A Project report on

RIDER HELMET DETECTION FOR FUEL FILLING

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology

in

Computer Science and Engineering

Submitted by

C SRIKANTH
(20H55A0501)
D SATHISH
(20H55A0505)
J POOJA
(20H55A0512)

Under the esteemed guidance of
B GAYATHRI
(Assistant Professor)

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING &TECHNOLOGY

(An Autonomous Institution under UGC & JNTUH, Approved by AICTE, Permanently Affiliated to JNTUH, Accredited by NBA.)
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2019- 2023

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Major Project Phase-II report entitled **"RIDER HELMET DETECTION FOR FUEL FILLING"** being submitted by **C. SRIKANTH *(20H55A0501)*, D. SATHISH (20H55A0505), J. POOJA *(20H55A0512)*** in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embody in this project report have not been submitted to any other University or Institute for the award of any Degree.

GUIDE	**HEAD OF THE DEPT.**
B GAYATHRI	**Dr. Siva Skandha Sanagala**
Assistant Professor	**Associate Professor and HOD**
Dept. of CSE	**Dept. of CSE**

ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express my heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **B GAYATHRI,** Assistant Professor, Department of Computer Science and Engineering for his valuable technical suggestions and guidance during the execution of this project work.

We would like to thank **Dr. Siva Skandha Sanagala,** Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete my project work successfully.

We are very grateful to **Dr. Vijaya Kumar Koppula**, Dean-Academic, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Dr. V A Narayana,** Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the Teaching & Non- teaching staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Mr. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care.

Finally, we extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

C SRIKANTH	20H55A0501
D SATHISH	20H55A0505
J POOJA	20H55A0512

TABLE OF CONTENTS

List of Figures

ABSTRACT

Motorcycle accidents have been rapidly growing through the years in many countries. In India more than 37 million people use two wheelers. Therefore, it is necessary to develop a system for automatic detection of helmet wearing for road safety. Therefore, a custom object detection model is created using a Machine learning based algorithm which can detect motorcycle riders with helmet. By this system it detects the rider and if the rider with helmet, then it opens the gate and allows the biker for fuel filling. By this system we can avoid the motorcycle accidents and if we keep helmet as a mandatory for rider while filling fuel at petroleum bunks every biker will maintain and wearied with helmet using this caution, we can save the lives and they start following the traffic rules and road safety rules. This Application can be implemented in real-time using a Webcam or a CCTV as input.

CHAPTER 1
INTRODUCTION

CHAPTER 1

INTRODUCTION

The two-wheeler is a common means of transportation in practically every country. However, due to the lack of safeguards, there is a significant danger associated. The wearing of a helmet by bike riders can significantly lessen the risks they face on the road. Because of the importance of wearing a helmet, governments have made it a crime to ride a bike without helmet. Now accidents on roads have become a serious concern for all the thousands of people especially youngsters. Transport Ministry Road accident report 2021 which stresses the need to tackle this issue as soon as possible seventy-five thousand youngsters died in road accidents last year.

Rider helmet detection for fuel filling is a project aimed at ensuring the safety of riders at gas stations. The idea is to use computer vision technology to detect whether a rider is wearing a helmet or not, before allowing them to fuel their vehicle. This is important because wearing a helmet is a critical safety measure for riders, and accidents can happen even at gas stations. The project involves installing a camera or a system of cameras at the fuel pump, which can detect the presence of a helmet on the rider's head. The system can then prevent fuel from being dispensed until the rider puts on a helmet, thereby promoting safety for all.

Rider helmet detection for fuel filling project is a system designed to promote safety for motorcyclists by ensuring that they wear helmets when refueling their motorcycles at fuel stations. This system employs computer vision technology to detect whether or not a rider is wearing a helmet and provides an alert to the fuel station attendant if a rider is not wearing a helmet.

The primary objective of this project is to prevent accidents and promote the use of helmets by motorcyclists. According to studies, helmets are the most effective way to reduce the risk of head injuries and fatalities in motorcycle accidents. However, many riders do not wear helmets, especially when they are refueling their motorcycles at fuel stations.

To address this issue, the rider helmet detection system for fuel filling project utilizes image processing algorithms to detect the presence or absence of a helmet on a rider's head. The system uses a camera and a computer vision model to analyze the video feed and determine if a rider is wearing a helmet or not. If the system detects that a rider is not wearing a helmet, an alert is sent to the fuel station attendant who can then take appropriate action.

Overall, the rider helmet detection for fuel filling project is an innovative solution that enhances road safety by promoting the use of helmets among motorcyclists. By detecting and alerting riders who are not wearing helmets, this system helps to reduce the risk of accidents and protect the lives of riders.

1.1 PROBLEM STATEMENT

Our main objective is to develop a system which detects the motorcycle rider with helmet and allows him to fuel filling at pumps of petroleum bunk. Here we allowing the rider for fuelling after detection of helmet, if helmet detected successfully then the system process the instructions to the Arduino then the gates raised and system allows the biker for fueling.

The problem statement aims to find a solution to this issue by developing a rider helmet that can be easily and safely removed during fuel filling. The helmet should provide convenience to riders and comply with safety regulations, such as enabling gas station attendants to properly identify the rider and ensure their safety. The solution may involve incorporating new technology or materials into the helmet or redesigning the helmet to make it easier to remove without compromising safety.

1.2 RESEARCH OBJECTIVE

We have gone through a news and some social media where we saw the people who are teenagers and youth were mostly meet with an accident by driving the motorcycles without wearing the Helmet**.**In India more than 37 million people use two wheelers. Therefore, it is necessary to develop a system for automatic detection of helmet wearing for road safety.

The research objective for rider helmet for fuel filling could be to investigate and develop a helmet design that incorporates a mechanism for safe and easy fuel filling for motorcycles. This would involve studying the current methods of fuel filling while wearing a helmet, identifying the challenges and risks involved, and designing a solution that addresses these issues. The objective would be to create a helmet that allows riders to easily and safely refill their motorcycle tanks without having to remove their helmets, thereby improving convenience and safety for riders. The research could also involve testing the new helmet design for its effectiveness and durability, as well as gathering feedback from riders to ensure that the new design meets their needs and expectations.

1.3 PROJECT SCOPE AND LIMITATIONS

- SCOPE
  - The helmet detection ensures motorcycle rider safety and also takes the necessary steps to reduce accident causality.
  - We keep helmet as a mandatory for rider while filling fuel at petroleum bunks every biker will maintain and wearied with helmet using this caution.

- LIMITATIONS
  - We can’t sure that bike rider can wear helmet through-out the ride.
  - Our project checks only riders helmet detection.
  - We just implemented for only two wheeler helmet detection.

FEASABILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CHAPTER 2

BACKGROUND WORK

CHAPTER 2

2.1 NUMBER PLATE DETECTION WITHOUT HELMET

2.1.1 INTRODUCTION

The two-wheeler is a common means of transportation in practically every country. However, due to the lack of safeguards, there is a significant danger associated. The wearing of a helmet by bike riders can significantly lessen the risks they face on the road. Because of the importance of wearing a helmet, governments have made it a crime to ride a bike without one, and they've implemented enforcement measures including random checks and other manual methods to detect those who do. In contrast, the current image-based surveillance technologies are passive and require a lot of human intervention. Due to the fact that these systems rely on humans, their efficiency degrades over time. With automation, the monitoring of these violations will be more reliable and resilient, as well as lower the quantity of human resources required. In addition, a growing number of countries have implemented systems that use cameras to monitor people in public locations. As a result, existing infrastructure may be used to detect offenders at a low cost. However, there are a number of hurdles to overcome before such automated solutions may be implemented.

Implementation in real time It's difficult to deal with a large volume of information in a short amount of time. In order to accomplish real-time implementation, operations like segmentation, feature extraction, classification, and tracking necessitate rapid processing of large amounts of data.

The dynamic objects generally obstruct each other in real life circumstances; thus, the object of interest is only partially visible. When things are only partially visible, it becomes more difficult to classify and segment them.

Three-dimensional objects, by their very nature, appear to move in different directions when viewed from various perspectives. The accuracy of classifiers is generally known to depend on the features utilized, which in turn depends on the angle. Consider the front and side views of a bike rider as an illustration.

Environmental circumstances, such as illumination, shadows, and so on, undergo numerous variations over time. Background modeling, for example, can become more difficult as a result of small or large shifts in the environment.
Image Feed Quality: CCTV cameras often collect images with a low quality. This is made much more difficult by factors such as poor lighting and adverse weather conditions.

Stricter rules make activities like segmentation, classification and tracking harder. A good surveillance framework should be able to perform in real time, be finely tuned, be resilient to rapid changes, and be able to forecast what will happen next. These constraints and desired qualities prompted us to develop a system for detecting bike riders without helmets using existing security camera footage in real time.

2.1.2 MERITS, DEMERITS AND CHALLENGES

Merits:

- Detects the number plates if they don't ware helmet.
- Makes the work easy to detect and punish who is illegal.

Demerits:

- Hard to find multiple peoples
- Only checks helmet & number plate
- Doesn't store all the data
- Only checks for images

CHALLENGES

If a helmet is not present in an image, the application will identify the number plate; if a helmet is present, the application will not identify the number plate.

This is because we don't have enough images to train the CNN model, so our application can detect the presence of a helmet from 25 different images

The Number Plate Detection Project, which is aimed at automatically detecting and recognizing license plate numbers on vehicles, faces several challenges that need to be addressed for its successful implementation. Some of these challenges include:

1. Lighting and Weather Conditions: The success of license plate detection largely depends on lighting and weather conditions. Images captured in low-light conditions or during heavy rainfall or snowfall may be difficult to process accurately. It is therefore important to develop robust algorithms that can handle a wide range of lighting and weather conditions.
2. Vehicle Speed and Distance: License plate detection becomes more challenging when the vehicle is moving at high speeds or at a significant distance from the camera. The system must be able to accurately capture and process the license plate information in these situations.

3. License Plate Variations: License plates come in different sizes, shapes, and colors, and may be made of various materials. In addition, license plates may contain special characters, logos, or symbols that are specific to a particular region. The license plate detection system must be able to recognize these variations and accurately identify the license plate number.
4. Vehicle Angle and Orientation: License plate detection may also be challenging when the vehicle is at an angle or in a position that is difficult for the camera to capture. The system must be able to adjust for the vehicle angle and orientation to accurately capture the license plate number.
5. Privacy Concerns: License plate detection systems capture images of vehicles and their occupants, which may raise privacy concerns. It is therefore important to ensure that the system is compliant with relevant privacy laws and regulations and that appropriate safeguards are in place to protect the privacy of individuals.
6. Maintenance and Support: The success of the license plate detection project also depends on the availability and reliability of maintenance and support services. The system must be regularly maintained and updated to ensure that it remains accurate and effective over time.

Addressing these challenges will require a combination of technical expertise, robust algorithms, and effective project management strategies. With proper planning and execution, license plate detection can improve road safety and aid in law enforcement efforts.

2.1.3 IMPLEMENTATION OF NUMBER PLATE DETECTION

Thresholding, a segmentation technique, is used to isolate an object from its surrounding environment. The pixel intensity of each pixel is compared to a predetermined threshold in this process. After thresholding, adjacent pixels are joined to form a ternary pattern. A wide range of ternary values can be found in a histogram; therefore, the ternary pattern is divided into two binary ones. In order to construct a descriptor twice as large as LBP, histograms are joined together. The technology used to discover and identify items in an image or image sequence is called object recognition. The color and character elements of a license plate are used to extract it, followed by texture-based segmentation.

Low-level image processing techniques such as noise removal may be used first, followed by the extraction of features such as lines, regions, and maybe even areas with specific textures.

It then uses a method known as a feature descriptor to produce feature descriptors/feature vectors from a picture. The convolutional neural network (CNN) is then employed, which includes one or more convolutional layers and is mostly used for image processing, classification, segmentation, and other auto-correlated data processing. ' Using optical character recognition and thresholding/template matching, a license plate recognition system may then be implemented.

