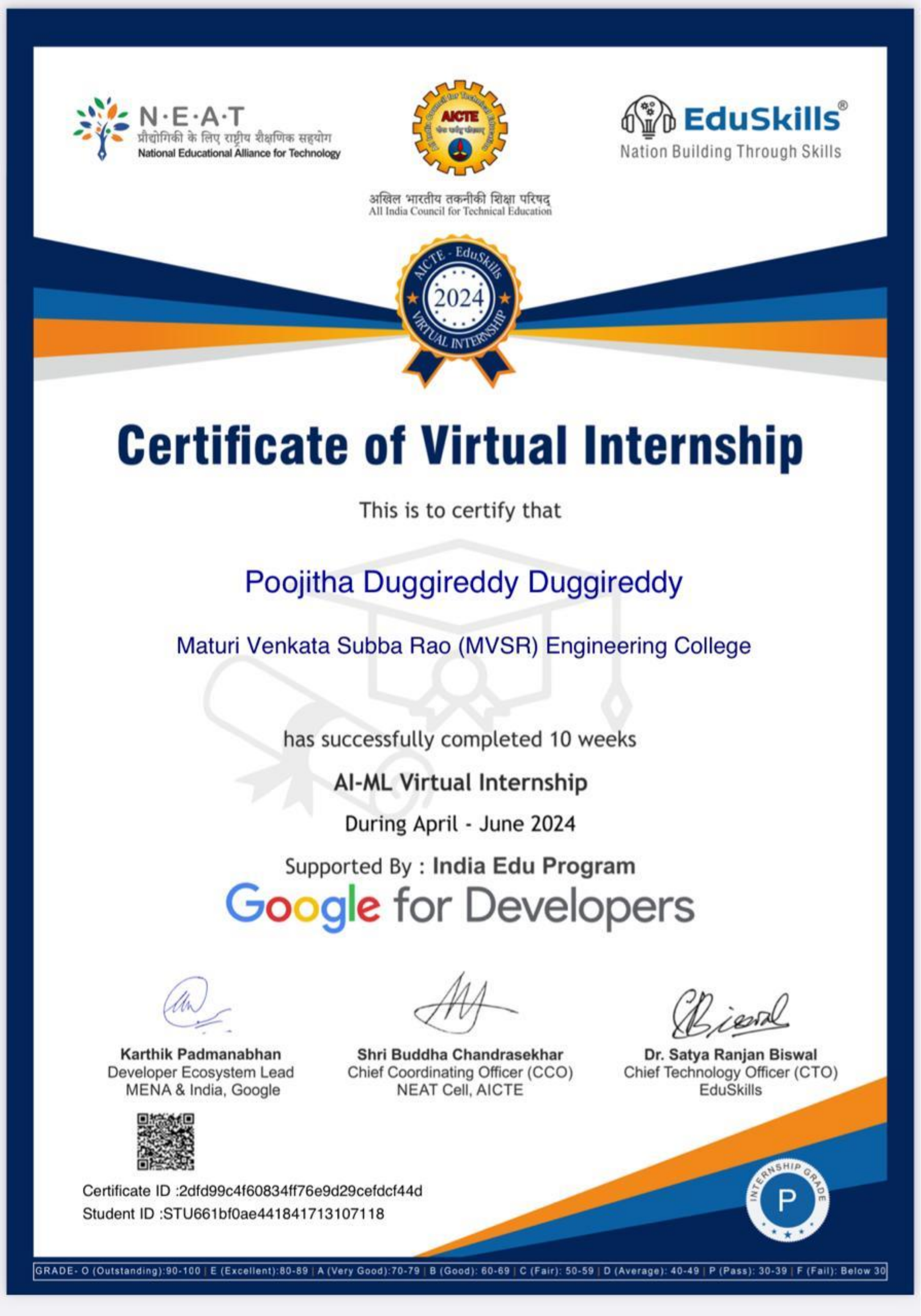
Google for Developers

Karthik Padmanabhan Developer Ecosystem Lead MENA & India, Google	**Shri Buddha Chandrasekhar** Chief Coordinating Officer (CCO) NEAT Cell, AICTE	**Dr. Satya Ranjan Biswal** Chief Technology Officer (CTO) EduSkills

Certificate ID :6309d4f5228c694b520ae9d29a8c6f3f
Student ID :STU661beca5b473a1713106085

INTERNSHIP GRADE
D

GRADE- O (Outstanding):90-100 | E (Excellent):80-89 | A (Very Good):70-79 | B (Good): 60-69 | C (Fair): 50-59 | D (Average): 40-49 | P (Pass): 30-39 | F (Fail): Below 30



N·E·A·T
प्रौद्योगिकी के लिए राष्ट्रीय शैक्षणिक सहयोग
National Educational Alliance for Technology

AICTE

अखिल भारतीय तकनीकी शिक्षा परिषद्
All India Council for Technical Education

EduSkills®
Nation Building Through Skills

AICTE - EduSkills
2024
VIRTUAL INTERNSHIP

Certificate of Virtual Internship

This is to certify that

Poojitha Duggireddy Duggireddy

Maturi Venkata Subba Rao (MVSR) Engineering College

has successfully completed 10 weeks

AI-ML Virtual Internship

During April - June 2024

Supported By : **India Edu Program**

Google for Developers

Karthik Padmanabhan
Developer Ecosystem Lead
MENA & India, Google

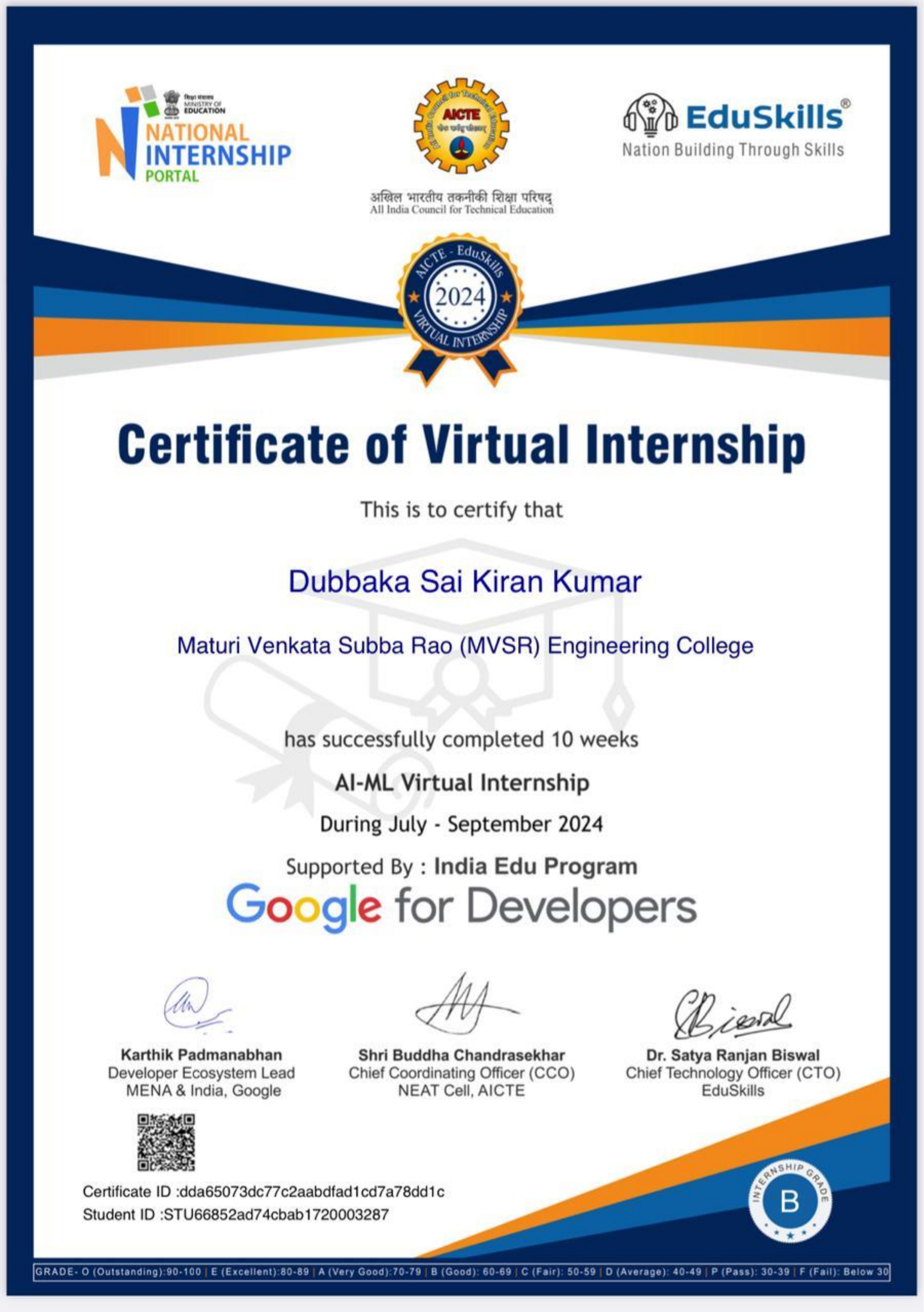
Shri Buddha Chandrasekhar
Chief Coordinating Officer (CCO)
NEAT Cell, AICTE

Dr. Satya Ranjan Biswal
Chief Technology Officer (CTO)
EduSkills

Certificate ID :2dfd99c4f60834ff76e9d29cefdcf44d
Student ID :STU661bf0ae441841713107118

INTERNSHIP GRADE
P

GRADE- O (Outstanding):90-100 | E (Excellent):80-89 | A (Very Good):70-79 | B (Good): 60-69 | C (Fair): 50-59 | D (Average): 40-49 | P (Pass): 30-39 | F (Fail): Below 30



शिक्षा मंत्रालय
MINISTRY OF EDUCATION
NATIONAL INTERNSHIP PORTAL

AICTE

अखिल भारतीय तकनीकी शिक्षा परिषद्
All India Council for Technical Education

EduSkills®
Nation Building Through Skills

AICTE - EduSkills
2024
VIRTUAL INTERNSHIP

Certificate of Virtual Internship

This is to certify that

Dubbaka Sai Kiran Kumar

Maturi Venkata Subba Rao (MVSR) Engineering College

has successfully completed 10 weeks

AI-ML Virtual Internship

During July - September 2024

Supported By : **India Edu Program**

Google for Developers

Karthik Padmanabhan
Developer Ecosystem Lead
MENA & India, Google

Shri Buddha Chandrasekhar
Chief Coordinating Officer (CCO)
NEAT Cell, AICTE

Dr. Satya Ranjan Biswal
Chief Technology Officer (CTO)
EduSkills

Certificate ID :dda65073dc77c2aabdfad1cd7a78dd1c
Student ID :STU66852ad74cbab1720003287

INTERNSHIP GRADE
B

GRADE- O (Outstanding):90-100 | E (Excellent):80-89 | A (Very Good):70-79 | B (Good): 60-69 | C (Fair): 50-59 | D (Average): 40-49 | P (Pass): 30-39 | F (Fail): Below 30

DECLARATION

This is to certify that the work reported in the present Summer Internship case study report entitled "*SafeDrive – ML Based Distraction Monitoring System*" is a record of bonafide work done by us in the Department of Information Technology, Maturi Venkata Subba Rao (M.V.S.R.) Engineering College, an Autonomous Institution, affiliated to Osmania University. The reports are based on the case study done entirely by us and not copied from any other source. The results embodied in this report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

G. Rashmitha	D. Poojitha	D. Sai Kiran Kumar
(2451-21-737-002)	(2451-21-737-005)	(2451-21-737-055)

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to Shri Buddha Chandrasekhar Chief Coordinating officer (CCO), NEAT Cell, AICTE, and Dr.Satya Ranjan Biswal Chief Technology officer (CTO), EduSkills for allowing carrying out a virtual internship through AICTE, EduSkills and giving valuable suggestions and interest throughout the internship.

We would like to express our sincere gratitude to our Summer Internship Coordinator, **Dr. H. Jayasree**, Professor, for her valuable recommendations and attention throughout the course.

We are also thankful to our Principal **Dr. Vijaya Gunturu**, **Professor** and **Dr. K. Venugopal Rao,** Dean Academics, Professor and Head – the Department of Information Technology, Maturi Venkata Subba Rao Engineering College for providing excellent infrastructure and a nice atmosphere to carry out a case study on the training obtained through internship.

Finally, we would like to take this opportunity to thank our family for their support through the work. We sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this work.

G. Rashmitha	2451-21-737-002
D. Poojitha	2451-21-737-005
D. Sai Kiran Kumar	2451-21-737-055

VISION & MISSION,

PROGRAM EDUCATIONAL OUTCOMES

Vision of the Department:

To impart technical education producing competent and socially responsible engineering professionals in the field of Information Technology.

Mission of the Department:

M1. To make the teaching learning process effective and stimulating.

M2. To provide adequate fundamental knowledge of sciences and Information Technology with positive attitude.

M3. To create an environment that enhances skills and technologies required for industry.

M4. To encourage creativity and innovation for solving real world problems.

M5. To cultivate professional ethics in students and inculcate a sense of responsibility towards society

Program Educational Objectives:

After 3 to 4 years of graduation, graduates of the Information Technology program will:

I. Apply knowledge of mathematics and Information Technology to analyze, design and implement solutions for real world problems in core or in multidisciplinary areas.

II. Communicate effectively, work in a team, practice professional ethics and

III. Engage in Professional development or postgraduate education to be a life-long learner.

PROGRAM OUTCOMES & PROGRAM SPECIFIC OUTCOMES

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs):

PSO1: **Hardware design**: An ability to analyze, design, simulate and implement computer hardware / software and use basic analog/digital circuits, VLSI design for various computing and communication system applications.

PSO2: **Software design**: An ability to analyze a problem, design algorithm, identify and define the computing requirements appropriate to its solution and implement the same.

COURSE OBJECTIVES & COURSE OUTCOMES

COURSE OBJECTIVES:

Students should be able to

- To gain practical experience and develop skills relevant to their field of study or career aspirations.
- To explore the work environment and dynamics of the industry they are interested in helping them gain insights into professional practices and expectations.
- To apply theoretical knowledge gained from their academic studies to real-world projects and challenges, enhancing their understanding and competence in their chosen field.
- To build professional networks by interacting with professionals, mentors and fellow interns in their field, which can lead to future career opportunities.

COURSE OUTCOMES:

After completion of the course student will be able to:

- Demonstrate improved technical skills, problem-solving abilities, critical thinking, and other relevant skills specific to their field.
- Demonstrate hands-on experience in executing tasks, working on projects, and utilizing tools and technologies relevant to their field.
- Exhibit enhanced professionalism in areas such as communication, teamwork, time management and work ethics.
- Display increased self-confidence in their abilities, having successfully completed tasks, projects and assignments during their internship.
- Expand their professional network through interactions with colleagues, mentors and industry professionals, creating valuable connections for future career opportunities.

Overview of Internship Activity

Details of the Internship : AICTE-EduSkills – COHORT 8
Mode of the internship: Online
Duration of the internship: 10 weeks
Technology Explored through internship: Machine Learning
Domain Knowledge Explored through internship: AI-ML

Weekly Report of Internship Activity

Week No.	Activity carried out
1	Program neural networks with TensorFlow
2	Get started with object detection
3	Go further with object detection
4	Get started with product image search
5	Go further with product image search
6	Go further with image classification
7	Demonstartion of project on Images detection
8	Demonstartion of project on Image classification

ABSTRACT

Distracted driving is a major contributor to traffic accidents, responsible for thousands of fatalities each year worldwide. It occurs when a driver's attention is diverted from the primary task of driving, often due to activities like texting, eating, or talking on the phone. Despite advancements in driver assistance technologies, current systems are limited in their ability to detect and mitigate a wide range of distractions effectively. This project aims to develop an advanced machine learning model that can predict and classify different types of driver distractions by analyzing images captured inside the vehicle. The system will be capable of identifying visual, manual, and cognitive distractions, such as texting, eating, adjusting the radio, and talking on the phone. Safe Drive utilizes deep learning, specifically leveraging the VGG-16 convolutional neural network (CNN) architecture, to analyze real-time images from in-vehicle cameras and detect driver distractions.Trained on a comprehensive labeled dataset, VGG-16 classifies specific activities, such as texting or eating, and triggers voice alerts to help drivers refocus. This system could also integrate with vehicle safety features to add a proactive layer of protection, aiming to reduce distraction-related accidents and enhance road safety.