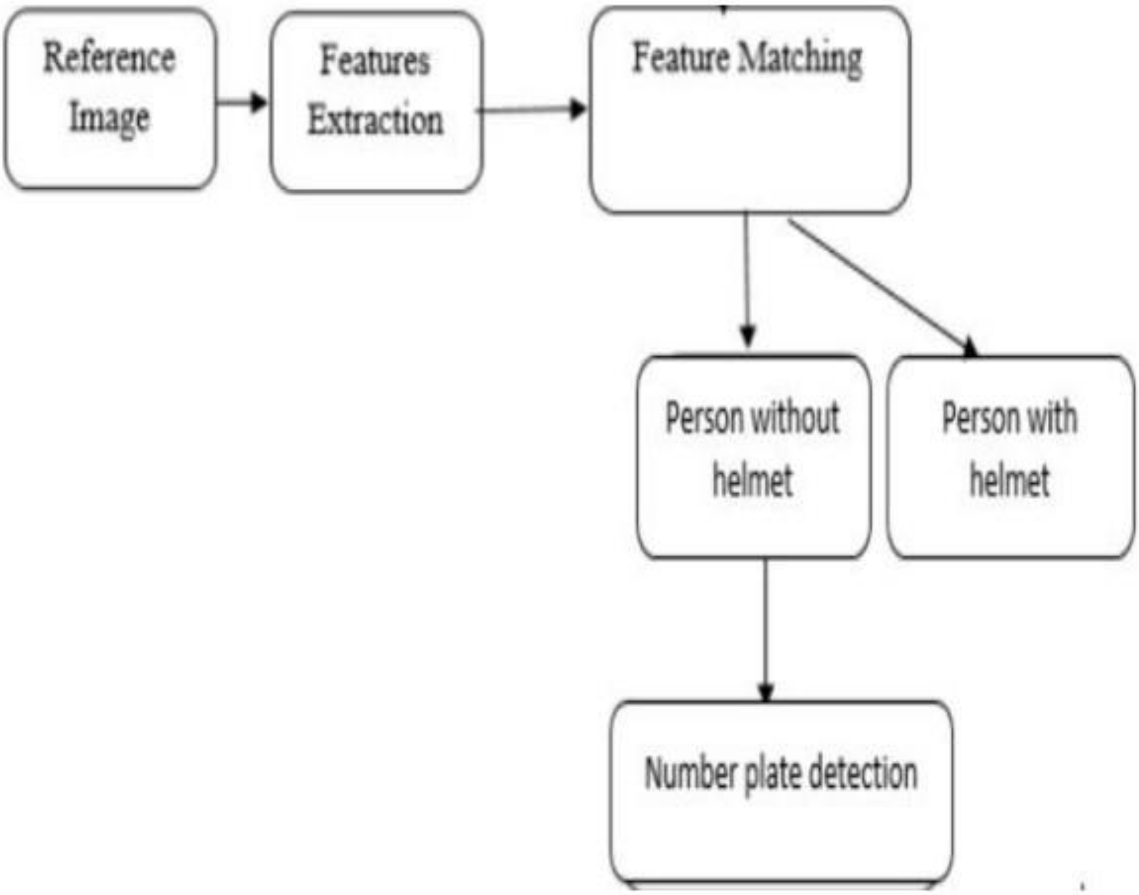


Fig 1: *System Architecture*

This is a system for detecting whether or not someone is wearing a helmet and reading their license plate based on photos obtained from a traffic monitoring system and fed into convolutional neural networks. If the person is not wearing a helmet, convolutional neural networks will be used to recognize the number plate characters of the bike rider and create an accurate and effective output in the next step.

This module involved in this project is

1. Upload image
2. Detect motor bike & person
3. Detect helmet

Using either an ipcam or a webcam, the bike can be identified in the image. In order to categorize whether or not a person is wearing a helmet, this technique first detects the picture of a motorcycle and the driver. In this study, we used CNN models to tackle the challenge of identifying bikers and their helmets from surveillance images. When we'd finished collecting the photos that would make up our training dataset, we divided them into two piles: one for training data, and the other for testing data. The results of this experiment are based on the use of CNN models to classify images. To ensure the accuracy of the detection of the biker with and without a helmet in the image, all photographs will be examined and analyzed. Convolutional neural networks, such as CNNs, include several layers (and some other layers). The convolutional process is carried out by a number of filters in a convolutional layer. After comparing the results of earlier phases, we arrive at our final decision. The license plate number is found using morphological procedures on a segmented image. The license plate area will be improved (smoothed) using the dilation and erosion method by deleting any extra pixels from the plate's outer region. We will be able to differentiate the foreground and background after morphological processing. This number plate has been snatched away.

The implementation of number plate detection involves using image processing techniques and machine learning algorithms to automatically detect and recognize license plates in images or videos captured by cameras. The process typically involves the following steps:

1. Image Acquisition: Images or videos of vehicles are captured by cameras installed at strategic locations such as toll booths, parking lots, or traffic intersections.
2. Pre-processing: The acquired images are pre-processed to enhance the quality of the image, remove noise, and improve contrast. Techniques such as noise removal, color conversion, and image enhancement are applied to the image.
3. Region of Interest (ROI) Extraction: The region of interest (ROI) containing the license plate is extracted from the pre-processed image using techniques such as edge detection, morphological operations, and object detection.
4. Character Segmentation: The individual characters of the license plate are segmented using techniques such as connected component analysis and contour analysis.
5. Character Recognition: The segmented characters are recognized using machine learning algorithms such as Support Vector Machines (SVMs), Convolutional Neural Networks (CNNs), or other pattern recognition techniques.
6. Output Generation: The recognized characters are then used to generate the license plate number, which can be stored in a database, used for toll collection, parking management, or other applications.

The implementation of number plate detection has several benefits, including increased accuracy and speed of license plate recognition, reduced manual errors, and improved traffic management. However, the implementation also faces challenges such as variations in license plate size, shape, and color, which may require the use of multiple algorithms and techniques to ensure accurate detection and recognition. Additionally, the system may need to be trained on a large number of license plate samples to ensure high accuracy and performance.

Fig 2 : *Helmet Detection With And Without*

2.2 E-CHALLAN AUTOMATION

2.1.1 INTRODUCTION

Governance is a challenge in a country as diverse, vast and rapidly developing as India. India needs a new and latest technology for large-scale transformation and implementation of government plans. While India is among the fastest developing economies in the world, India's equitable growth remains a critical imperative. This project is an attempt in this very direction of e-governance for a country like India with a large population and high density. India's road network has grown at an annual rate of 4% since 1951. Along with the rural and urban population density the density of roads has risen in India. The rising population has resulted in more vehicles on roads. This has led to a high rate of accidents. One of the major reasons for the high number of accidents on the road is that traffic rules are violated and not followed. According to a survey, 78% of the accidents happen due to violation of traffic rules by the driver such as speeding, driving under the influence of alcohol or drugs, and hit and run cases. India needs a highly regulated foolproof system of governance to prevent these avoidable accidents and manage the traffic on the roads. A system which makes the people follow the rules and drive safely, without violating any rules. E-Challan is that foolproof regulated system. E-Challan is an online Egovernance system to facilitate the traffic managers to manage the traffic violation as well as for the drivers to manage the penalties. E-Challan provides a wide range of support needed for managing and monitoring traffic penalties. It is also a type of decentralized information system which allows all the stakeholders to access the needed information anytime anywhere. In the following sections a detailed methodology along and a comparative study with the past works is done to provide a detailed overview about the E-Challan system. This Project is mainly about an e-governance management system which allows different features to the stakeholders related to challan, vehicle details and license details. The Software has different types of users who are dependent on each other for the fully functioning of the application. The Software allows creating and maintaining a database containing information about different registered users, their license details and challans issued. The system also maintains a database of vehicles which are registered in the local RTO. This information can be used to verify drivers and in case of any violation of traffic rules to issue a challan to the driver. Also, the software allows the system administrator to access and update the databases when a new driver or vehicle is registered. The system admin also creates initial login credentials for the traffic department personnel. The administrator is the sole authority who can manipulate all the information in these databases.

The E-Challan Automation project is an initiative aimed at streamlining the process of traffic violation monitoring and control. The project aims to eliminate manual processing of traffic tickets and replace it with an automated system that is more efficient, accurate, and reliable.

The project involves the use of cameras, sensors, and other technologies to monitor traffic violations, such as over speeding, signal jumping, and lane violations. These devices capture images of the license plates of offending vehicles, which are then analyzed to determine the identity of the driver and the vehicle owner.

Once the identity of the offender is established, an e-challan is generated and sent to the offender's mobile phone or email address, along with the details of the violation and the amount of the fine. The offender can then pay the fine online, through a mobile app, or at a designated payment center.

The E-Challan Automation project has several advantages over the traditional manual process of issuing traffic tickets. First, it reduces the workload of traffic police officers, who can focus on other important duties instead of manually issuing tickets. Second, it eliminates the possibility of human error in issuing and processing tickets, which can lead to disputes and delays in collecting fines.

Moreover, the automated system ensures that all traffic violations are captured and recorded, making it easier to identify repeat offenders and enforce traffic laws more effectively. The project also has the potential to generate significant revenue for the government through the collection of fines, which can be used to improve road infrastructure and other transportation-related initiatives.

Overall, the E-Challan Automation project is a major step towards creating a safer, more efficient, and technology-driven transportation system.

2.2.2 MERITS, DEMERITS AND CHALLENGES

Merits:

- The System keeps track of the transactions in the RTO office.
- It is more user friendly: The section such as, registration, license, etc. are combined together in a single window.
- Use of RFID tags to check every registered vehicle.
- Every detail of vehicle is maintained in a database.
- Less time consuming

Demerits:

- Utilizing more of man power
- Manual work

- Less alert ness
- No on spot punishment

CHALLENGES:

The project mainly focuses on issue and view or pay challan along with details extraction of vehicles, license numbers and challan details. The project can be extended to a full stack functional website in future providing other features such as license creation, RTO vehicle registrations and many more. The system can also be modified by using the latest technologies as discussed in the literature survey like QR code and RFID scanner. This will limit human intervention and will result in a more efficient model of the existing system

While the E-Challan Automation project offers numerous benefits, there are also several challenges that need to be addressed for its successful implementation. Some of these challenges include:

1. Technical Challenges: The E-Challan Automation system relies on various technologies such as cameras, sensors, and automated license plate recognition systems. These technologies must be reliable, accurate, and able to function effectively in all weather conditions. Any technical failures in the system could result in incorrect or delayed e-challans being issued, which could impact the project's success.
2. Infrastructure Challenges: The success of the E-Challan Automation project is dependent on the availability and reliability of the necessary infrastructure, including the availability of network connectivity, power supply, and maintenance and support services. In some areas, these infrastructure challenges may limit the effectiveness of the system.
3. Legal Challenges: There may be legal challenges that need to be addressed in the implementation of the E-Challan Automation project. For example, privacy concerns related to the use of cameras and sensors to capture vehicle and driver information may need to be addressed. In addition, there may be legal challenges related to the enforcement of e-challans and the collection of fines.
4. Public Awareness: The success of the E-Challan Automation project also depends on the public's awareness and understanding of the system. Some drivers may not be aware of the system's existence, or may not understand how to pay e-challans. It is therefore important to educate the public about the system and how it works.
5. Resistance to Change: Finally, there may be resistance to change among stakeholders such as traffic police officers, who may be accustomed to the traditional manual process of

issuing traffic tickets. It is therefore important to provide adequate training and support to stakeholders to ensure a smooth transition to the new system.

2.2.3 IMPLEMENTATION OF E-CHALLAN AUTOMATION

The project has used a variety of front and back end frameworks for implementations such as:

1. HTML: For front-end development
2. CSS: For front-end development
3. JS: For animations and display time
4. Php: For front and back end connections, session creation and queries
5. AJAX: With Ajax, web application can send and retrieve data from a server asynchronously without interfering with the display and behavior of the existing page.
6. jQuery: For animations
7. MySQL: Back end development

The application provides a number of features such as display vehicle details, display driver details, pay challan and issue challan. The core application of the website is to issue and pay challan, working of which is described in the following section.

The implementation of E-Challan Automation involves the following steps:

1. Selection of Technology: The first step is to select the appropriate technology for the system. The technology used in the E-Challan Automation project includes cameras, sensors, automatic number plate recognition (ANPR) systems, and software for generating and sending e-challans.
2. Installation of Equipment: After selecting the technology, the next step is to install the equipment, including cameras, sensors, and ANPR systems, at strategic locations, such as busy intersections, highways, and toll booths.
3. Integration with Backend System: Once the equipment is installed, it needs to be integrated with the backend system, which includes software for generating and sending e-challans, and a database for storing the data of the violators.
4. Training of Personnel: The personnel responsible for monitoring the traffic violations need to be trained on how to operate and maintain the equipment and the backend system.
5. Pilot Test: A pilot test of the system is conducted to identify any glitches or issues and to ensure that the system is functioning as intended.
6. Rollout: After successful pilot testing, the E-Challan Automation system can be rolled out on a larger scale.
7. Promotion and Awareness: The public needs to be made aware of the new system and the penalties for traffic violations. This can be achieved through various awareness campaigns, social media promotions, and other communication channels.

The implementation of E-Challan Automation requires collaboration and coordination between different government agencies, such as traffic police, transport department, and IT department. The system can significantly reduce traffic congestion and promote traffic discipline, leading to a safer and more efficient transportation system.

Fig 3: *Traffic Cops to Go After Vehicles Without Helmet*

FIG 4: *E-Challan Processing Through the Nearest Police Control Room*

2.3 IOT BASED TOLL COLLECTION SYSTEM USING RFID

2.3.1 INTRODUCTION

The toll collection system using RFID is a fully digitalize system which has no need of any manual operations at the toll stations. To reduce wait times at toll booths and eliminate need for cash, engineers turned to microcontrollers, RFIDs, and sensors to create an automated toll collecting system. Automated toll collection utilizing RFID is primary focus of this project. The word "automation" refers to use of machines to take place of people in a process. In the meanwhile, we'll take a look back at the Toll plazas' history. Toll plazas were manually operated prior. Two persons are responsible for unlocking and unlocking the gate, while two others are responsible for receiving and storing money and other valuables. Semiautomatic toll plazas began to be used in 1995, after the Expressways had been completed. In this case, only two people are needed to operate a single booth.

However, this is human-free plaza that we are about to see. The project's most fundamental need in real life is: It saves time and money by reducing the need for manual operation, avoiding fuel loss, and reducing time spent collecting tolls. Toll collecting procedure for a single car used to take one minute under the old method. Just 40 to 42 seconds will be needed to finish whole procedure using this automated method. Because the process takes less time to complete, there will be less traffic, and less traffic means less fuel waste, which is the whole point of creating roads in the first place: to cut down on travel time and decrease financial losses.

The Internet of Things (IoT) has revolutionized the way we interact with the world around us. One of the key areas where IoT is being implemented is in the transportation industry, where it has the potential to make our roads safer, more efficient, and less congested. The use of RFID technology in toll collection is a prime example of how IoT is being utilized in this industry.

An IoT-based toll collection system using RFID technology is a project that aims to streamline the toll collection process by eliminating the need for toll booths and cash transactions. This system uses RFID tags that are attached to the windshield of vehicles, which can be read by RFID readers installed on the road.

The system automatically deducts the toll fee from the vehicle owner's account as soon as the RFID tag is scanned. This eliminates the need for drivers to stop and pay toll fees manually, which can lead to long queues and traffic congestion.

In this project, we will explore the design, development, and implementation of an IoT-based toll collection system using RFID technology. We will discuss the advantages of using this system, the technical details of how it works, and the potential challenges that may arise during its implementation. The goal of this project is to create a more efficient, cost-effective, and safer toll collection system for the benefit of drivers and transportation authority's alike.

An IoT-based toll collection system using RFID (Radio Frequency Identification) technology is a modern and efficient way of collecting tolls on highways and other roads. This system enables automated toll collection, eliminating the need for physical toll booths and reducing traffic congestion and delays.

The RFID-based toll collection system works by installing RFID readers at the toll plaza and RFID tags on the vehicles. When a vehicle approaches the toll plaza, the RFID reader detects the RFID tag on the vehicle and automatically deducts the toll amount from the driver's account. The driver can then continue on the road without having to stop and pay the toll manually.

The implementation of an IoT-based toll collection system using RFID offers several benefits, including:

1. Improved Efficiency: RFID-based toll collection systems are faster and more efficient than traditional toll booths, as vehicles do not need to stop to pay tolls. This reduces traffic congestion and delays, and improves overall traffic flow.
2. Reduced Operating Costs: An IoT-based toll collection system using RFID eliminates the need for physical toll booths and reduces the cost of toll collection. This results in cost savings for toll operators, who can pass on the benefits to their customers.
3. Increased Revenue: An IoT-based toll collection system using RFID enables toll operators to collect tolls more efficiently and accurately, resulting in increased revenue generation.
4. Improved Security: The use of RFID technology in toll collection systems enhances security by reducing the risk of theft and fraud, as the system automatically deducts tolls from the driver's account.
5. Data Collection and Analysis: An IoT-based toll collection system using RFID allows toll operators to collect and analyze data on traffic flow, toll revenue, and other important metrics. This data can be used to improve operations and make data-driven decisions.

Implementing an IoT-based toll collection system using RFID requires careful planning, technical expertise, and effective project management. However, the benefits of such a system make it a worthwhile investment for toll operators and governments looking to improve traffic flow and generate revenue for infrastructure development and maintenance.

2.3.2 MERITS, DEMERITS AND CHALLENGES

Merits:

- Easy Process
- Less manual work
- Faster way to collecting tolls

Demerits:

- If Vehicles not having Fast Tag, it requires manual work.
- It won't check the rider wearing seat belt or not.

Challenges:

The traffic police personnel login in the web- application using his credentials. On his desktop he/she can see all the functionalities which can be used by him/her. For issuing the challan the personnel click on the issue challan icon and an issue challan form is opened.

The implementation of an IoT-based toll collection system using RFID technology presents several challenges that must be addressed for its successful deployment. Some of the key challenges include:

1. Integration with Existing Infrastructure: The first challenge is integrating the new IoT-based toll collection system with existing toll infrastructure. This may require retrofitting existing toll booths and payment systems or building entirely new toll plazas.
2. Compatibility with Multiple RFID Systems: RFID technology is available in multiple frequency bands, and different regions may have different systems in place. Ensuring compatibility with different RFID systems is essential to enable seamless toll collection across regions.
3. Accuracy and Reliability: The RFID system must be accurate and reliable in reading the toll tag data, regardless of the speed and distance of the vehicle passing through the toll plaza. Any errors or delays in toll collection can cause congestion and inconvenience to the commuters.
4. Data Security and Privacy: The RFID-based toll collection system collects personal data of the vehicle owner and their journey information, which must be stored and transmitted securely to avoid any privacy violations or cyberattacks.
5. Maintenance and Upgrades: The RFID system needs to be regularly maintained and upgraded to ensure its continued effectiveness and reliability. This involves conducting routine maintenance checks, performing repairs and upgrades as needed, and ensuring that the system remains compliant with relevant regulations and standards.
6. Cost-effectiveness: The deployment and maintenance of the IoT-based toll collection system using RFID technology can be expensive. It is important to evaluate the costs and

benefits of the system to ensure that it is a cost-effective solution in the long run.

Addressing these challenges requires a combination of technical expertise, effective project management strategies, and stakeholder cooperation. With proper planning and execution, an IoT-based toll collection system using RFID technology can improve road safety and traffic flow while providing a reliable source of revenue for infrastructure development and maintenance.

2.3.3 IMPLEMENTATION OF TOLL COLLECTION SYSTEM

Sensor technology- It is used to detect vehicle arrived and the speed is also detected using ultrasonic sensor. If the speed of vehicle and if the speed exceeds then vehicle must pay the fine along with toll tax. RFID technology- It mainly aims for identifying the card sensed on the toll plazas. RFID tag is scanned through RFID reader, by Radio frequency in form of waves within a particular distance. RFID tag is linked with owner account, each RFID tag has a wallet to which the owner has to top-up with prepaid amount.

IOT technology-Node MCU is used in this study to send a message to an IoT server, which has global coverage, in order to report amount deducted. It is presently most effective means of communication. Using IoT to submit whole amount, with or without a fee, to authorities is best option. Node MCU technology: To report the amount deducted, the work utilizes the Node MCU to send message through Internet of Things (IoT) server . To report the Total amount with or without fine deducted to experts, IoT is best possible solution. The normal criteria is to make a payment using RFID reader. If there is no balance in RFID tag then should make a recharge to it by using the keypad.

The implementation of a toll collection system involves several steps and considerations to ensure its successful deployment. Some of the key steps include:

1. Planning and Design: The first step is to plan and design the toll collection system. This involves identifying the toll plazas, determining the types of tolls to be collected, selecting the toll collection technology, and designing the layout of the toll plaza.
2. Procurement of Equipment: Once the toll collection system has been designed, the necessary equipment needs to be procured, including toll booths, electronic toll collection (ETC) systems, cameras, sensors, and other hardware and software.
3. Installation and Configuration: The next step is to install and configure the toll collection equipment at the designated toll plaza. This involves setting up the toll booths, installing the ETC systems, and configuring the cameras and sensors.
4. Testing and Commissioning: Before the toll collection system can be put into operation, it must be thoroughly tested and commissioned. This involves testing the ETC systems,

cameras, and sensors to ensure they are functioning properly, and conducting trial runs to ensure that the toll collection process is efficient and accurate.

5. Training and Support: Once the toll collection system is operational, it is important to provide adequate training and support to toll operators and maintenance personnel. This includes training on how to operate and maintain the equipment, as well as providing ongoing technical support and assistance.
6. Maintenance and Upgrades: The toll collection system must be regularly maintained and upgraded to ensure its continued effectiveness and reliability. This involves conducting routine maintenance checks, performing repairs and upgrades as needed, and ensuring that the system remains compliant with relevant regulations and standards.

Implementing a toll collection system can be a complex process that requires careful planning, technical expertise, and effective project management. With proper planning and execution, however, a toll collection system can improve road safety and traffic flow, while providing a reliable source of revenue for infrastructure development and maintenance.

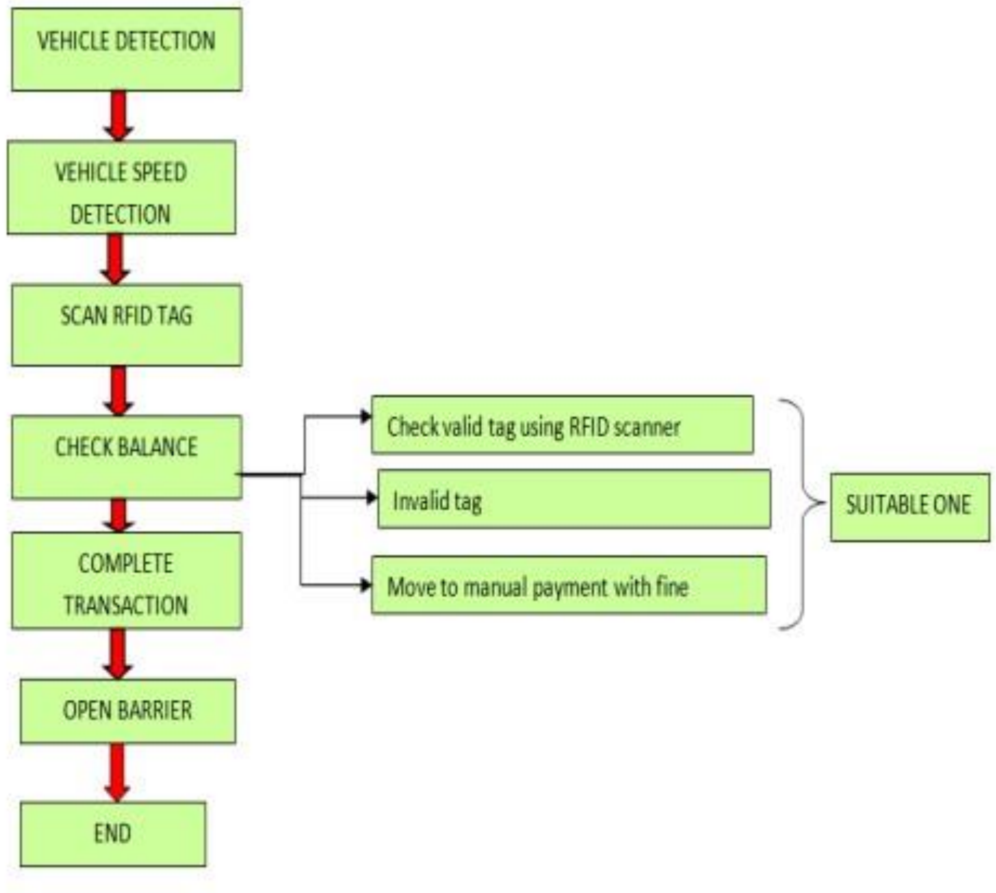


Fig 5: *Flowchart of the system*

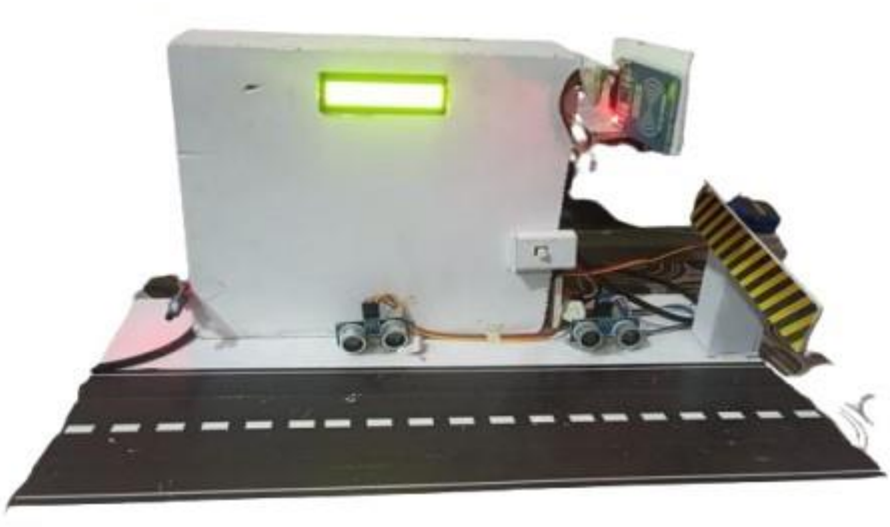


Fig 6: *Model for IOT Based Model of TOLL Collection System*

Fig 6 shows the model developed for IOT based Toll collection system using RFID. This is complete over look of the model developed using IOT.