TABLE OF CONTENTS

CONTENTS	PAGE NO
Institution Certificate	i
Internship Certificate	ii-iv
Declaration	v
Acknowledgment	vi
Vision & Mission, PEOs	vii
POs and PSOs	viii-ix
Course Objectives & Course Outcomes	x
Overview of Internship Activities	Xi
Abstract	xii
List of figures	xiv
List of tables	xv
Chapters	
1 **Introduction**	1 - 2
1.1 Problem statement	
1.2 Existing system	
1.3 Proposed system	
1.4 Scope	
2 **System Requirement Specifications**	3-4
2.1 Software Requirements	
2.2 Hardware Requirements	
2.3 System Architecture / Block Diagram	
3 **System Design**	5-8
3.1 Methodology	
3.2 Structural Diagram	
3.3 Behavioural Diagrams	
4 **Results & Discussion**	9-19
4.1 Environmental Setup	
4.2 Implementation	
4.3 Results	
4.4 Test Cases	
5 **Conclusion & Future Enhancements**	20-20
5.1 Conclusion	
5.2 Future Enhancements	
Appendix	
A: source/pseudo code	21-24

LIST OF FIGURES

Figure no.	Name of the figure	Page no
2.1	Block diagram	4
3.1	Structure of dataset	5
3.2	Architecture of VGG-16	6
3.3	CLass Diagram	7
3.4	State Diagram	7
3.5	Use Case Diagram	8
3.6	Sequence Diagram	8
4.1	Visual Studio code installation	9
4.2	Python Installation	10
4.3	Python-Specific Release Installation	10
4.4	Video studio welcome page	11
4.5	Modules Installation	11
4.6	Page redirection after running the server	13
4.7	Project Directory Structure	14
4.8	Resulted web page	15

LIST OF TABLES

Table No.	Name of the Table	Page No.
4.1	Test Cases	15-19

CHAPTER-1

INTRODUCTION

Distracted driving has emerged as a critical safety concern, accounting for a substantial number of road accidents. Activities such as texting, eating, and phone usage divert a driver's focus, significantly raising the risk of collisions. The Safe Drive project seeks to tackle this issue through a machine learning-based monitoring system that can detect these distractions in real time by analyzing in-vehicle images. By issuing immediate voice alerts, the system aims to reduce unsafe driving behaviors and enhance road safety.

1.1 Problem Statement

Distracted driving is a leading cause of road accidents, as it diverts the driver's attention away from the road, resulting in increased risk of collisions. Common distractions include activities like texting, eating, talking on the phone. Current measures often fail to detect and address these behaviors effectively. There is a need for a robust approach to identify and understand different types of distractions to enhance road safety.

1.2 Existing System

Current systems for detecting and mitigating distracted driving include manual observation by traffic authorities, in-vehicle driver monitoring systems (DMS), and surveillance cameras. Traffic authorities can observe driver behaviors but are limited by their ability to cover large areas and respond in real-time. DMS, which uses cameras to monitor eye movement and head position, focus mainly on detecting fatigue but are ineffective at identifying other distractions like texting or eating. Surveillance cameras are often used for traffic enforcement but lack the ability to distinguish specific distractions. Some vehicles have integrated systems that issue warnings for minor distractions, but these systems are typically not comprehensive or accurate enough to detect all types of distractions in real time. While there are emerging AI-based systems that leverage computer vision for distraction detection, they still face challenges in terms of dataset diversity, real-time processing, and scalability.

1.3 Proposed System

The proposed Safe Drive - ML-Based Distracted Monitoring System aims to address the limitations of existing distracted driving detection methods by utilizing a machine learning model to analyze real-time images captured from in-vehicle cameras. The system will employ the VGG-16 convolutional neural network (CNN) architecture to identify and classify various types of distractions, including visual, manual, and cognitive distractions, such as texting, eating, adjusting controls, or talking on the phone. The model will be trained on a large, labeled dataset to ensure accurate classification of these behaviors. Once a distraction is detected, the system will trigger voice alerts to warn the driver and prompt them to refocus on the road. The Safe Drive system could also integrate with existing vehicle safety features like lane-keeping assist or emergency braking, adding an additional layer of proactive protection. This real-time, automated solution aims to reduce distraction-related accidents and enhance overall road safety by providing immediate feedback to drivers.

1.4 Scope

The Safe Drive system has broad applicability across various sectors. Automobile manufacturers can integrate it into vehicles to enhance in-vehicle safety by actively monitoring driver behavior and issuing alerts for distractions. Fleet management companies can use this system to track and improve driver attentiveness, reducing the likelihood of accidents and optimizing operational safety. Insurance companies may leverage the system for assessing driver risk profiles and providing safe driving incentives. Additionally, traffic enforcement agencies could adopt Safe Drive to monitor and ensure compliance with road safety regulations. Ultimately, the system benefits drivers, passengers, and public road users by promoting safer driving practices and reducing distraction-related accidents.

CHAPTER – 2

SOFTWARE REQUIREMENT SPECIFICATIONS

The Software Requirements Specifications (SRS) outline the essential software components and functionalities needed to develop an effective machine learning-based distracted driving monitoring system. This section details the tools, frameworks, and libraries required for image processing, machine learning, real-time monitoring, and alert generation to identify and mitigate driver distractions in real time.

2.1 Software Requirements

- **Web Development:**
  - Web Framework: Flask
  - Frontend Development: HTML, CSS, JavaScript
- **Image Processing and Machine Learning:**
  - Image Processing Library: OpenCV
  - Machine Learning Framework: TensorFlow, to implement the deep learning model for distraction detection.
  - Machine Learning Model : VGG - 16
- **Alert and Notification System:**
  - Audio Library: Pyttsx3 , for generating voice alerts to warn the driver.

2.1.1 Functional Requirements:

- **Detecting Driver Distractions:** Identify distractions like texting, eating, etc., using image data with technologies such as ML (TensorFlow, PyTorch), Computer Vision (OpenCV), and Deep Learning (CNNs).
- **Real-time Image Processing:** Process in-vehicle camera images to detect distractions in real time using technologies like OpenCV.
- **Trigger Voice Alerts:** Provide instant voice alerts when a distraction is detected using TTS (Google TTS), Audio Output Systems.
- **Monitor Driver Behavior:** Continuously observe driver actions during vehicle operation using Video Feed Analysis (OpenCV), and ML Algorithms.
- **Integrate with ADAS:** Integrate with Advanced Driver Assistance Systems for safety enhancement

2.1.2 Non-Functional Requirements

- **Performance:** The system should process and analyze images with minimal latency, ensuring real-time detection of driver distractions.
- **Accuracy:** The distraction detection model must be highly accurate to minimize false positives or negatives.
- **Scalability:** The system should be scalable to work with different vehicle models and setups.
- **User Interface:** Alerts and notifications should be clear, intuitive, and non intrusive to the driver.

2.2 Hardware Requirements

- CPU- 11th Gen Intel(R) Core (TM) i5-1135G7 @ 2.40GHz 2.42 GHz
- RAM – 16 GB
- High-speed internet