Fig 7: *Toll Pass Detection of Fasttag & Allowing to Pass Vehicle through it*

CHAPTER 3
PROPOSED SYSTEM

CHAPTER 3

PROPOSED SYSTEM

3.1 Objective of Proposed Model

Our main objective is to develop a system which detects the motorcycle rider with helmet and allows him to fuel filling at pumps of petroleum bunk.

Here we allowing the rider for fuelling after detection of helmet, if helmet detected successfully then the system process the instructions to the Arduino then the gates raised and system allows the biker for fuelling.

Implementing the helmet detection at fuel bunks using the webcam or cctv using python iot models here we are detecting the helmet of the rider who are at petroleum pump here it detects and then allows the rider to fill fuel
So, here firstly it detects helmet if helmet detected successfully then a gate will rised and allows the rider to fill fuel.
Here we using technologies **Tensorflow**, **YOLO3 (You Only Look Once)**, **CNN**(**Convolutional Neural Network**), **openCv** using **Python**.

The objective of rider helmet detection for fuel filling is to ensure the safety of riders and petrol station attendants by preventing fueling of motorcycles with riders not wearing helmets. This technology aims to enforce compliance with helmet laws and prevent accidents that can result from riding without a helmet. By using an automated system to detect the presence of a rider's helmet, petrol station attendants can refuse to dispense fuel to motorcycles whose riders are not wearing helmets, promoting rider safety and reducing the incidence of accidents. The ultimate goal of this system is to promote safer roads and protect the lives of riders and other road users.

3.2 Stepwise Implementation

Here is a proposed system data for the rider helmet detection for fuel filling project:

1. Hardware Components:

    - CCTV Camera with IR Night Vision
    - Computer with high-end processor and graphics card
    - Helmet detection software
    - Fuel Pump Dispenser

2. Software Components:

    - Image Processing and Object Detection Libraries
    - Machine Learning Libraries
    - Database Management System

3. System Architecture:

    - The CCTV camera will be installed near the fuel pump dispenser.

- The camera will capture live footage of the rider as he approaches the dispenser to fuel up.
- The live footage will be analysed using the image processing and object detection libraries to detect if the rider is wearing a helmet or not.
- The machine learning libraries will analyse the footage to identify the rider and match him with a pre-existing database of registered riders.
- If the rider is not wearing a helmet, the system will notify the fuel pump dispenser to stop fuelling.
- The database management system will keep track of all the fuel fillings and maintain a record of riders who are not wearing helmets.

4. Implementation:

   - The system can be implemented at fuel stations or petrol pumps where riders can fuel up their vehicles.
   - The system can be integrated with existing fuel dispensers and can be controlled by the fuel station staff.
   - The system can be connected to a central database where the data can be stored and accessed by relevant authorities.
   - The system can be tested and fine-tuned to achieve maximum accuracy and efficiency.

Overall, this proposed system will help promote road safety and encourage riders to wear helmets while riding. It will also help authorities monitor and enforce the laws related to wearing helmets while riding a motorcycle or scooter.

ABOUT THE TECHNOLOGIES USED IN THE PROJECT

PYTHON LANGUAGE:

Python is an ideal language for project reports due to its simplicity, readability, and versatility. Python has a large and active community of developers who have created many libraries and frameworks that can be used for various tasks, such as data analysis, visualization, web development, and machine learning.

When writing a project report in Python, there are a few key components to consider:

1. Data Preparation: If the project involves data analysis, it is important to prepare and clean the data before analysis. Python has many libraries, such as Pandas and NumPy, that can be used to manipulate and prepare data.
2. Analysis and Visualization: Once the data is prepared, Python can be used to perform analysis and create visualizations. Libraries such as Matplotlib and Seaborn can be used to create visualizations, while SciPy and Scikit-learn can be used for statistical analysis and machine learning.
3. Code Organization: When writing code for a project report, it is important to ensure that the code is organized and easy to read. This can be achieved by using comments, descriptive variable names, and separating code into functions or modules.
4. Reporting: Finally, Python can be used to generate the actual report. There are many libraries, such as Jupyter Notebook and Sphinx, that can be used to generate reports in various formats, including HTML, PDF, and LaTeX.

Overall, Python is a powerful tool for writing project reports. Its simplicity, versatility, and extensive libraries make it an ideal language for data analysis, visualization, and reporting.

TensorFlow:

TensorFlow is an open-source software library for dataflow and differentiable programming across a range of tasks. It is a popular machine learning framework designed by Google Brain Team to simplify the process of building and training machine learning models.

TensorFlow provides a high-level interface for building and deploying machine learning models, making it easy for developers to create complex neural networks with minimal code. It has a flexible architecture that allows for efficient computation on CPUs, GPUs, and TPUs, making it ideal for large-scale training and deployment.

TensorFlow is designed to be used with Python, a popular programming language, and provides a Python API for building and deploying machine learning models. This makes it easy for developers to integrate TensorFlow into their existing Python projects.

In a project report, TensorFlow can be used for a variety of machine learning tasks, including image recognition, natural language processing, and predictive analytics. It allows developers to easily build and train complex models, while also providing a wide range of pre-built models that can be easily adapted to different use cases.

In summary, TensorFlow is a powerful machine learning framework that can be used to build and train machine learning models. It provides a high-level interface for building and deploying models, making it easy for developers to create complex neural networks with minimal code. When used in a project report, TensorFlow can be used to address a wide range of machine learning tasks, making it an essential tool for developers and data scientists.

The Python YOLO3 (You Only Look Once v3):

YOLO3 is an open-source neural network-based object detection algorithm that can detect objects in real-time. It was developed by Joseph Redmon, Ali Farhadi, and others at the University of Washington, and is now widely used in computer vision applications.

The YOLO3 algorithm uses a single convolutional neural network (CNN) that simultaneously predicts bounding boxes and class probabilities for multiple objects in an image. The algorithm divides the input image into a grid and each grid cell is responsible for detecting objects within that cell. The output of the algorithm is a list of bounding boxes and class probabilities for each detected object.

In Python, the YOLO3 algorithm is implemented using the Keras deep learning library. The YOLO3 model can be pre-trained on a large dataset such as COCO (Common Objects in Context) or trained on a custom dataset for specific object detection tasks. The pre-trained YOLO3 model can be fine-tuned on a smaller dataset to improve object detection accuracy for specific classes of objects.

The Python YOLO3 algorithm is a powerful tool for object detection tasks such as pedestrian detection, traffic monitoring, and surveillance. It has been used in various projects and research studies, including autonomous vehicles, facial recognition, and object tracking.

In summary, the Python YOLO3 algorithm is an effective and efficient object detection algorithm that is easy to use and customize. Its versatility and real-time performance make it a valuable tool for many computer vision applications.

Convolutional Neural Networks (CNNs):

CNN are a type of deep learning algorithm that are commonly used for image recognition and computer vision tasks. Python is a popular programming language for implementing CNNs due to its ease of use, extensive library support, and strong community.

To implement a CNN in Python, one can use popular libraries such as TensorFlow, Keras, and PyTorch. These libraries provide a range of tools and functions for building, training, and evaluating CNNs.

The basic structure of a CNN consists of multiple layers, including convolutional layers, pooling layers, and fully connected layers. Convolutional layers use filters to extract features from images, while pooling layers reduce the dimensionality of the extracted features. Fully connected layers use the extracted features to make predictions about the input image.

To train a CNN in Python, one needs to provide a dataset of images with corresponding labels. This dataset is then used to train the CNN using backpropagation, a method for updating the weights of the network to improve its accuracy.

Once the CNN is trained, it can be used to make predictions on new images. This can be done by feeding the new image through the network and obtaining the output of the final layer, which represents the predicted class or label.

In summary, implementing a CNN in Python requires knowledge of the architecture of the network, the available libraries, and the principles of training and evaluation. However, with the appropriate tools and skills, Python CNNs can be a powerful tool for image recognition and computer vision applications.

OpenCV (Open-Source Computer Vision):

OpenCV is a popular open-source library for computer vision and image processing in Python. OpenCV provides a wide range of functions and algorithms for image processing, feature detection, object detection, and recognition. It is widely used for applications such as robotics, surveillance, medical imaging, and augmented reality.

OpenCV in Python is accessed through the Python module cv2. This module provides functions to read and write image files, manipulate images, and perform various image processing tasks. Some of the common tasks that can be performed with OpenCV in Python include image filtering, image thresholding, edge detection, image segmentation, feature detection, and object detection.

OpenCV in Python can be used for a wide range of projects, including face detection, object tracking, gesture recognition, and many more. For example, OpenCV can be used to detect faces in a video stream and track them as they move across the screen. It can also be used to detect objects in real-time and trigger actions based on their presence.

In a project report, the use of OpenCV in Python should be described in detail, including the specific algorithms and functions used, the reasons for choosing OpenCV, and any challenges encountered during implementation. Additionally, the report should include a detailed explanation of the project goals and objectives, the methods used to achieve them, and the results achieved.

This will help readers understand the impact and significance of using OpenCV in Python for the project.

IOT (INTERNET OF THINGS):

Refers to the connection of physical devices, vehicles, buildings, and other objects to the internet, allowing them to collect and exchange data. The data collected by these devices can be analyzed to gain insights and inform decisions, leading to more efficient and effective processes, improved customer experiences, and new revenue streams.

In the context of a project report, IoT can be a powerful tool for improving the performance and outcomes of various projects. By integrating IoT devices and sensors into a project, teams can collect and analyze data in real-time, allowing them to make informed decisions quickly and effectively.

For example, in a construction project, IoT sensors can be used to monitor building materials, equipment, and personnel in real-time. This information can be used to optimize resource allocation, reduce waste, and improve safety. Similarly, in a manufacturing project, IoT sensors can be used to track the performance of machines and equipment, enabling predictive maintenance and reducing downtime.

In a project report, the use of IoT can also demonstrate a commitment to sustainability and environmental responsibility. For example, IoT sensors can be used to monitor energy consumption in buildings, allowing for more efficient use of resources and reduced energy waste.

Overall, the use of IoT in project management can lead to improved efficiency, cost savings, and better outcomes. When including IoT in a project report, it is important to highlight the specific ways in which IoT technology has been integrated and the benefits that it has provided to the project.

IOT ARDUINO:

IOT ARDUINO is a popular platform for building Internet of Things (IoT) projects. It combines the power and flexibility of the Arduino microcontroller board with wireless connectivity, allowing devices to communicate with each other and the internet. Arduino is an open-source hardware and software platform that has gained a large community of developers and enthusiasts who have created numerous libraries and resources for building IoT projects.

The IoT Arduino platform provides a variety of options for wireless communication, including Wi-Fi, Bluetooth, and cellular. These communication options allow IoT devices to connect to the internet and communicate with other devices or cloud services. The Arduino platform also supports a range of sensors and actuators, making it possible to build a wide range of IoT applications.

One of the key advantages of using the IoT Arduino platform for building IoT projects is its ease of use. Arduino's simple and intuitive programming language, coupled with a vast library of pre-existing code and tutorials, makes it easy for beginners to get started with IoT development. Additionally, the platform is highly customizable, with a variety of add-on modules and shields available to extend its functionality.

Another advantage of using IoT Arduino is its low cost. Arduino boards and modules are relatively inexpensive compared to other microcontroller-based development boards, making it possible to build IoT projects at a low cost. This makes it ideal for small-scale and prototype projects.