2.3 System Architecture

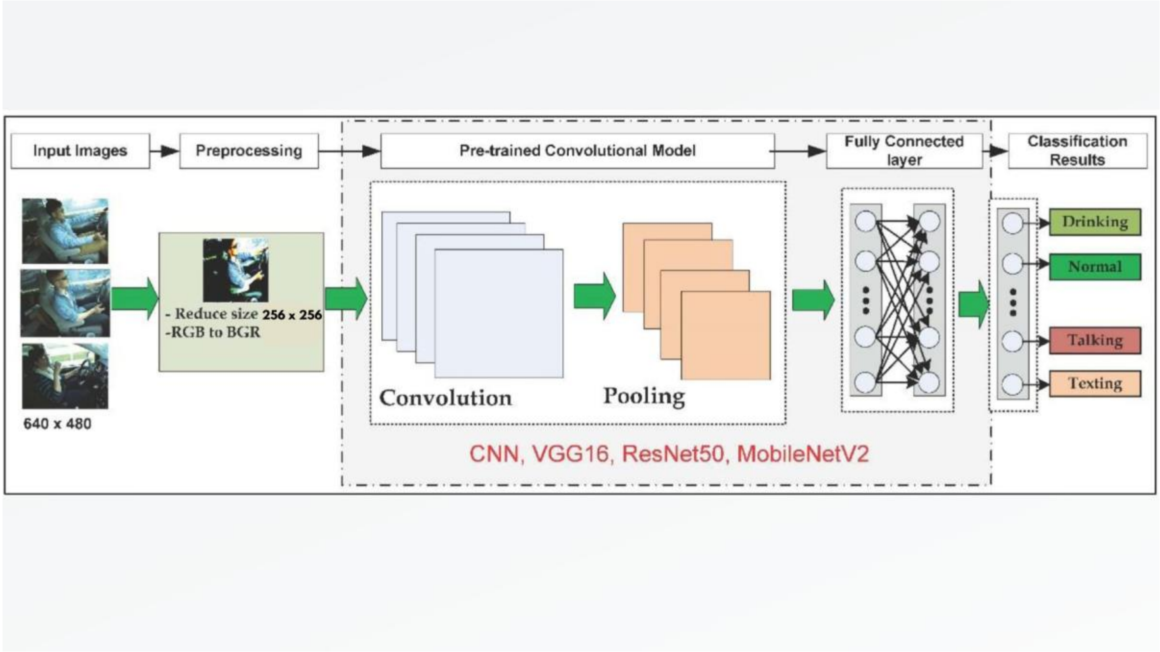


Fig 2.1: System Architecture

CHAPTER – 3

SYSTEM DESIGN

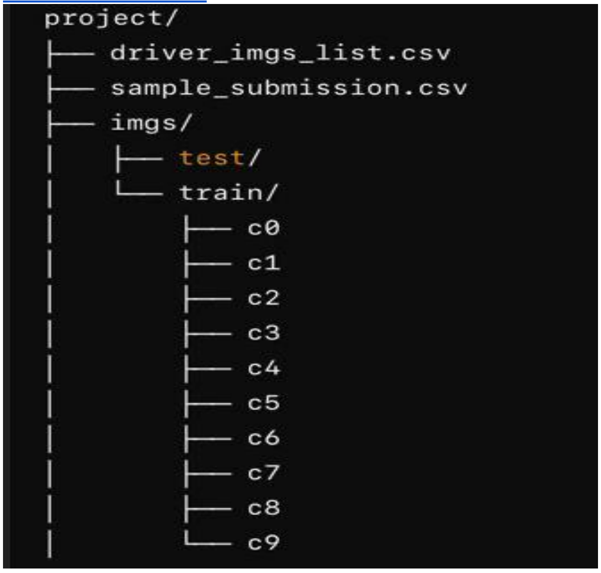
This chapter provides an in-depth look at the system design, including the dataset structure, algorithm implementation, and detailed diagrams that depict the system's architecture and behavior for efficient driver distraction detection.

3.1 Methodology

3.1.1 Dataset Description

The dataset consists of images of drivers exhibiting various behaviors, classified into different categories of distractions. This includes activities such as texting, talking on the phone, drinking, and other common in-car distractions. The images are labeled to indicate the specific type of activity each driver is engaged in, facilitating effective supervised learning for distraction classification. To ensure robust model training, the dataset was augmented, increasing its size and diversity to improve the model's ability to generalize to real-world scenarios.

Source:
https://www.kaggle.com/competitions/state-farm-distracted-driver-detection/data



```
project/
├── driver_imgs_list.csv
├── sample_submission.csv
├── imgs/
│   ├── test/
│   └── train/
│         ├── c0
│         ├── c1
│         ├── c2
│         ├── c3
│         ├── c4
│         ├── c5
│         ├── c6
│         ├── c7
│         ├── c8
│         └── c9
```

Fig 3.1: Structure of Dataset

3.1.2 Algorithms Description

For distraction detection, we utilized **VGG-16**, a popular convolutional neural network (CNN) model known for its deep architecture and effective feature extraction capabilities. VGG-16 was introduced by Simonyan and Zisserman in 2014 and is widely used for various image classification tasks. The VGG-16 architecture consists of 16 weight layers, including 13 convolutional layers and 3 fully connected layers, arranged in a simple yet deep design that uses small 3x3 convolution filters. These filters help capture fine-grained features in images, allowing the model to learn intricate details that are crucial for differentiating between similar driver behaviors. Each convolutional layer is followed by a Rectified Linear Unit (ReLU) activation, which introduces non-linearity into the network, enabling it to learn complex patterns. Max-pooling layers are used after every few convolutional layers to reduce spatial dimensions, improving computational efficiency while preserving important features. The final fully connected layers in VGG-16 perform the classification by mapping extracted features into the appropriate classes, representing different distraction types in this project. The model was fine-tuned to suit the specific needs of distraction detection, with the last few layers retrained to recognize driver behaviors.

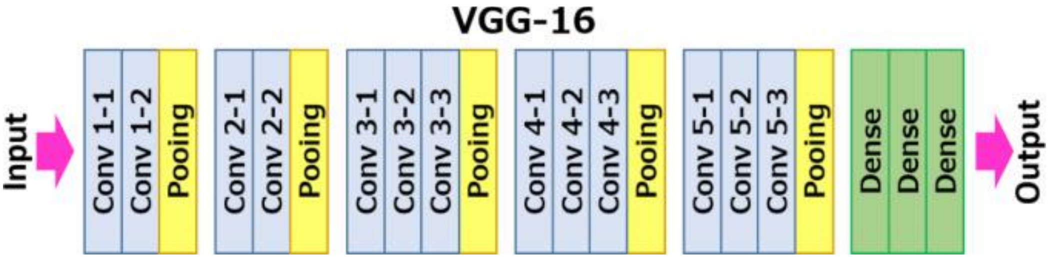


Fig 3.2: Architecture of VGG-16

3.2 Structural Diagram

- Class Diagram

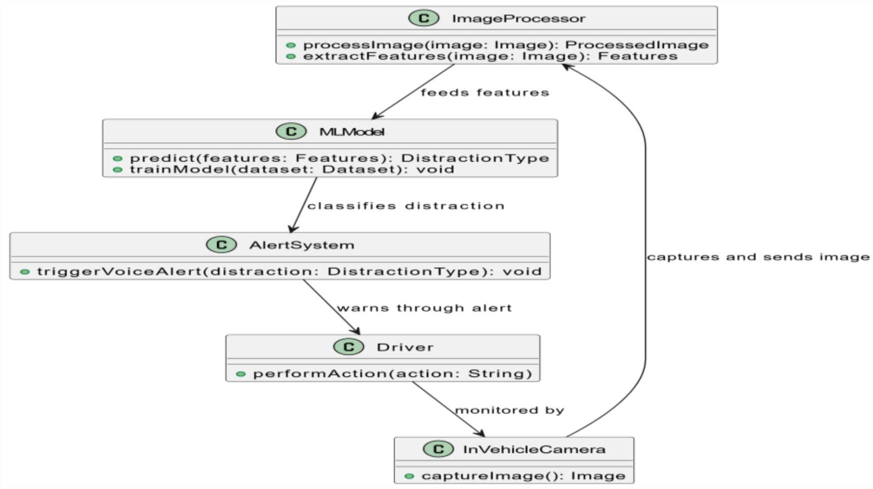

C ImageProcessor
processImage(image: Image): ProcessedImage
extractFeatures(image: Image): Features
feeds features
C MLModel
predict(features: Features): DistractionType
trainModel(dataset: Dataset): void
classifies distraction
C AlertSystem
triggerVoiceAlert(distraction: DistractionType): void
warns through alert
C Driver
performAction(action: String)
monitored by
C InVehicleCamera
captureImage(): Image
captures and sends image

Fig 3.3: Class Diagram

- State Diagram

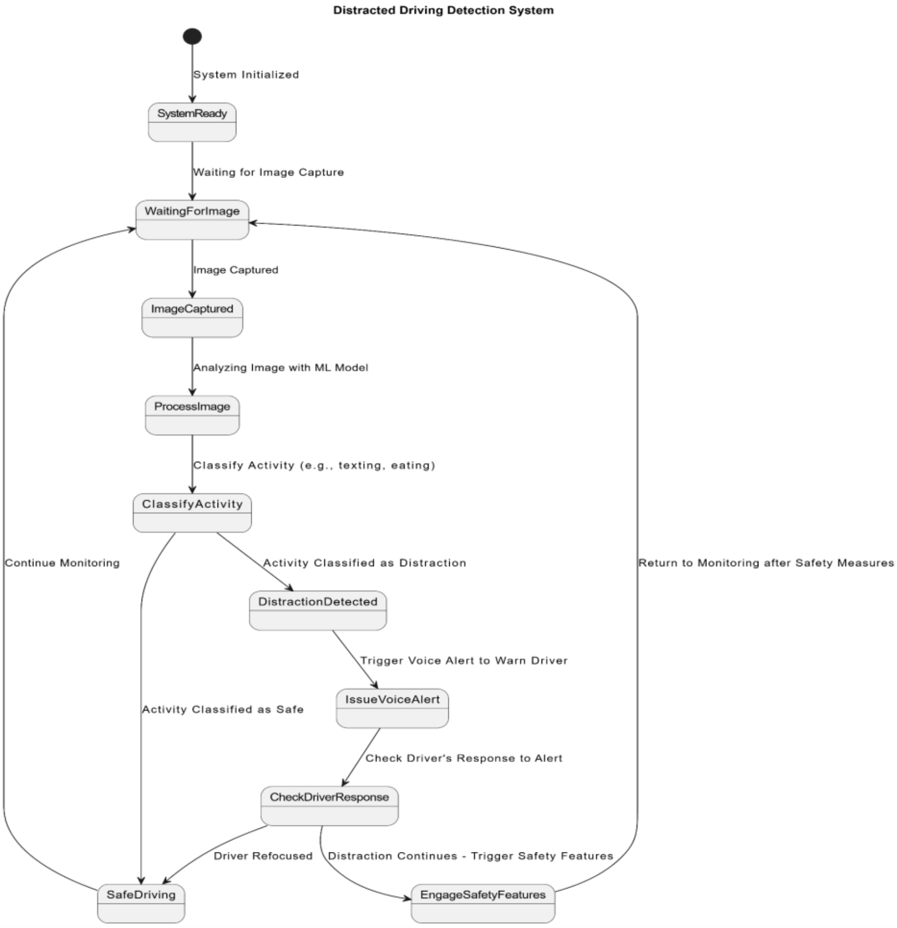

Distracted Driving Detection System
System Initialized
SystemReady
Waiting for Image Capture
WaitingForImage
Image Captured
ImageCaptured
Analyzing Image with ML Model
ProcessImage
Classify Activity (e.g., texting, eating)
ClassifyActivity
Continue Monitoring
Activity Classified as Distraction
Return to Monitoring after Safety Measures
DistractionDetected
Trigger Voice Alert to Warn Driver
Activity Classified as Safe
IssueVoiceAlert
Check Driver's Response to Alert
CheckDriverResponse
Driver Refocused
Distraction Continues - Trigger Safety Features
SafeDriving
EngageSafetyFeatures

Fig 3.4: State Diagram

3.3 Behavioral Diagram

- Use case Diagram

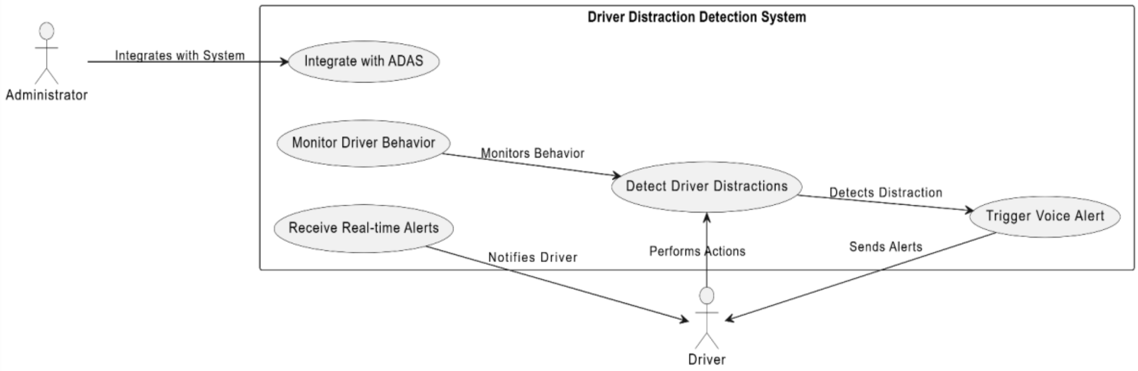


Fig 3.5: Usecase Diagram

- Sequence Diagram

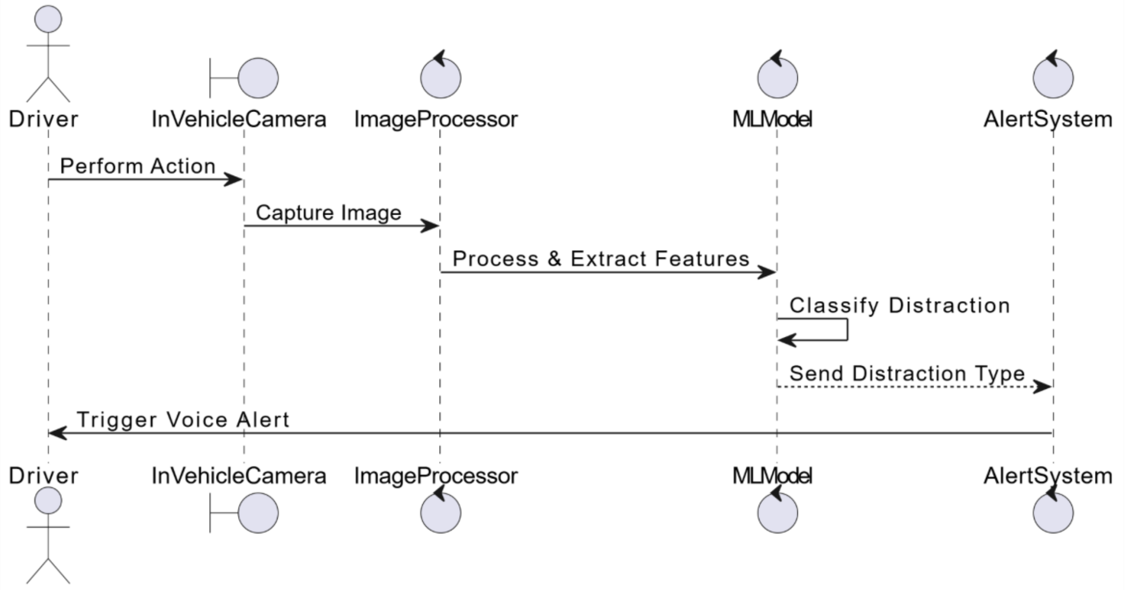


Fig 3.6: Sequence diagram

CHAPTER – 4

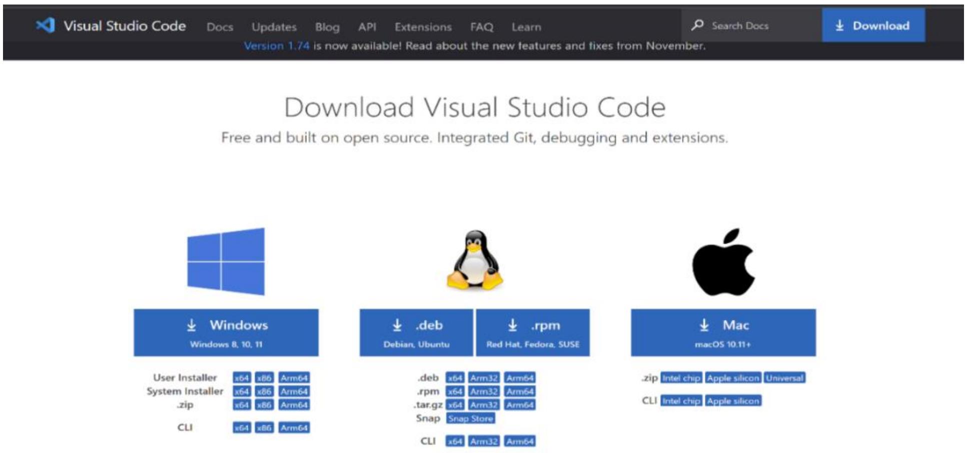
RESULTS & DISCUSSION

4.1 Environmental Setup

Installing an editor: Visual Studio (VS) Code

1. Download the VS Code by below mentioned link based on the operating system:

   https://code.visualstudio.com/download



Version 1.74 is now available! Read about the new features and fixes from November.

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

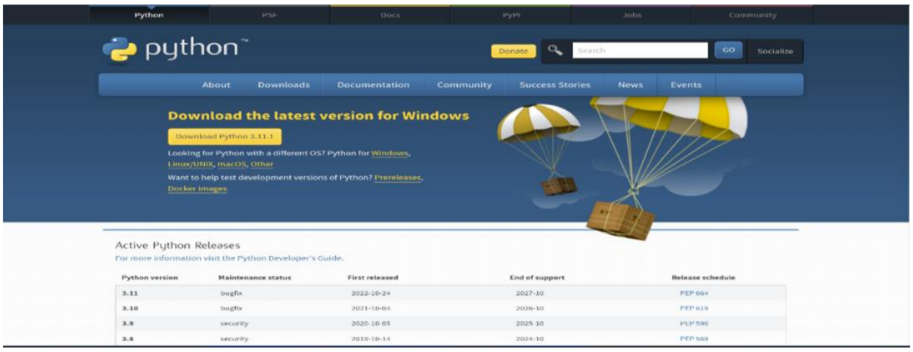
Fig 4.1: Visual Studio Code Installation

2. The VS Code executable file is downloaded. Double-click and start the installation by selecting the installation path and agreeing to the terms of installation.
3. After successful completion of VS Code installation, Install Python and select the installed Python as an interpreter in VS Code.