Overall, IoT Arduino is a powerful platform for building IoT projects that combines the flexibility of the Arduino microcontroller with wireless connectivity. Its ease of use, low cost, and vast library of resources make it a popular choice for developers and hobbyists alike.

SOFTWARE REQUIREMENTS:

The software requirements for a rider helmet detection system for fuel filling project typically include the following:

1. Operating System: The system should be compatible with a range of operating systems, including Windows, Linux, and macOS.
2. Image Processing Library: The system should use an image processing library to analyze the images captured by the cameras installed at the fuel filling station. The library should be able to detect the presence or absence of a helmet on the rider.
3. Artificial Intelligence/Machine Learning Framework: The system may incorporate an artificial intelligence/machine learning framework for improved accuracy and reliability. The framework should be trained to recognize helmet-wearing riders and non-helmet wearing riders based on a set of predefined criteria.
4. Database Management System: The system should use a database management system to store rider data, including their helmet status and any violations. The system should be able to retrieve data from the database quickly and efficiently.
5. User Interface: The system should have a user-friendly interface for the fuel station operator to view the rider's helmet status, and any violation status, and to take appropriate actions if necessary.
6. Notification System: The system should have a notification system to alert the operator of any violations or anomalies detected in the system. The system should also have the ability to generate reports and analytics for management purposes.
7. Integration with Fuel Dispensing System: The system should integrate with the fuel dispensing system to ensure that the rider's helmet status is checked before fuel dispensing. If the rider is not wearing a helmet, the system should prevent fuel dispensing and notify the operator.

These software requirements are critical to the successful implementation of a rider helmet detection system for fuel filling project, and they must be carefully considered during the design and development phases of the project.

HARDWARE REQUIREMENTS:

The hardware requirements for a rider helmet detection system for fuel filling project may vary depending on the specific implementation and technology used. However, some of the key hardware components that may be required include:

1. Camera: A camera is required to capture the images of the rider and their helmet. It should be of high resolution and capable of capturing clear images in low light conditions.
2. Image Processing Unit: An image processing unit is required to process the images captured by the camera and detect whether the rider is wearing a helmet or not. This may include hardware components such as microcontrollers, processors, or dedicated image processing units.
3. Sensors: Sensors may be used to detect when a rider approaches the fuel station and triggers the system. This may include infrared sensors, proximity sensors, or motion sensors.
4. Display Unit: A display unit may be used to provide feedback to the rider on whether they are wearing a helmet or not. This may include an LED display, an LCD display, or an audio feedback system.
5. Communication Modules: Communication modules may be required to transmit the helmet detection data to a central server or monitoring system. This may include Wi-Fi, Bluetooth, or cellular communication modules.
6. Power Supply: A power supply is required to power the various hardware components of the system. This may include batteries, AC/DC adapters, or solar panels.

Overall, the hardware requirements for a rider helmet detection system for fuel filling project will depend on the specific implementation and requirements of the project. It is important to carefully consider the hardware components required to ensure that the system is accurate, reliable, and cost-effective.

USER INTERFACE:

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

REQUIREMENTS SPECIFIFCATIONS:

The requirements specifications for a rider helmet detection for fuel filling project may include:

1. Detection Accuracy: The system must be accurate in detecting whether the rider is wearing a helmet or not. The accuracy rate should be high enough to avoid any false alarms or missed detections.
2. Real-time Detection: The system must be capable of detecting the rider's helmet status in real-time, i.e., within a few seconds of the rider entering the fuel filling station.
3. Non-intrusive Detection: The detection system must be non-intrusive and not require any physical contact with the rider. This is important for the safety and comfort of the rider.
4. Compatibility with Different Helmet Types: The system must be compatible with different types of helmets and should be able to detect whether the rider is wearing a full-face helmet, an open-face helmet, or a half-face helmet.
5. Integration with Fuel Dispensing System: The system must be integrated with the fuel dispensing system and should be able to prevent fuel filling if the rider is not wearing a helmet.
6. Low Maintenance: The system should be easy to maintain and should not require frequent maintenance or replacement of parts.
7. User-friendly Interface: The system should have a user-friendly interface that is easy to operate and understand for both the fuel station attendant and the rider.
8. Data Security and Privacy: The system must be designed to ensure data security and privacy of the rider's personal information.
9. Compliance with Safety Regulations: The system must comply with all safety regulations and standards to ensure the safety of the rider and the fuel filling station.
10. Cost-effectiveness: The system should be cost-effective and should provide a reasonable return on investment for the fuel filling station.

FUNCTIONAL REQUIREMENTS:

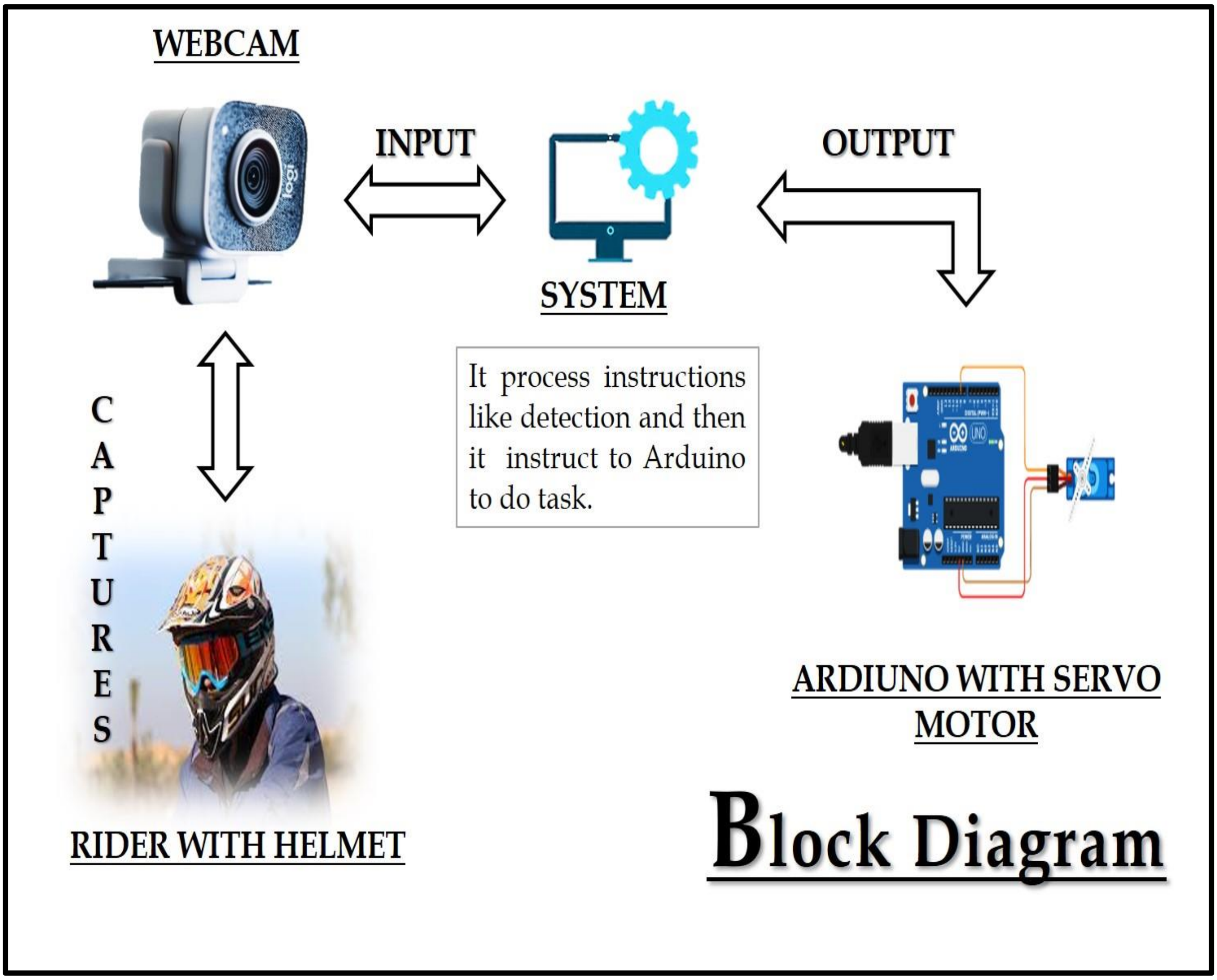
The functional requirements for a rider helmet detection system for fuel filling project may include:

1. Detection of Helmet: The system should be able to detect whether the rider is wearing a helmet or not. This can be achieved using a camera or a sensor placed at the fuel station.
2. Real-time Notification: Once the system detects that the rider is not wearing a helmet, it should notify the fuel station attendant or the rider in real-time.
3. Display: The system should display a message on the screen or through a speaker, notifying the rider to wear a helmet before filling the fuel.
4. Security: The system should be secure and tamper-proof to prevent any misuse or unauthorized access.
5. Compatibility: The system should be compatible with different types of helmets and bikes to ensure that it works for a wide range of riders.
6. Reliability: The system should be reliable and accurate in detecting whether the rider is wearing a helmet or not. False positives or false negatives could cause inconvenience to the rider and fuel station attendants.
7. User-friendly: The system should be user-friendly and easy to operate for both the fuel station attendants and the riders.

8. Data storage: The system should store the data related to the rider and their helmet, including the date and time of the detection, to ensure compliance with safety regulations.
9. Maintenance: The system should be easy to maintain and should require minimal upkeep to ensure its continued effectiveness.
10. Integration: The system should be easily integrated with the fuel station's existing infrastructure and payment systems to provide a seamless and efficient experience for the riders.

By meeting these functional requirements, a rider helmet detection system for fuel filling project can help promote road safety and ensure compliance with safety regulations.

3.3 BLOCK DIAGRAM



UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business Modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

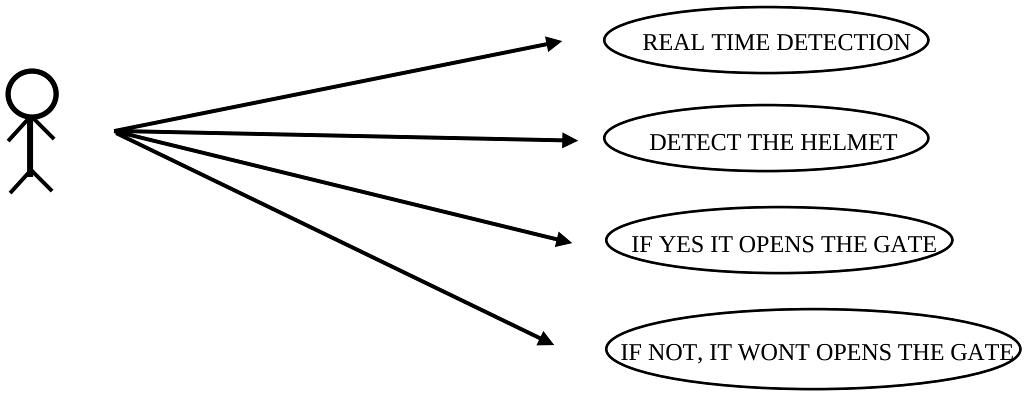
The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM:

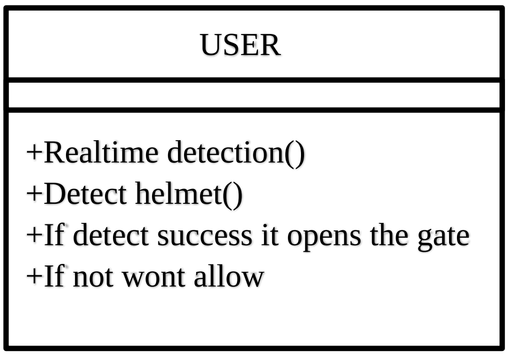
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