Installing Python:

1. Install Python from the link below by selecting the desired version.

   https://www.python.org/downloads/



Download the latest version for Windows

Looking for Python with a different OS? Python for Windows, Linux/UNIX, macOS, Other

Want to help test development versions of Python? Prereleases, Docker images

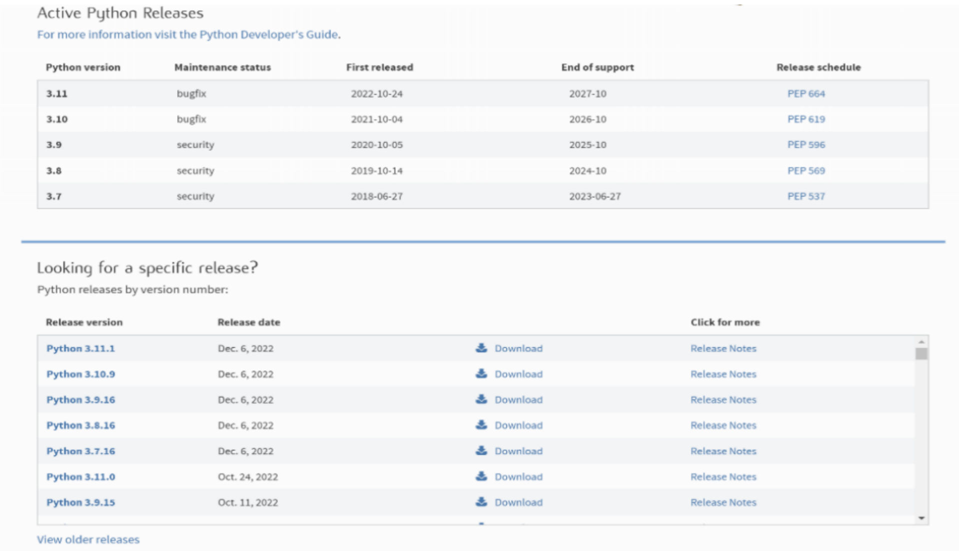
Active Python Releases

For more information visit the Python Developer's Guide.

Python version	Maintenance status	First released	End of support	Release schedule
3.11	bugfix	2022-10-24	2027-10	PEP 664
3.10	bugfix	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569

Fig 4.2: Python Installation

2. We chose to install Python 3.10.9. So, scroll down the installation page and look for a specific release of Python.



Active Python Releases

For more information visit the Python Developer's Guide.

Python version	Maintenance status	First released	End of support	Release schedule
3.11	bugfix	2022-10-24	2027-10	PEP 664
3.10	bugfix	2021-10-04	2026-10	PEP 619
3.9	security	2020-10-05	2025-10	PEP 596
3.8	security	2019-10-14	2024-10	PEP 569
3.7	security	2018-06-27	2023-06-27	PEP 537

Looking for a specific release?

Python releases by version number:

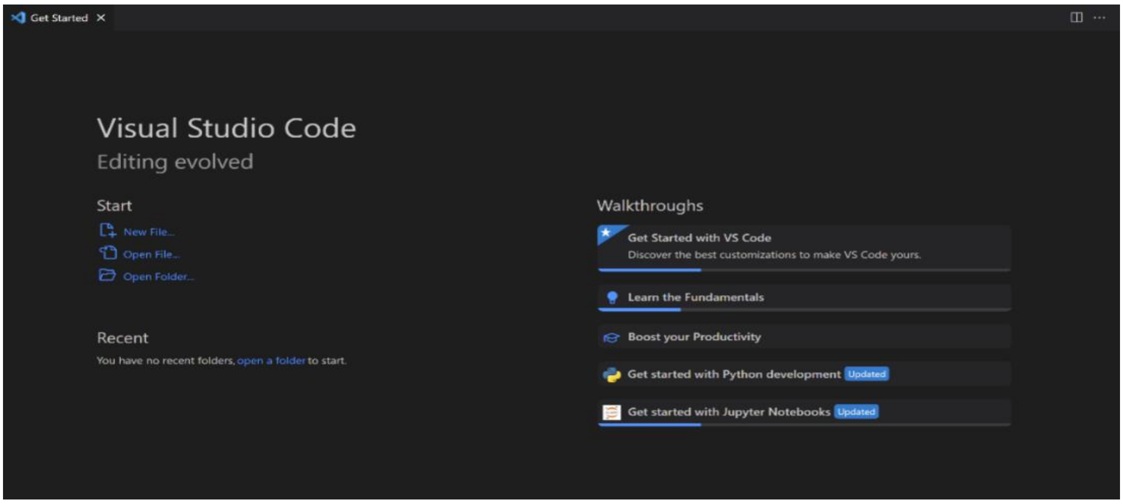
Release version	Release date		Click for more
Python 3.11.1	Dec. 6, 2022	Download	Release Notes
Python 3.10.9	Dec. 6, 2022	Download	Release Notes
Python 3.9.16	Dec. 6, 2022	Download	Release Notes
Python 3.8.16	Dec. 6, 2022	Download	Release Notes
Python 3.7.16	Dec. 6, 2022	Download	Release Notes
Python 3.11.0	Oct. 24, 2022	Download	Release Notes
Python 3.9.15	Oct. 11, 2022	Download	Release Notes

View older releases

Fig 4.3: Python-Specific Release Installation

3. Click on the link below for specific Python 3.10.9 release installation. https://www.python.org/downloads/release/python-3109/
4. Look for the installer file and download it. Once the download is complete, run the exe to install.
5. On the next screen, Change the installation path if required. Click “Next”.
6. On the next screen, you can create a desktop shortcut If you want and click on “Next”.
7. Once the installation is finished, you should receive a message screen that Python is installed.

After Python installation, Open VSCode and select the interpreter path as the Python installation path.



Visual Studio Code

Editing evolved

Start

Recent

You have no recent folders, open a folder to start.

Walkthroughs

Get Started with VS Code

Discover the best customizations to make VS Code yours.

Learn the Fundamentals

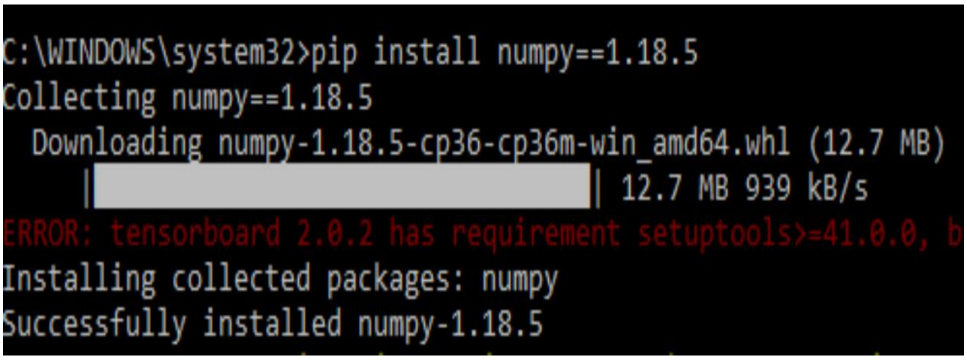
Boost your Productivity

Get started with Python development

Get started with Jupyter Notebooks

Fig 4.4: Visual Studio Welcome page

- Open the new project folder and start writing the source code.
- To execute the source code, a few modules need to be installed. To install those modules, select the ‘terminal’ in the VSCode window and select the new terminal.
- Terminal will be opened with the path of source code. Type “python -m pip install package name” which you want to install (like NumPy, pandas, matplotlib,tensorflow, sklearn, flask, open-cv , Pyttsx3, ultralytics).



```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
     |████████████████████████████████| 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, b
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

Fig 4.5: Modules Installation

Modules Installation

- **Scikit-learn** is a toolbox for machine learning in Python. It offers a wide range of tools for tasks like teaching machines to make predictions or recognize patterns in data. It's popular because it's easy to use and comes with a bunch of useful features that make it great for different kinds of machine learning problems.

```
pip install scikit-learn
```

- **TensorFlow** is an open-source deep learning framework developed by Google. It provides a comprehensive ecosystem for building and deploying machine learning models. In this project, TensorFlow is used to create, train, and evaluate deep learning models.

```
pip install tensorflow
```

- **OpenCV** (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. It is used for image processing tasks such as reading images, image transformation, and manipulation.

```
pip install opencv-python
```

- **Pyttsx3** is a text-to-speech conversion library in Python. It enables the Safe Drive system to generate voice alerts to warn drivers when distractions are detected.

```
pip install pyttsx3
```

- **Ultralytics** offers tools for computer vision tasks, including object detection and segmentation models. In Safe Drive, it supports specific functionality for model training or performance tracking.

```
pip install ultralytics
```

Setup Of Flask

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask.

```
python –m pip install flask
```

The **Requests** module allows you to send HTTP requests using Python. Requests are used for making GET and POST requests. It abstracts the complexities of making requests behind a beautiful, simple API.

Running Server

Start the server and it redirects to localhost

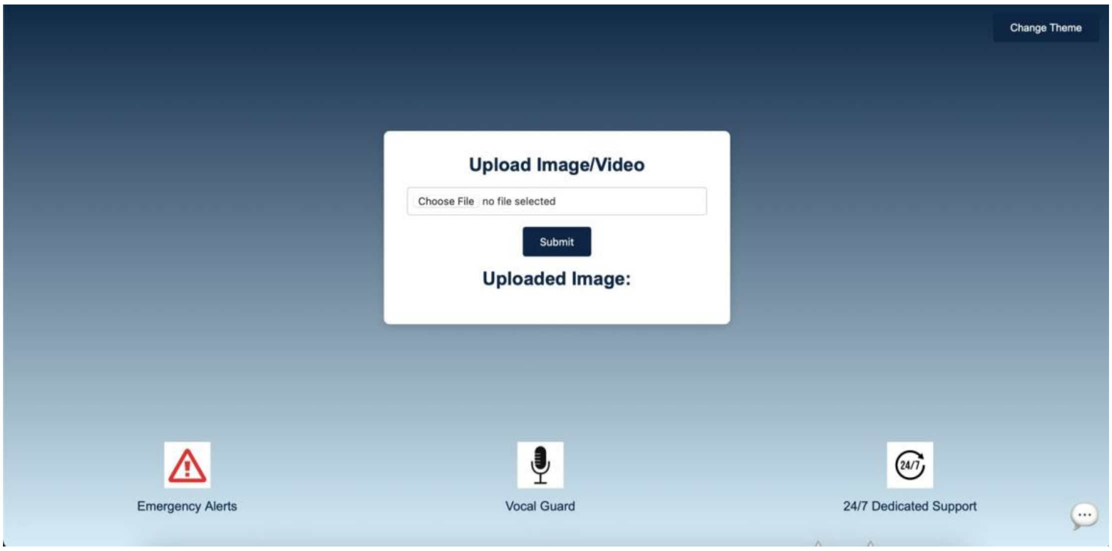


Fig 4.6: Page redirection after running the server

4.2 Implementation Details

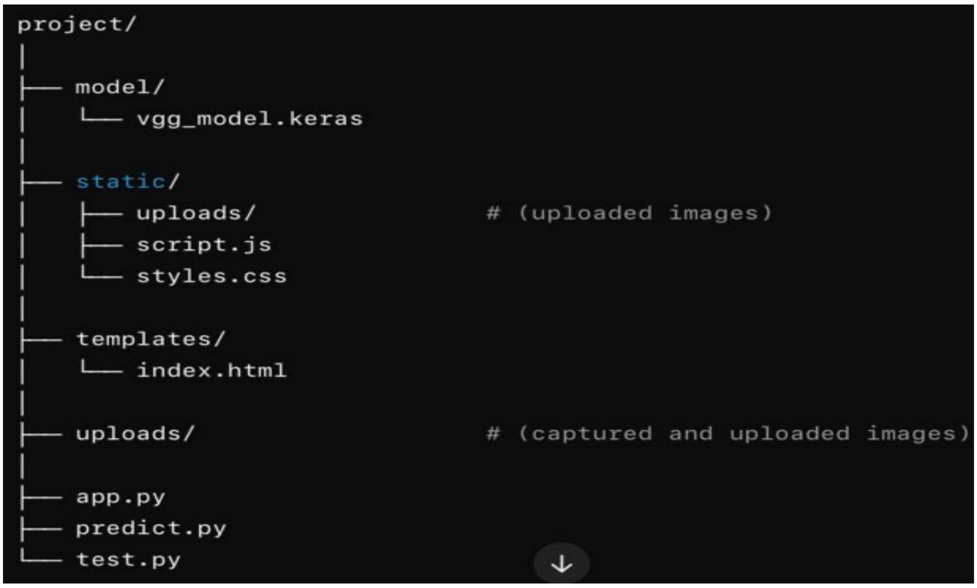
Follow the steps below to implement the project:

1. **Setup and Prerequisites**

- **Clone the Repository**: Ensure that you have cloned your project repository to your local machine.
- **Install Docker**: Ensure Docker is installed on your system. You can download it from here.
- **Install Python**: Make sure you have Python installed. You can download it from here.

2. **Directory Structure**

Your project directory structure should be as follows:



```
project/
|
├── model/
|   └── vgg_model.keras
|
├── static/
|   ├── uploads/              # (uploaded images)
|   ├── script.js
|   └── styles.css
|
├── templates/
|   └── index.html
|
├── uploads/                  # (captured and uploaded images)
|
├── app.py
├── predict.py
└── test.py
```

Fig 4.7: Project Directory Structure

- **model/vgg_model.keras**: Trained VGG model file for detecting distracted driving from uploaded images.
- **static/**
  I. **uploads/**: Directory for storing user-uploaded images.
  II. **script.js**: JavaScript for client-side interactivity (e.g., handling image uploads and user actions).
  III. **styles.css**: CSS for styling the web interface.
- **templates/index.html**: Main HTML file rendered by Flask, providing the user interface for uploading images and displaying predictions.
- **uploads/ (various images)**: Directory for storing captured and uploaded images.
- **app.py** - Flask application file that handles routes, image uploads, and invokes prediction functions.
- **predict.py** - Script for loading the trained model(vgg_model.keras) and performing predictions on uploaded images.

3. **Create the Flask App**

   app.py: This file initializes the Flask application and sets up the routes for the web interface.

4. **Model Prediction Script**

   predict.py: This script loads the trained model and performs predictions on the uploaded images.

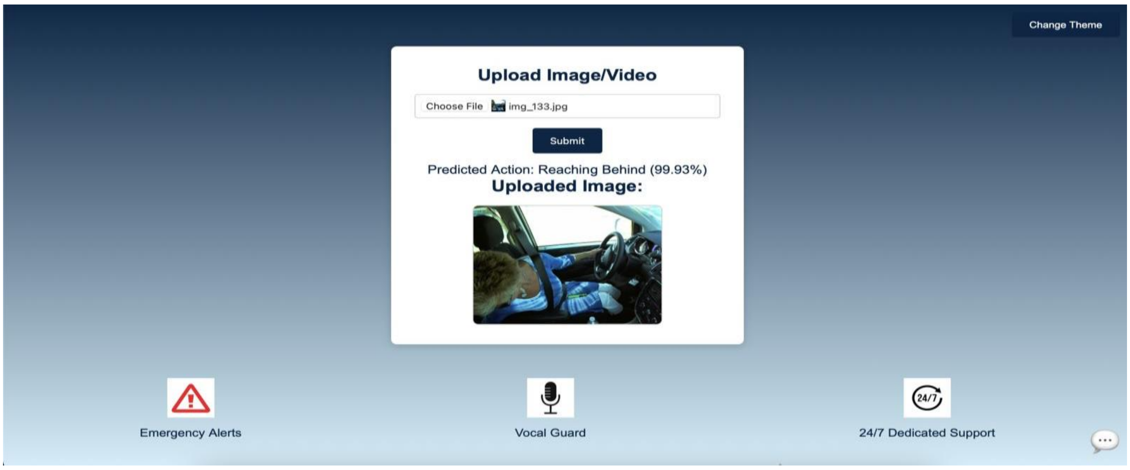
5. **Frontend Development**
   - **index.html**: The main HTML file for the web interface.
   - **styles.css**: Add custom styles for your web page.
   - **script.js**: Add JavaScript functionalities if needed (e.g., handling image uploads and user actions).

6. **Access the Application**

   Open your web browser and navigate to http://localhost:5000. To see lung disease detection web application.

4.3 Results

Our web-based distracted driving detection system uses machine learning to analyze images of drivers, identifying signs of distraction such as phone use or lack of focus on the road. By providing real-time alerts, it helps improve road safety, reduce accidents, and promote responsible driving, ultimately contributing to safer roads and better outcomes for all.


Upload Image/Video

Predicted Action: Reaching Behind (99.93%)

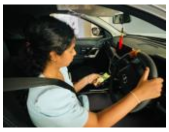
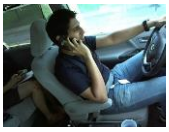
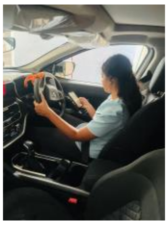
Uploaded Image:

Fig 4.8: Resulted webpage

4.4 Test Cases

Test Case ID	Test Case Objective	Prerequisite	Steps	Input Data	Expected Output	Actual Output	Status
TC_01	Test Safe Driving	The model should predict whether the driver is driving safe .	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.		Safe Driver	Safe Drive	PASS
TC_02	Test Reaching Behind	The model should predict whether the	● Click the "upload image/video" button on the user interface.		Reaching Behind	Reachin g Behind	PASS

		driver is reaching behind.	● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.				
TC_03	Test Talking to Passenger	The model should predict whether the driver is talking to passenger.	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.		Talking to passenger	Talking to passenger	PASS
TC_04	Test Talking on phone-left	The model should predict whether the driver is talking on phone-left.	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be		Talking on phone left	Talking on phone left	PASS

			displayed below.				
TC_05	Test texting-left	The model should predict whether the driver is texting-left.	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.		Texting left	Texting left	PASS
TC_06	Test Talking on phone-Right	The model should predict whether the driver is Talking on the phone - right.	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.		Talking on phone-right	Talking on the phone right.	PASS
TC_07	Test Texting-Right	The model should predict whether the driver is texting right.	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in		Texting right	Texting right	PASS

			camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.				
TC_08	Test Drinking	The model should predict whether the driver is drinking.	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.		Drinking	Drinking	PASS
TC_09	Test Hair And Makeup	The model should predict whether the driver is doing hair and makeup.	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.		Hair and makeup	Hair and makeup	PASS

TC_010	Test Operating the Radio	The model should predict whether the driver is reaching behind.	● Click the "upload image/video" button on the user interface. ● Upload an image or video captured in camera that is fixed in a car ● Click the "predict" button. ● The predicted action in the image/video will be displayed below.		Operating the radio	Operating the radio	PASS

Table 4.1: Test Cases

CHAPTER – 5

CONCLUSION & FUTURE ENHANCEMENTS

Conclusion

The Safe Drive ML-Based Distracted Monitoring System offers a promising solution to the growing problem of distracted driving. By utilizing advanced machine learning techniques, specifically the VGG-16 CNN, the system can accurately detect and classify a wide range of driver distractions in real time, such as texting, eating, and phone use. This proactive approach, which triggers voice alerts to refocus the driver, has the potential to significantly reduce distraction-related accidents. The system's integration with existing vehicle safety features further enhances its effectiveness, creating an additional layer of protection. With broad applicability across automotive manufacturers, fleet management, and insurance sectors, Safe Drive aims to improve road safety, promote safer driving behaviors, and ultimately save lives.

Future Enhancements

- Extend detection capabilities to include distractions like drowsiness and emotional distress using facial recognition and physiological sensors.
- Implement advanced AI models for higher accuracy and faster real-time prediction in challenging driving scenarios.
- Enable vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication for coordinated safety alerts and responses.
- Integrate adaptive learning for personalized alerts tailored to individual driver behavior patterns.
- Enhance scalability for widespread adoption across diverse vehicle types and regions, meeting regulatory standards.

Source / Pseudo code :

main.py