UML DIAGRAM



CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



IMPLEMENTATION

Alogorithm
Yolo

```
def loadLibraries(): #function to load yolov3 model weight and class labels
	global class_labels
	global cnn_model
	global cnn_layer_names
	class_labels = open('yolov3model/yolov3-labels').read().strip().split('\n') #reading labels
from yolov3 model
	print(str(class_labels)+" == "+str(len(class_labels)))
	cnn_model = cv.dnn.readNetFromDarknet('yolov3model/yolov3.cfg',
'yolov3model/yolov3.weights') #reading model
	cnn_layer_names = cnn_model.getLayerNames() #getting layers from cnn model
	cnn_layer_names = [cnn_layer_names[i[0] - 1] for i in
cnn_model.getUnconnectedOutLayers()] #assigning all layers

def detectFromImage(imagename): #function to detect object from images
	#random colors to assign unique color to each label
	label_colors = (0,255,0)#np.random.randint(0,255,size=(len(class_labels),3),dtype='uint8')
	try:
		image = cv.imread(imagename) #image reading
		image_height, image_width = image.shape[:2] #converting image to two dimensional
array
	except:
		raise 'Invalid image path'
	finally:
		image, _, _, _, _ = detectObject(cnn_model, cnn_layer_names, image_height,
image_width, image, label_colors, class_labels,indexno)#calling detection function
		displayImage(image,0)#display image with detected objects label

def detectFromVideo(videoFile): #function to read objects from video

	#random colors to assign unique color to each label
	label_colors = (0,255,0)#np.random.randint(0,255,size=(len(class_labels),3),dtype='uint8')
	indexno = 0
	try:

		video = cv.VideoCapture(videoFile)
		frame_height, frame_width = None, None  #reading video from given path
		video_writer = None
	except:
		raise 'Unable to load video'
	finally:
		while True:
			frame_grabbed, frames = video.read() #taking each frame from video
			#print(frame_grabbed)
			if not frame_grabbed: #condition to check whether video loaded or not
				break
			if frame_width is None or frame_height is None:
				frame_height, frame_width = frames.shape[:2] #detecting object from frame
			frames, _, _, _, _ = detectObject(cnn_model, cnn_layer_names, frame_height,
frame_width, frames, label_colors, class_labels,indexno)
```

```
                #displayImage(frames,index)
                #indexno = indexno + 1
                print(indexno)
                if indexno == 5:
                    video.release()
                    break

        print ("Releasing resources")
        #video_writer.release()
        video.release()


if __name__ == '__main__':
        loadLibraries()
        print("sample commands to run code with image or video")
        print("python yolo.py image input_image_path")
        print("python yolo.py video input_video_path")
        if len(sys.argv) == 3:
                if sys.argv[1] == 'image':
                        detectFromImage(sys.argv[2])
                elif sys.argv[1] == 'video':
                        detectFromVideo(sys.argv[2])
                else:
                        print("invalid input")
        else:
                print("follow sample command to run code")

            #video_path = None
            #video_output_path = "out.avi"
```

Sample Code

```
from tkinter import *
import tkinter
from tkinter import filedialog
import numpy as np
from tkinter.filedialog import askopenfilename
import pandas as pd
from tkinter import simpledialog
import numpy as np
import cv2 as cv
import subprocess
import time
import os
from yoloDetection import detectObject, displayImage
import sys
from time import sleep
from tkinter import messagebox
import pytesseract as tess
from keras.models import model_from_json
from keras.utils.np_utils import to_categorical

main = tkinter.Tk()
main.title("Helmet Detection") #designing main screen
```

```
main.geometry("800x700")

global filename
global loaded_model

global class_labels
global cnn_model
global cnn_layer_names

frame_count = 0
frame_count_out=0

confThreshold = 0.5
nmsThreshold = 0.4
inpWidth = 416
inpHeight = 416
global option
labels_value = []
with open("Models/labels.txt", "r") as file: #reading MRC dictionary
    for line in file:
        line = line.strip('\n')
        line = line.strip()
        labels_value.append(line)
    file.close()


with open('Models/model.json', "r") as json_file:
    loaded_model_json = json_file.read()
    plate_detecter = model_from_json(loaded_model_json)

plate_detecter.load_weights("Models/model_weights.h5")
plate_detecter._make_predict_function()

classesFile = "Models/obj.names";
classes = None

with open(classesFile, 'rt') as f:
    classes = f.read().rstrip('\n').split('\n')

modelConfiguration = "Models/yolov3-obj.cfg";
modelWeights = "Models/yolov3-obj_2400.weights";

net = cv.dnn.readNetFromDarknet(modelConfiguration, modelWeights)
net.setPreferableBackend(cv.dnn.DNN_BACKEND_OPENCV)
net.setPreferableTarget(cv.dnn.DNN_TARGET_CPU)

def getOutputsNames(net):
    layersNames = net.getLayerNames()
    return [layersNames[i[0] - 1] for i in net.getUnconnectedOutLayers()]

def loadLibraries(): #function to load yolov3 model weight and class labels
        global class_labels
        global cnn_model
```

```
        global cnn_layer_names
        class_labels = open('yolov3model/yolov3-labels').read().strip().split('\n') #reading labels
from yolov3 model
        print(str(class_labels)+" == "+str(len(class_labels)))
        cnn_model = cv.dnn.readNetFromDarknet('yolov3model/yolov3.cfg',
'yolov3model/yolov3.weights') #reading model
        cnn_layer_names = cnn_model.getLayerNames() #getting layers from cnn model
        cnn_layer_names = [cnn_layer_names[i[0] - 1] for i in
cnn_model.getUnconnectedOutLayers()] #assigning all layers


def upload(): #function to upload tweeter profile
    global filename
    filename = filedialog.askopenfilename(initialdir="bikes")
    #messagebox.showinfo("File Information", "image file loaded")



def detectBike():
    global option
    option = 0
    indexno = 0
    label_colors = (0,255,0)
    try:
        image = cv.imread(filename)
        image_height, image_width = image.shape[:2]
    except:
        raise 'Invalid image path'
    finally:
        image, ops = detectObject(cnn_model, cnn_layer_names, image_height, image_width,
image, label_colors, class_labels,indexno)
        if ops == 1:
            displayImage(image,0)#display image with detected objects label
            option = 1
        else:
            displayImage(image,0)




def drawPred(classId, conf, left, top, right, bottom,frame,option):
    global frame_count
    #cv.rectangle(frame, (left, top), (right, bottom), (255, 178, 50), 3)
    label = '%.2f' % conf
    if classes:
        assert(classId < len(classes))
        label = '%s:%s' % (classes[classId], label)
    labelSize, baseLine = cv.getTextSize(label, cv.FONT_HERSHEY_SIMPLEX, 0.5, 1)
    top = max(top, labelSize[1])
    label_name,label_conf = label.split(':')
    print(label_name+" === "+str(conf)+"==  "+str(option))
    if label_name == 'Helmet' and conf > 0.50:
        if option == 0 and conf > 0.90:
```

```
            cv.rectangle(frame, (left, top - round(1.5*labelSize[1])), (left + round(1.5*labelSize[0]),
top + baseLine), (255, 255, 255), cv.FILLED)
            cv.putText(frame, label, (left, top), cv.FONT_HERSHEY_SIMPLEX, 0.75, (0,0,0), 1)
            frame_count+=1
        if option == 0 and conf < 0.90:
            cv.putText(frame, "Helmet Not detected", (10, top), cv.FONT_HERSHEY_SIMPLEX,
0.75, (0,255,0), 2)
            frame_count+=1
            img = cv.imread(filename)
            img = cv.resize(img, (64,64))
            im2arr = np.array(img)
            im2arr = im2arr.reshape(1,64,64,3)
            X = np.asarray(im2arr)
            X = X.astype('float32')
            X = X/255
            preds = plate_detecter.predict(X)
            predict = np.argmax(preds)
            #img = cv.imread(filename)
            #img = cv.resize(img,(500,500))
            #text = tess.image_to_string(img, lang='eng')
            #text = text.replace("\n"," ")
            #messagebox.showinfo("Number Plate Detection Result", "Number plate detected as
"+text)
            textarea.insert(END,filename+"\n\n")
            textarea.insert(END,"Number plate detected as "+str(labels_value[predict]))
        if option == 1:
            cv.rectangle(frame, (left, top - round(1.5*labelSize[1])), (left + round(1.5*labelSize[0]),
top + baseLine), (255, 255, 255), cv.FILLED)
            cv.putText(frame, label, (left, top), cv.FONT_HERSHEY_SIMPLEX, 0.75, (0,0,0), 1)
            frame_count+=1


    if(frame_count> 0):
        return frame_count

def postprocess(frame, outs, option):
    frameHeight = frame.shape[0]
    frameWidth = frame.shape[1]
    global frame_count_out
    frame_count_out=0
    classIds = []
    confidences = []
    boxes = []
    classIds = []
    confidences = []
    boxes = []
    cc = 0
    for out in outs:
        for detection in out:
            scores = detection[5:]
            classId = np.argmax(scores)
            confidence = scores[classId]
            if confidence > confThreshold:
```

```
                center_x = int(detection[0] * frameWidth)
                center_y = int(detection[1] * frameHeight)
                width = int(detection[2] * frameWidth)
                height = int(detection[3] * frameHeight)
                left = int(center_x - width / 2)
                top = int(center_y - height / 2)
                classIds.append(classId)
                #print(classIds)
                confidences.append(float(confidence))
                boxes.append([left, top, width, height])

    indices = cv.dnn.NMSBoxes(boxes, confidences, confThreshold, nmsThreshold)
    count_person=0 # for counting the classes in this loop.
    for i in indices:
        i = i[0]
        box = boxes[i]
        left = box[0]
        top = box[1]
        width = box[2]
        height = box[3]
        frame_count_out = drawPred(classIds[i], confidences[i], left, top, left + width, top +
height,frame,option)
        my_class='Helmet'
        unknown_class = classes[classId]
        print("===="+str(unknown_class))
        if my_class == unknown_class:
            count_person += 1

    print(str(frame_count_out))
    if count_person == 0 and option == 1:
        cv.putText(frame, "Helmet Not detected", (10, 50), cv.FONT_HERSHEY_SIMPLEX, 0.75,
(0,255,0), 2)
    if count_person >= 1 and option == 0:
        #path = 'test_out/'
        #cv.imwrite(str(path)+str(cc)+".jpg", frame)     # writing to folder.
        #cc = cc + 1
        frame = cv.resize(frame,(500,500))
        cv.imshow('img',frame)
        cv.waitKey(50)


def detectHelmet():
    textarea.delete('1.0', END)
    if option == 1:
        frame = cv.imread(filename)
        frame_count =0
        blob = cv.dnn.blobFromImage(frame, 1/255, (inpWidth, inpHeight), [0,0,0], 1, crop=False)
        net.setInput(blob)
        outs = net.forward(getOutputsNames(net))
        postprocess(frame, outs,0)
        t, _ = net.getPerfProfile()
        label = 'Inference time: %.2f ms' % (t * 1000.0 / cv.getTickFrequency())
        print(label)
```

```
        cv.putText(frame, label, (0, 15), cv.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255))
        print(label)
    else:
        messagebox.showinfo("Person & Motor bike not detected in uploaded image", "Person &
Motor bike not detected in uploaded image")


def videoHelmetDetect():
    global filename
    videofile = askopenfilename(initialdir = "videos")
    video = cv.VideoCapture(videofile)
    while(True):
        ret, frame = video.read()
        if ret == True:
            frame_count = 0
            filename = "temp.png"
            cv.imwrite("temp.png",frame)
            blob = cv.dnn.blobFromImage(frame, 1/255, (inpWidth, inpHeight), [0,0,0], 1,
crop=False)
            net.setInput(blob)
            outs = net.forward(getOutputsNames(net))
            postprocess(frame, outs,1)
            t, _ = net.getPerfProfile()
            #label=''
            #cv.putText(frame, label, (0, 15), cv.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255))
            cv.imshow("Predicted Result", frame)
            if cv.waitKey(5) & 0xFF == ord('q'):
                break
        else:
            break
    video.release()
    cv.destroyAllWindows()


def exit():
    global main
    main.destroy()

font = ('times', 16, 'bold')
title = Label(main, text='Number Plate Detection without Helmet', justify=LEFT)
title.config(bg='lavender blush', fg='DarkOrchid1')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=100,y=5)
title.pack()

font1 = ('times', 14, 'bold')
model = Button(main, text="Upload Image", command=upload)
model.place(x=200,y=100)
model.config(font=font1)

uploadimage = Button(main, text="Detect Motor Bike & Person", command=detectBike)
uploadimage.place(x=200,y=150)
```

```
uploadimage.config(font=font1)

classifyimage = Button(main, text="Detect Helmet", command=detectHelmet)
classifyimage.place(x=200,y=200)
classifyimage.config(font=font1)

exitapp = Button(main, text="Exit", command=exit)
exitapp.place(x=200,y=250)
exitapp.config(font=font1)

font1 = ('times', 12, 'bold')
textarea=Text(main,height=15,width=60)
scroll=Scrollbar(textarea)
textarea.configure(yscrollcommand=scroll.set)
textarea.place(x=10,y=300)
textarea.config(font=font1)

loadLibraries()

main.config(bg='light coral')
main.mainloop()
```

CHAPTER 4
RESULTS AND DISCUSSION

CHAPTER 4
RESULTS AND DISCUSSION

4.1 COMPARISON OF EXISTING SOLUTIONS

"The cameras record traffic violations, and click photos every time a vehicle breaks rules and then it sends the data – vehicle number and type of violation – to the control center. Along with the photo, these cameras also capture a short video as proof. Date, time and camera number is encrypted in the image itself so even authorities cannot tamper with it. The encrypted image goes from the local server to the central server (police control room). Then the data is sent VAHAN (the national vehicle registry database of the Ministry of Road Transport and Highways). Then a message is sent to the phone number which is registered with the vehicle."