```
Import libraries for data processing, model building, image handling, and evaluation
Load 'driver_imgs_list.csv'
Initialize a dictionary to store images for each class
For each image in the CSV file:
    Add image file name to the corresponding class list
Define main dataset path
Check if 'smallest' dataset folder exists; if yes, delete it
Create new 'smallest' dataset folder with subfolders for each class
For each class:
    Shuffle the list of images
    Copy a subset (70%) of images to the 'smallest' dataset
Initialize paths for train, validation, and test directories
Create subfolders for each class in these directories
For each class:
    Split images into train (80%), validation (10%), and test (10%)
    Copy images to the respective folders
Initialize image data generators with data augmentation (rotation, zoom, etc.)
Return data generators for train, validation, and test sets
Set up early stopping callback to monitor validation accuracy
Stop training if no improvement for 5 consecutive epochs
1. Dense Model:
    - Define a sequential model with dense layers and dropout
    - Compile the model (Adam optimizer, categorical crossentropy loss)
    - Train the model using train and validation data generators
    - Evaluate the model on the test set
    - Save the model
2. CNN Model:
    - Define a convolutional model with Conv2D, MaxPooling, and Dense layers
    - Compile the model
    - Train and evaluate the model
    - Save the model
3. VGG16 Model (Transfer Learning):
    - Load VGG16 model without top layers
    - Freeze pre-trained layers
    - Add custom dense layers on top
    - Compile and train the model
    - Evaluate and save the model
4. ResNet50 Model (Optional):
    - Load ResNet50 model without top layers
```

```
   - Freeze pre-trained layers
   - Add custom dense layers on top
   - Compile and train the model
   - Evaluate and save the model
Define function to plot training and validation accuracy and loss for each model
Display the plots
Print test accuracy for Dense, CNN, VGG16, and ResNet50 models
```

predict.py

```
IMPORT TensorFlow AS tf
IMPORT NumPy AS np
IMPORT OpenCV AS cv2
IMPORT pyttsx3
DEFINE class_labels = {
   0: 'Safe Driving',
   1: 'Texting - Right',
   2: 'Talking on the Phone - Right',
   3: 'Texting - Left',
   4: 'Talking on the Phone - Left',
   5: 'Operating the Radio',
   6: 'Drinking',
   7: 'Reaching Behind',
   8: 'Hair and Makeup',
   9: 'Talking to Passenger'
}
SET model TO LOAD 'vgg_model.keras'
SET engine TO INITIALIZE pyttsx3
SET engine property 'rate' TO 150
DEFINE alert_spoken = {1: False, 2: False, 3: False, 4: False, 5: False,
                  6: False, 7: False, 8: False, 9: False}
FUNCTION speak_message(message)
   CALL engine.say(message)
   CALL engine.runAndWait()
END FUNCTION
FUNCTION preprocess_image(frame)
   SET IMG_SIZE TO (256, 256)
   RESIZE frame TO IMG_SIZE
   CONVERT frame FROM BGR TO RGB
   NORMALIZE frame (scale pixel values to [0, 1])
   EXPAND frame dimensions TO (1, 256, 256, 3)
   RETURN preprocessed frame
END FUNCTION
FUNCTION predict_action(frame)
```

```
    CALL preprocess_image(frame) AND STORE IN processed_image
    PREDICT USING model ON processed_image AND STORE RESULT IN predictions
    FIND index OF MAX probability IN predictions AS predicted_class_index
    MAP predicted_class_index TO predicted_action USING class_labels
    RETURN predicted_action, predicted_class_index
END FUNCTION
FUNCTION process_video(video_path)
    OPEN video file AT video_path
    IF video NOT opened
        PRINT "Error: Could not open video."
        EXIT FUNCTION
    WHILE True
        READ frame FROM video
        IF frame NOT read
            PRINT "End of video or error in reading frame."
            BREAK
CALL predict_action(frame) AND STORE RESULT IN predicted_action, predicted_class_index
        DISPLAY frame WITH predicted_action label
IF predicted_class_index IN alert_spoken AND alert_spoken[predicted_class_index] IS False
            SET alert_message_en TO "Concentrate on driving rather than on predicted_action."
            SET alert_message_te TO "Telugu alert message for predicted_action."
            CALL speak_message(alert_message_en)
            CALL speak_message(alert_message_te)
            SET alert_spoken[predicted_class_index] TO True
        IF predicted_class_index IS 0
            RESET alert_spoken dictionary (all values TO False)
        IF 'q' key pressed
            BREAK
    RELEASE video capture AND CLOSE all OpenCV windows
END FUNCTION
FUNCTION process_image(image_path)
    READ image FROM image_path
    IF image NOT loaded
        PRINT "Error: Image not loaded properly!"
        RETURN
    CALL predict_action(image) AND STORE RESULT IN predicted_action
    PRINT "Predicted Action: predicted_action"
END FUNCTION
```

```
FUNCTION main()
    PROMPT user TO select input type ('image' OR 'video')
    IF user selects 'image'
        PROMPT user FOR image_path
        CALL process_image(image_path)
    ELSE IF user selects 'video'
        PROMPT user FOR video_path
        CALL process_video(video_path)
    ELSE
        PRINT "Invalid choice. Please select either 'image' or 'video'."
END FUNCTION
CALL main()
```