We gone through some base papers so, the existing system monitors the traffic violations primarily through CCTV recordings, where the traffic police have to look into the frame where the traffic violation is happening, zoom into the license plate in case rider is not wearing helmet. But this requires lot of manpower and time as the traffic violations frequently and the number of people using motorcycles is increasing day-by-day. In existing system they're not used any algorithms because it uses only manual work. In our proposed system which detects the motorcycle rider with helmet and allows him to fuel filling at pumps of petroleum bunk. Here we allowing the rider for fuelling after detection of helmet, if helmet detected successfully then the system process the instructions to the Arduino then the gates raised and system allows the biker for fuelling. And we are using the tensor flow (is an open source frame work to run the machine learning and deep learning), CNN (is used to detect the helmet of the rider), openCV (is used to process images).

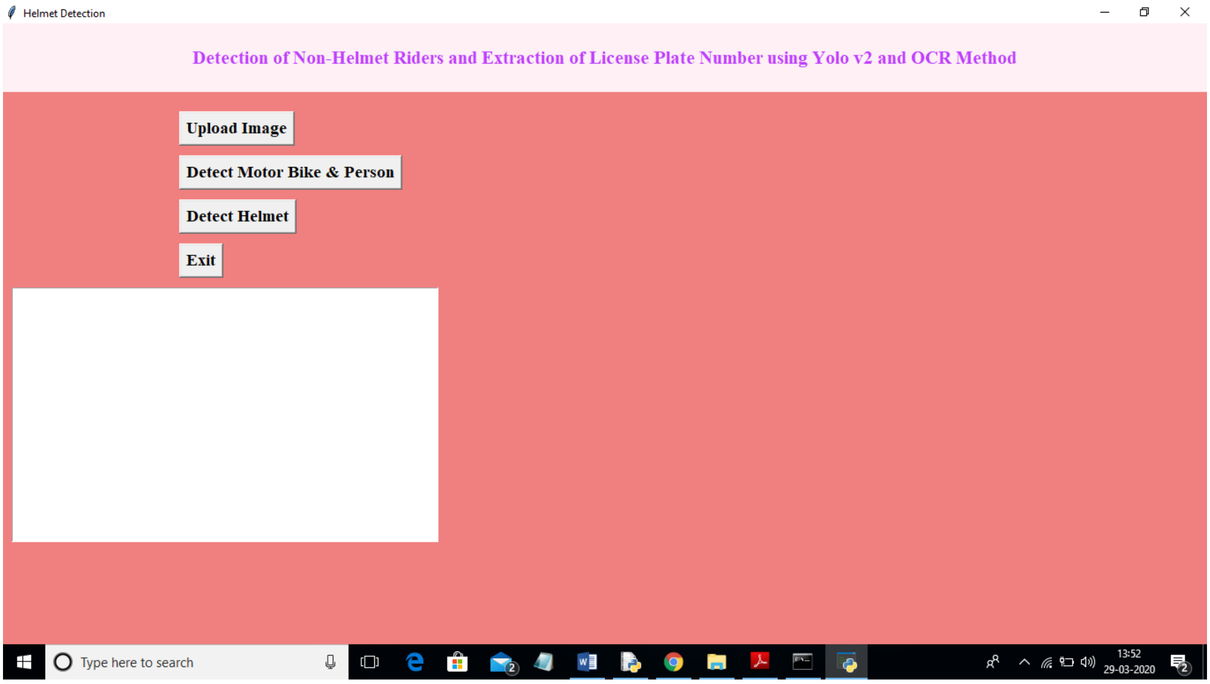
4.2 DATA COLLECTION

Fig 8a: *Rider Without Helmet* fig 8b: *Rider with Helmet*

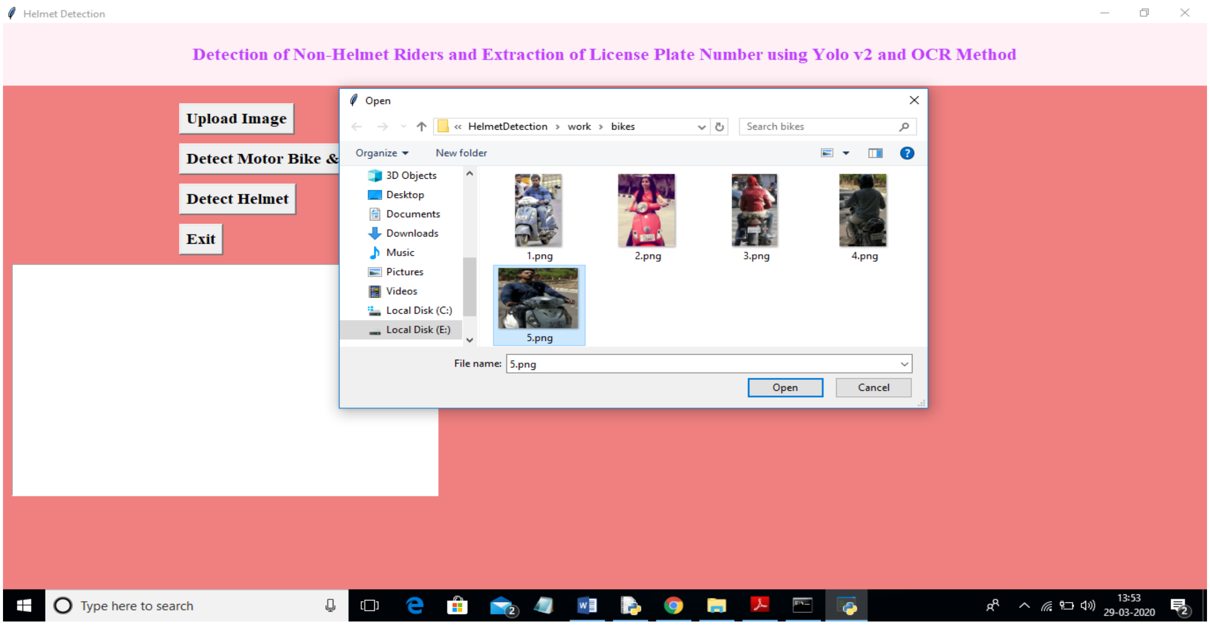
- Utilizing more of man power
- Manual work
- Less alert ness
- No on spot punishment

4.3 RESULTS SCREENSHOTS

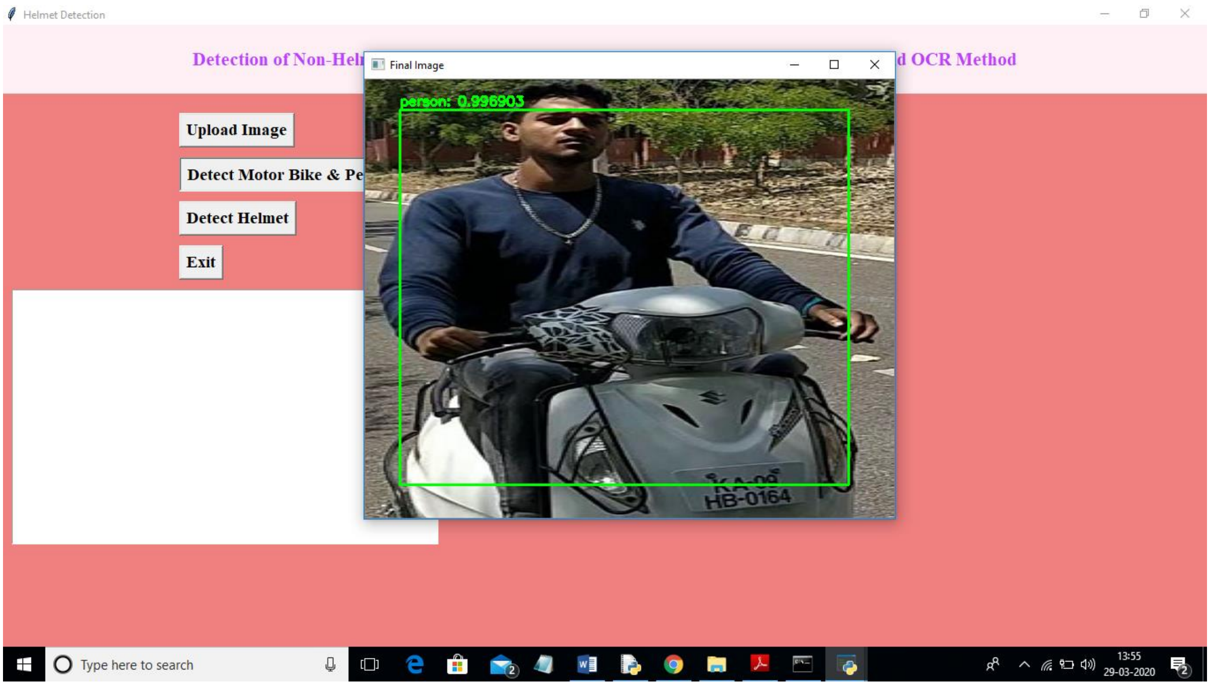
After setting path double click on ‘run.bat’ file to run project and to get below screen



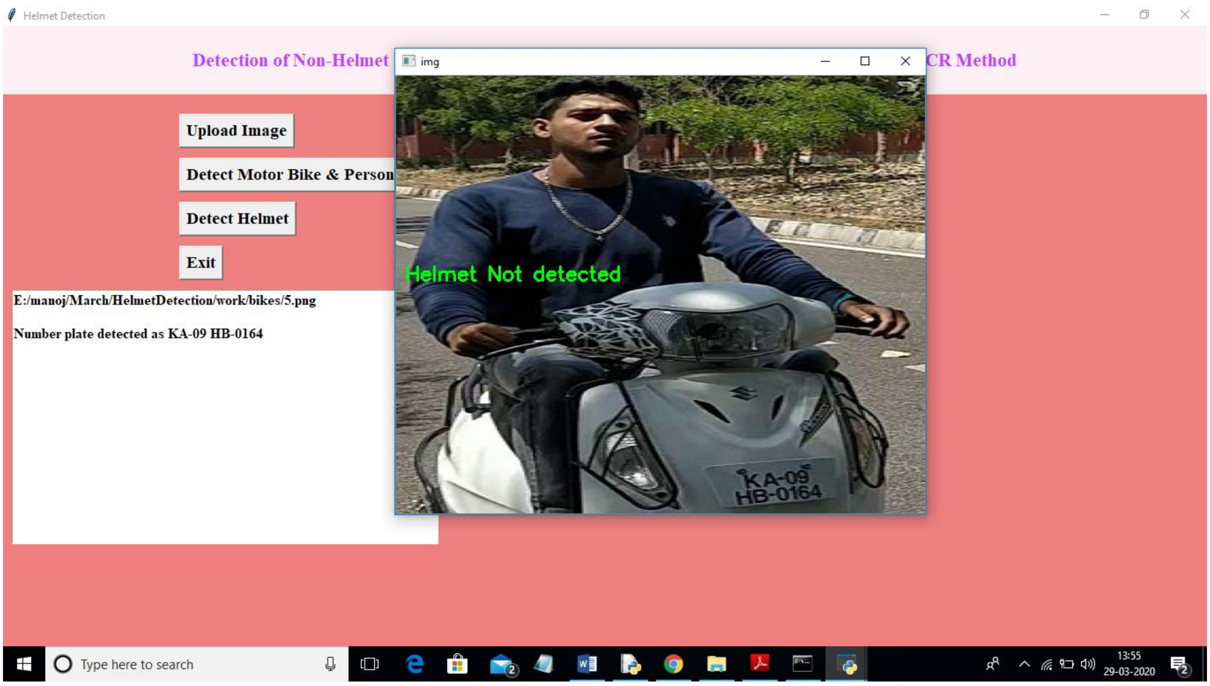
In above screen click on ‘Upload Image’ button and upload image



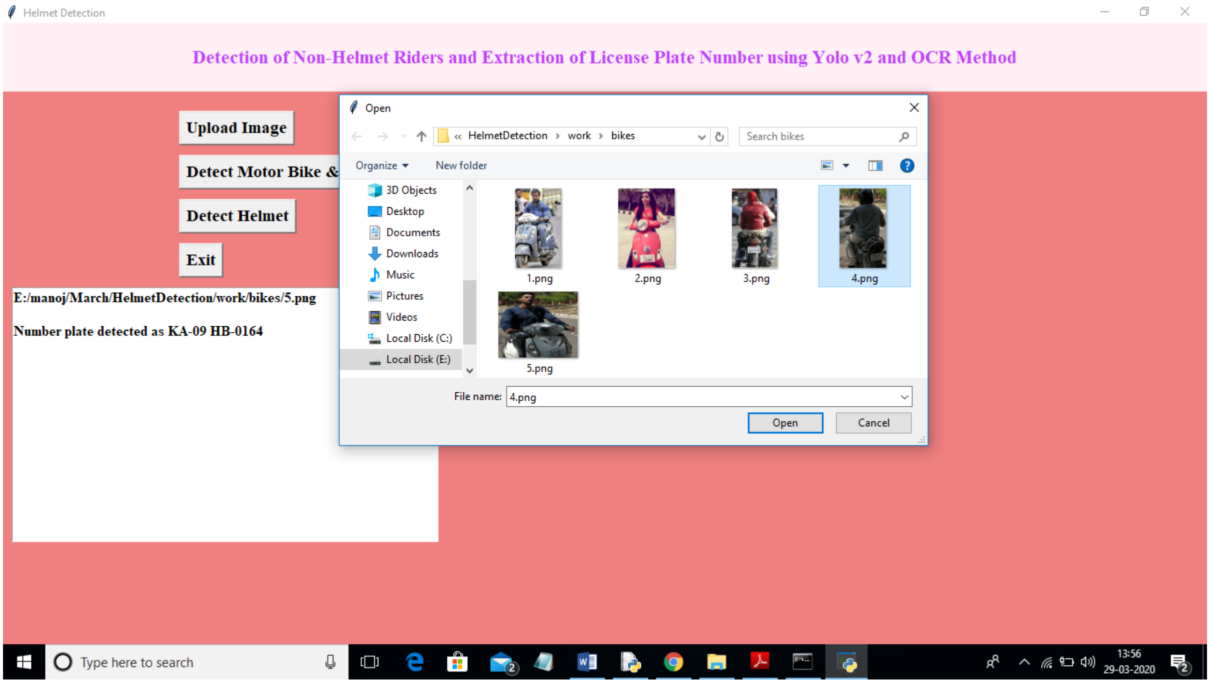
In above screen I selected one image as ‘5.png’ and click on ‘Open’ button to load image. Now click on ‘Detect Motor Bike & Person’ button to detect whether image contains person with motor bike or not



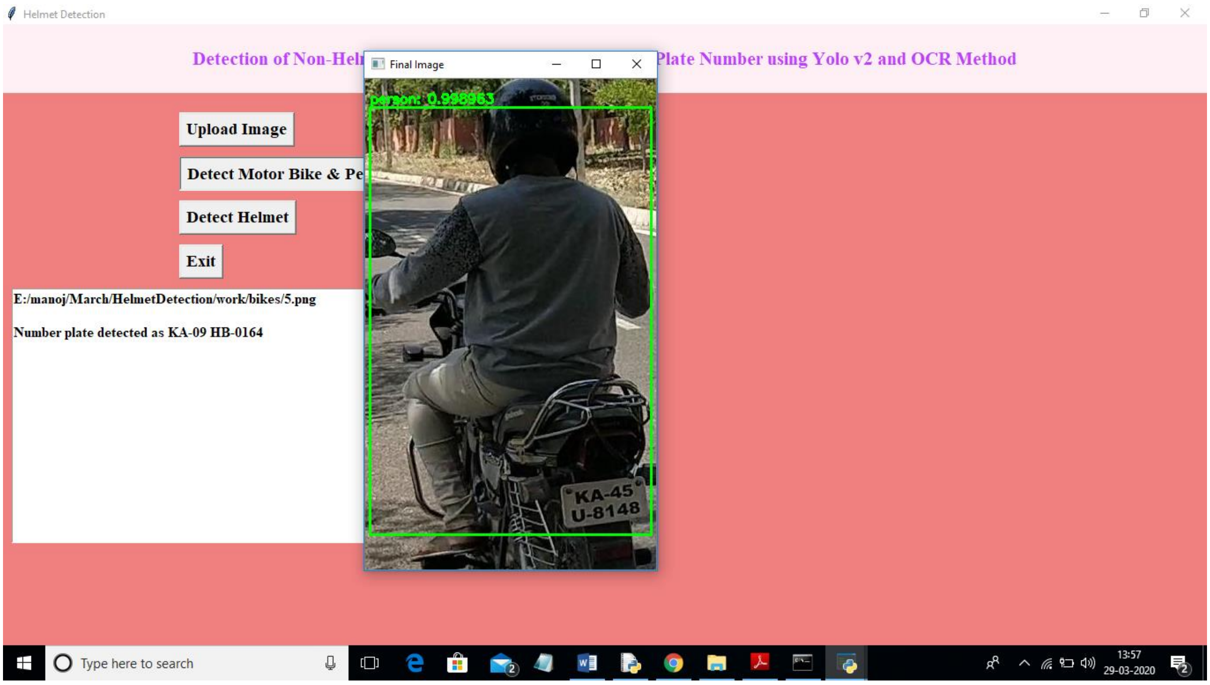
In above screen yolo detected image contains person and bike and now click on ‘Detect Helmet’ button to detect whether he is wearing helmet or not



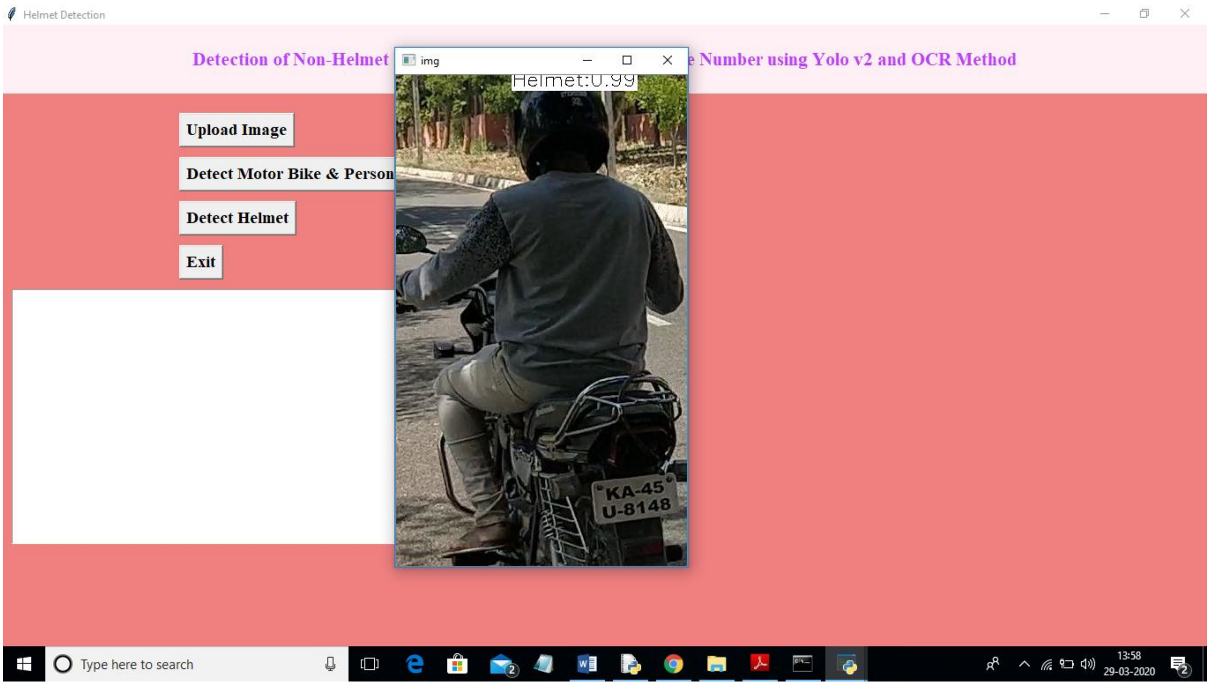
In above screen application detected that person is not wearing helmet and its extracted number from vehicle and display in beside text area. Now we will check with helmet image



In above screen I am uploading 4.png which is wearing helmet and now click on ‘Detect Motor Bike & Person’ button to get below result



In above screen yolo detected person with motor bike and now click on ‘Detect Helmet’ button to get below result



In above screen application detected person is wearing helmet and that label is displaying around his head and application stop there itself and not scanning number plate.

Note: To implement this project and to extract number plate we have trained few images and if u want to extract for new images, then send those new images to us, so we include those images in yolo model to extract new images number plate also.

SYSTEM TEST

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

TEST CASES:

S.NO	Test Case	Excepted Result	Result
1.	click on ‘Upload Image’ button to upload image	selecting and uploading ‘6.jpg’ file and then click on ‘Open’ button to load image and then click on ‘Detect Motor Bike & Person’ button to detect whether image contains person with bike or not	Pass
2.	Upload image	if person with bike detected then it put bounding box and then click on ‘Detect Helmet’ button to get below output	Pass
3	Helmet not detected	we can see helmet not detected and then application identify number plate and display on the text area	Pass

		as ‘AP13Q 8815’. Now try with other image by uploading it	
4	Helmet detected	selecting and uploading ‘h3.jpg’ file and then click on ‘Open’ button then click on ‘Detect Motor Bike & Person’ button to get below result	pass

CHAPTER 5
CONCLUSION

CHAPTER 5
CONCLUSION

The proposed system aims to prevent motorcycle accidents and also reduce the fatality of accidents. Machine learning algorithm and IOT based model has high accuracy and automatically detects the helmet and immediately it gives response like lifting the gates for fuelling.

This system makes a habit to wear a helmet among motorcycle riders. The helmet would make a motorcycle journey more protected and safer.

So finally, the rider helmet detection system for fuel filling project is an effective solution for ensuring the safety of both the rider and the fuel station personnel. By using computer vision technology, the system can detect whether the rider is wearing a helmet or not, and prevent the fuel dispensing process if the rider is not wearing one. This helps to minimize the risk of accidents and injuries caused by helmetless riding. Overall, the implementation of this system can greatly enhance the safety and security of fuel stations and their customers.

CHAPTER 6
REFERENCES

CHAPTER 6

REFERENCES

[1]. J. Zarif, “16 deaths every hour : Indian roads claim the maximum number of lives in 2014,” 2020. [Online]. Available : http://timesofindia.indiatimes.com/india/16 -deaths -every -hourIndian - roads -claim -the -maximum -number -of-lives -in 2014/articleshow/48128946.cms . [Accessed: January, 2020 }
[2]. N. Divyasudha and P. Arulmozhivarman, “Analysis of Smart helmets and Designing an IoT based smart helmet : A cost -effective solution for Riders,” in 1st International Conference on Innovations in Information and Communication Technology (ICIICT) , 2019.
[3]. How traffic cameras issue e-challans [Apr 17, 2021] https://timesofindia.indiatimes.com/city/delhi/how-traffic-cameras-issue-e-challans/articleshow/82103731.cms
[4]. How traffic cam works and issue challan https://www.financialexpress.com/auto/industry/how-traffic-cameras-work-and-issue-challans-violation-tracking-fining-explained-delhi-mumbai-fines/2200053/
[5]. HELMET data set https://paperswithcode.com/dataset/helmet
[6]. "Helmet Detection System for Safe Bike Riding Using Deep Learning Techniques," by M. V. Madhav, S. S. S. S. S. S. Prasad, V. H. Rao, and P. C. Krishna, IEEE Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), 2016.
[7]. "A Deep Learning Approach for Helmet Detection in Motorcycles," by N. L. Romero-Tapia, E. M. Andrade-González, G. García-Sánchez, J. J. Mendoza-Fuentes, and J. R. de la Cruz-Avalos, IEEE Access, vol. 9, pp. 11183-11195, 2021.
[8]. "Rider Helmet Detection System using Image Processing Techniques," by S. S. S. S. S. S. Prasad, M. V. Madhav, V. H. Rao, and P. C. Krishna, IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2015.
[9]. "A Computer Vision System for Motorcycle Rider Helmet Detection and Face Recognition," by F. M. Hernández-Álvarez, M. A. Cruz-Ramos, J. J. Castillo-Secundino, and M. E. Oropeza-Rodríguez, Sensors, vol. 20, no. 7, pp. 1995, 2020.

[10]. GITHUB LINK: