

MinePlate: Innovating Vehicle Registration Plate Recognition in Mining Operations

Submitted for partial fulfillment of the requirements

for the award of

BACHELOR OF TECHNOLOGY

in

**COMPUTER SCIENCE ENGINEERING -
ARTIFICIAL INTELLIGENCE & MACHINE LEARNING**

by

PENUMALLI JYOTENDRANADH	- 20BQ1A4243
VINNAKOTA CHANDANA SRI	- 20BQ1A4263
MUTHAYALAPATI SAMPATH KUMAR	- 20BQ1A4238
NARRA SAI CHANDRA	- 20BQ1A4240

Under the guidance of

Mr. K. BALAKRISHNAB.Tech., MBA., M.Tech., DID., (PhD)

ASSISTANT PROFESSOR



**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING -
ARTIFICIAL INTELLIGENCE & MACHINE LEARNING**

(B. Tech Program is Accredited by NBA)

VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

Permanently Affiliated to JNTU Kakinada, Approved by AICTE

Accredited by NAAC with 'A' Grade, ISO 9001:2008 Certified

NAMBUR (V), PEDAKAKANI (M), GUNTUR – 522 508

Tel no: 0863-2118036, url: www.vvitguntur.com

(April 2024)

DECLARATION

We, Mr. P. Jyotendranadh, Ms. V. Chandana Sri, Mr. M. Sampath Kumar, Mr. N. Sai Chandra, hereby declare that the Project Report entitled “**MinePlate: Innovating Vehicle Registration Plate Recognition in Mining Operations**” done by us under the guidance of our guide Mr. K. Balakrishna, Assistant Professor, Computer Science & Engineering - Artificial Intelligence & Machine Learning at Vasireddy Venkatadri Institute of Technology is submitted for partial fulfillment of the requirements for the award of Bachelor of Technology in Computer Science Engineering - Artificial Intelligence & Machine Learning. The results embodied in this report have not been submitted to any other University for the award of any degree.

DATE :

PLACE :

SIGNATURE OF THE CANDIDATE (S)

ACKNOWLEDGEMENT

We take this opportunity to express my deepest gratitude and appreciation to all those people who made this project work easier with words of encouragement, motivation, discipline, and faith by offering different places to look to expand my ideas and helped me towards the successful completion of this project work.

First and foremost, we express my deep gratitude to **Mr. Vasireddy Vidya Sagar**, Chairman, Vasireddy Venkatadri Institute of Technology for providing necessary facilities throughout the B.Tech programme.

We express my sincere thanks to **Dr. Y. Mallikarjuna Reddy**, Principal, Vasireddy Venkatadri Institute of Technology for his constant support and cooperation throughout the B.Tech programme.

We express my sincere gratitude to **Dr. K. Suresh Babu**, Professor & HOD, Computer Science & Engineering - Artificial Intelligence & Machine Learning, Vasireddy Venkatadri Institute of Technology for his constant encouragement, motivation and faith by offering different places to look to expand my ideas.

We would like to express my sincere gratefulness to our Guide **Mr. K. Balakrishna**, Assistant Professor, Computer Science & Engineering - Artificial Intelligence & Machine Learning for his insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project.

We would like to express our sincere heartfelt thanks to our Project Coordinator **Mr. N. Balayesu**, Assistant Professor, Computer Science & Engineering - Artificial Intelligence & Machine Learning for his valuable advices, motivating suggestions, moral support, help and coordination among us in successful completion of this project.

We would like to take this opportunity to express my thanks to the **Teaching and Non-Teaching** Staff in the Department of Computer Science & Engineering - Artificial Intelligence & Machine Learning, VVIT for their invaluable help and support.

Name (s) of Students

Penumalli Jyotendranadh

Vinnakota Chandana Sri

Muthayalapati Sampath Kumar

Narra Sai Chandra



VASIREDDY VENKATADRI INSTITUTE OF TECHNOLOGY

Permanently Affiliated to JNTUK, Kakinada, Approved by AICTE

Accredited by NAAC with 'A' Grade, ISO 9001:20008 Certified

Nambur, Pedakakani (M), Guntur (Gt) -522508

**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING -
ARTIFICIAL INTELLIGENCE & MACHINE LEARNING**

CERTIFICATE

This is to certify that this **Project Report** is the Bonafide work of **Mr. P. Jyotendranadh, Ms. V. Chandana Sri, Mr. M. Sampath Kumar, Mr. N. Sai Chandra**, bearing Reg. No. **20BQ1A4243, 20BQ1A4263, 20BQ1A4238, 20BQ1A4240** respectively who had carried out the project entitled **"MinePlate: Innovating Vehicle Registration Plate Recognition in Mining Operations"** under our supervision.

Project Guide

(Mr. K. Balakrishna, Assistant Professor)

Head of the Department

(Dr. K. Suresh Babu, Professor)

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

TABLE OF CONTENTS

Ch. No.	Title	Page No.
	Contents	i
	List of Figures	iii
	List of Tables	v
	Nomenclature	vi
	Abstract	vii
1	INTRODUCTION	
	1.1 What is ANPR?	1
	1.2 What is OCR?	2
	1.2.1 Techniques of OCR	3
	1.2.2 Uses of OCR	4
	1.3 Market Value of ANPR System	4
	1.4 What is YOLOv5	5
	1.4.1 YOLOv5 Architecture	6
2	REVIEW OF LITERATURE	8
3	PROPOSED SOLUTION	
	3.1 Overview	11
	3.2 Process	11
	3.3 Vehicle Recognition	12
	3.4 License Plate Recognition	13
	3.4.1 Preparing the Dataset	13
	3.4.2 Train the Model	13
	3.5 License Plate Image Enhancement	14
	3.6 License Plate Text Extraction	15
4	SYSTEM REQUIREMENT SPECIFICATION	
	4.1 Introduction	16
	4.2 Functional and Non - Functional Requirements	16
	4.3 Hardware Requirements	18

	4.4 Software Requirements	18
5	SYSTEM DESIGN	
	5.1 Introduction of Input Design	19
	5.2 UML Diagrams	19
	5.2.1 Use Case Diagram	20
	5.2.2 Class Diagram	21
	5.2.3 Activity Diagram	22
	5.2.4 Data Flow Diagram	24
	5.2.5 Sequence Diagram	25
	5.2.6 Component Diagram	27
	5.2.7 Deployment Diagram	28
6	IMPLEMENTATION	
	6.1 Modules	29
	6.1.1 Data Pre-Processing	29
	6.1.2 Number Plate Extraction	29
	6.1.3 Character Segmentation	29
	6.1.4 Optical Character Recognition	30
	6.2 Source Code	30
	6.2.1 Implementation of Scanning the Image of Vehicle	30
7	RESULTS	34
8	CONCLUSION AND FUTURE SCOPE	44
9	REFERENCES	45
	Certificate	48
	Journal Published Paper	49

LIST OF FIGURES

Fig. No.	Figure Name	Page No.
1.2	Optical Character Recognition (OCR)	2
1.4	Object Detector Model Using YOLO	6
1.4.1.1	Anatomy of an Object Detector	7
1.4.1.2	Object Detection Process	7
3.2	Flowchart of Proposed System	11
3.3	YOLOv5 Architecture	12
3.4.2	YOLOv5 Model Training	13
3.5.1	Grayscale Image	14
3.5.2	Grayscale Conversion	14
3.6	EasyOCR Framework	15
5.2.1	Use Case Diagram	20
5.2.2	Class Diagram	21
5.2.3.1	Activity Diagram For Admin	22
5.2.3.2	Activity Diagram For Mining Company	23
5.2.3.3	Activity Diagram For Security Personnel	23
5.2.4	Data Flow Diagram	24
5.2.5.1	Sequence Diagram For Admin	25
5.2.5.2	Sequence Diagram For Mining Company	26
5.2.5.3	Sequence Diagram For Security Personnel	26
5.2.6	Component Diagram	27
5.2.7	Deployment Diagram	28
7.1	Upload the Vehicle Image	35
7.2	Result of the Uploaded Vehicle Image	35
7.3	Logging All the Vehicle Activities	36
7.4	Home Page	36
7.5	Mining Company Login Page	37
7.6	Security Personnel Login Page	37
7.7	Admin Login Page	38
7.8	Contact Us Page	38
7.9	Security Personnel Dashboard	39
7.10	Admin Dashboard	39

7.11	Adding the Mining Company	40
7.12	Managing the Mining Companies	40
7.13	Adding Security Personnel Details	41
7.14	Managing Security Personnel	41
7.15	Adding Vehicle Admin Database	42
7.16	Managing the vehicles	42
7.17	Mining Company Dashboard	43
7.18	Vehicle Activities in Mining Companies	43

LIST OF TABLES

Table No.	Table Name	Page No.
Table 2.1	Neural Networks Accuracy	8

NOMENCLATURE

ANPR	Automatic Number Plate Recognition
ROI	Region of Interest
OCR	Optical Character Recognition
YOLO	You Only Look Once
ML	Machine Learning
CNN	Convolution Neural Networks
MSER	Maximum Stable Extremal Regions
UML	Unified Modelling Language

ABSTRACT

Mining operations in remote and harsh environments require high-tech and efficient workflows to ensure uninterrupted delivery cycles and operational success. However, theft and misuse of movable assets, such as vehicles, pose significant challenges and financial losses to the mining industry. To address these issues, we propose an system that combines Machine Learning (ML) and image processing technologies. To achieve accurate registration number extraction, image processing techniques are employed to capture the number plate information from images or video streams of moving vehicles. Optical Character Recognition (OCR) algorithms based on ML are then utilized to convert the extracted characters into text, providing the registration number. Moreover, ML models are trained to classify vehicles based on their characteristics, enabling verification of whether the correct number plate is affixed to the corresponding vehicle, thereby mitigating malicious activities like plate swapping. The captured data, including registration numbers, vehicle locations, and any detected anomalies, is stored in the cloud for further analysis. ML-based analytics are applied to identify suspicious patterns and potential theft or unauthorized activities. Security measures, including authentication checks using ML, are employed to cross-reference number plates with a database of valid registration numbers, further enhancing the system's robustness. By integrating image processing and ML, our system offers superior accuracy in registration number extraction, enhanced vehicle classification, and strengthened security measures.

KEYWORDS: OCR(Optical Character Recognition), YOLOV5(You Only Look Once), EasyOCR, ALPR(Automatic License Plate Recognition), ROI(Region of Interest).

1.1 WHAT IS ANPR?

Automatic Number Plate Recognition or ANPR is a technology that uses pattern recognition to 'read' vehicle number plates. In simple terms ANPR cameras 'photograph' the number plates of the vehicles that pass them. This 'photograph' is then fed in a computer system to find out details about the vehicle itself. ANPR consists of cameras linked to a computer. As a vehicle passes, ANPR 'reads' Vehicle Registration Marks – more commonly known as number plates - from digital images, captured through cameras located either in a mobile unit, in-built in traffic vehicles or via Closed Circuit Television (CCTV). The digital image is converted into data, which is processed through the ANPR system. We proposed a method mainly based on edge detection, OCR operation and Finding Rectangles in a Vehicle Image. Owning a vehicle today is not merely a symbol of luxury but has become a necessity. However, considering vehicles, any catastrophic situation can take place. Therefore there is always an urgent need to arrange appropriate measures to increase the safety, security as well as monitor the vehicles to avoid any mishap. It would help us in the situations such as: Instantaneously obtain vehicle details using image processing. Allowing an agency to detect the location of its vehicles. Automatically notify the user if there are traffic violations registered to the vehicle. One such measure is the use of a vehicle tracking system using the GPS (Global Position System). Such a tracking system includes a mechanized device that is equipped in a vehicle. Using software present at an operational base, it helps track the location of the vehicle. This base station is used for monitoring purposes. It is accompanied by maps such as Google maps, Here maps, Bing maps etc for the representation of the location.

ANPR can be used to store the images captured by the cameras as well as the text from the license plate, with some configurable to store a photograph of the driver. Systems commonly use infrared lighting to allow the camera to take the picture at any time of the day. A powerful flash is included in at least one version of the intersection monitoring cameras, serving both to illuminate the picture and to make the offender aware of his or her mistake. ANPR technology tends to be region specific, owing to plate variation from place to place. The acquisition of digital image usually suffers from undesirable camera shakes and due to unstable random camera motions. Hence image enhancement algorithms are required to remove these unwanted camera shakes. . Python is used as the main programming language. We extract the information and save the data in JSON format for further processing and analysis.

1.2 WHAT IS OCR?

Optical character recognition or optical character reader (OCR) is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo). Widely used as a form of data entry from printed paper data records – whether passport documents, invoices, bank statements, computerized receipts, business cards, mail, printouts of static-data, or any suitable documentation – it is a common method of digitizing printed texts so that they can be electronically edited, searched, stored more compactly, displayed on-line, and used in machine processes such as cognitive computing, machine translation, (extracted) text-to-speech, key data and text mining. OCR is a field of research in pattern recognition, artificial intelligence and computer vision.



Fig. 1.2. Optical Character Recognition

Early versions needed to be trained with images of each character, and worked on one font at a time. Advanced systems capable of producing a high degree of recognition accuracy for most fonts are now common, and with support for a variety of digital image file format inputs. Some systems are capable of reproducing formatted output that closely approximates the original page including images, columns, and other non-textual components.

1.2.1 Techniques of OCR

- **Pre-Processing**

OCR software often "pre-processes" images to improve the chances of successful recognition. Techniques include:

- De-skew – If the image was not aligned properly when scanned, it may need to be tilted a few degrees clockwise or counterclockwise in order to make lines of text perfectly horizontal or vertical.
- Despeckle – remove positive and negative spots, smoothing edges
- Line removal – Cleans up non-glyph boxes and lines
- Layout analysis or "zoning" – Identifies columns, paragraphs, captions, etc. as blocks. Especially important in multi-column layouts and tables.

- **Text Recognition**

Matrix matching involves comparing an image to a stored glyph on a pixel-by-pixel basis; it is also known as "pattern matching", "pattern recognition", or "image correlation". This relies on the input glyph being correctly isolated from the rest of the image, and on the stored glyph being in a similar font and at the same scale. This technique works best with typewritten text and does not work well when new fonts are encountered. This is the technique the early physical photocell-based OCR implemented, rather directly.

- **Post – Processing**

OCR accuracy can be increased if the output is constrained by a lexicon – a list of words that are allowed to occur in a document. This might be, for example, all the words in the English language, or a more technical lexicon for a specific field. This technique can be problematic if the document contains words not in the lexicon, like proper nouns. Tesseract uses its dictionary to influence the character segmentation step, for improved accuracy. The output stream may be a plain text stream or file of characters, but more sophisticated OCR systems can preserve the original layout of the page and produce, for example, an annotated PDF that includes both the original image of the page and a searchable textual representation.

1.2.2 Uses of OCR

- Data entry for business documents, e.g. check, passport, invoice, bank statement and Receipt.
- Automatic number plate recognition
- In airports, for passport recognition and information extraction.
- Automatic insurance documents key information extraction.
- Traffic sign recognition
- Extracting business card information into a contact list.
- More quickly make textual versions of printed documents, e.g. book scanning for Project Gutenberg.
- Make electronic images of printed documents searchable, e.g. Google Books.
- Converting handwriting in real time to control a computer (pen computing).
- Defeating CAPTCHA anti-bot systems, though these are specifically designed to prevent OCR. The purpose can also be to test the robustness of CAPTCHA anti-bot systems.
- Assistive technology for blind and visually impaired users.
- Writing the instructions for vehicles by identifying CAD images in a database that are appropriate to the vehicle design as it changes in real time.
- Making scanned documents searchable by converting them to searchable PDFs.

1.3 MARKET VALUE OF ANPR SYSTEM

According to the new market research report "Automatic Number Plate Recognition (ANPR) System Market by Type (Fixed, Mobile, Portable), Component (ANPR Cameras, Software, Frame Grabbers, Triggers), Application (Traffic Management, Law Enforcement, Electronic Toll Collection, Parking Management), and Geography - Global Forecast to 2023", the automatic number plate recognition (ANPR) system market in 2016 was valued at USD 1.78 Billion and is expected to reach USD 3.57 Billion by 2023, at a CAGR of 9.74% between 2017 and 2023. Factors that are driving this market include the infrastructure growth in emerging economies, increasing allocation of funds by various governments on intelligent transport systems (ITS), deployment of camera technologies in security and surveillance, and traffic enforcement application, and the growing usage of video analytics technology for intelligent monitoring of vehicles. Europe was the largest market for ANPR systems in 2016. The large

market in this region can be attributed to the high adoption of intelligent transportation systems for traffic management, tolling management, law/police enforcement, and other applications. The market in Europe has been segmented into Germany, the UK, France and the Rest of Europe. Some major companies offering ANPR systems in Europe include ARH Inc. (Hungary), Digital Recognition Systems Ltd. (UK), NDI Recognition Systems Ltd. (UK), and Q-Free ASA (Norway).

The key players in the market include Kapsch TrafficCom AG (Austria), Conduent, Inc. (US), QFree ASA (Norway), Siemens AG (Germany), Genetec Inc. (Canada) Neology, Inc. (US), Bosch Security Systems GmbH (Germany), Tattile srl (Italy), TagMaster North America, Inc. (US), NDI Recognition Systems Ltd. (UK), Euro Car Parks Limited (UK), Quercus Technologies, S.L. (Spain) Vigilant Solutions, Inc. (US), Elsag North America, LLC (US), ARH Inc. (Hungary), Digital Recognition System Ltd. (UK), Beltech BV (Netherlands), ANPR International Ltd. (UK), HTS (New York), FF Group (Cyprus), and so on. This report categorizes the global ANPR system market on the basis of type, component, application, and geography. The report describes the drivers, restraints, opportunities, and challenges for the growth of this market.

1.4 WHAT IS YOLOv5?

YOLOv5 is a model in the You Only Look Once (YOLO) family of computer vision models. YOLOv5 is commonly used for detecting objects. YOLOv5 comes in four main versions: small (s), medium (m), large (l), and extra large (x), each offering progressively higher accuracy rates. Each variant also takes a different amount of time to train.

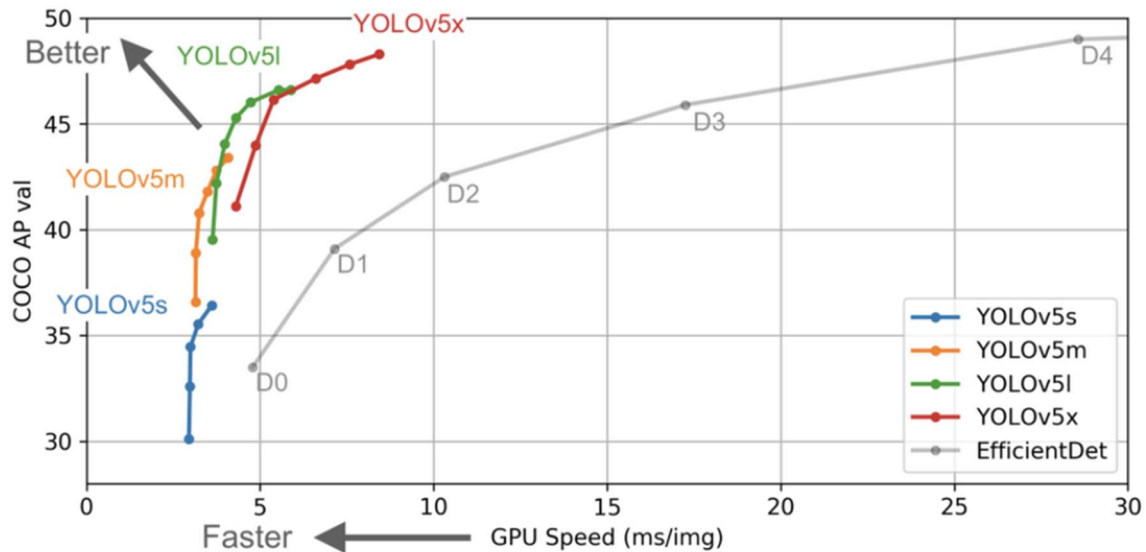


Fig. 1.4 Object Detector Model using YOLO

In the chart, the goal is to produce an object detector model that is very performant (Y-axis) relative to its inference time (X-axis). Preliminary results show that YOLOv5 does exceedingly well to this end relative to other state of the art techniques. In the chart above, you can see that all variants of YOLOv5 train faster than EfficientDet. The most accurate YOLOv5 model, YOLOv5x, can process images multiple times faster with a similar degree of accuracy than the EfficientDet D4 model. This data is discussed in more depth later in the post. YOLOv5 derives most of its performance improvement from PyTorch training procedures, while the model architecture remains close to YOLOv4.

1.4.1 YOLOv5 Architecture

Object detection, a use case for which YOLOv5 is designed, involves creating features from input images. These features are then fed through a prediction system to draw boxes around objects and predict their classes.

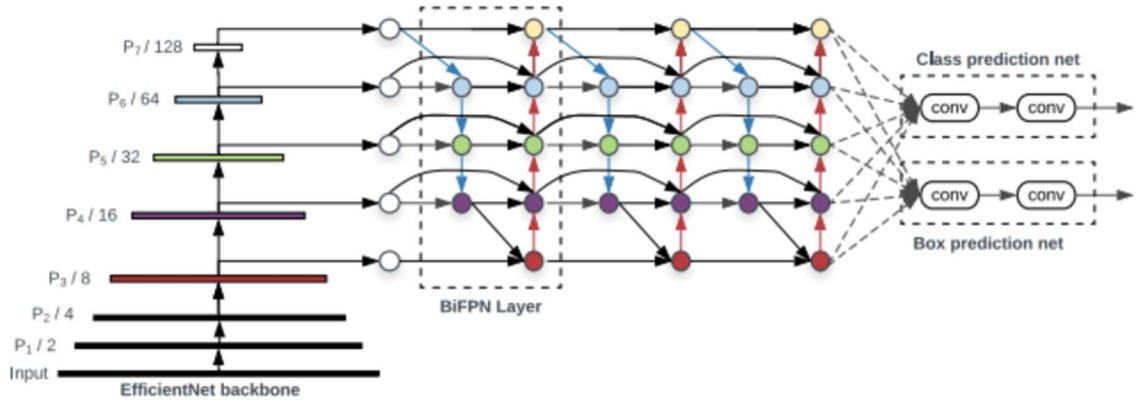


Fig. 1.4.1.1 Anatomy of an Object Detector

The YOLO model was the first object detector to connect the procedure of predicting bounding boxes with class labels in an end to end differentiable network.

The YOLO network consists of three main pieces.

1. **Backbone:** A convolutional neural network that aggregates and forms image features at different granularities.
2. **Neck:** A series of layers to mix and combine image features to pass them forward to prediction.
3. **Head:** Consumes features from the neck and takes box and class prediction steps.

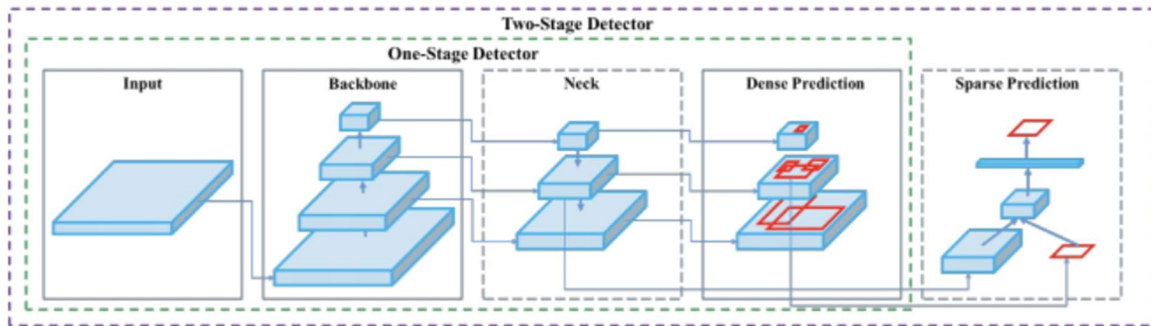


Fig. 1.4.1.2 Object Detection Process

“A Comparative Study of Four Deep Neural Networks for Automatic License Number Plate Recognition System, Authors: Ganga Krishnan. G and Natheera Beevi M” proposed that the system evaluates the performance of four deep neural networks. Finding the best object detection method is the main objective of this study. i.e., a best object detection algorithm can accurately find out the license plate region so that the license plate number can be extracted successfully. The system automatically detects the license plate using four deep learning algorithms. The input image to the system is a car image and detects the number plate using the deep neural networks CNN, VGG16, VGG19, and YOLOV3 separately. Finally, evaluate the performance of the four deep neural networks in terms of accuracy to find out the best algorithm for license plate detection. The VGG16 achieves the highest performance while evaluating the performance of each algorithm using the test set. VGG19 exhibits the least accuracy as well. The VGG16 is the final model employed for number plate recognition. CNN shows an accuracy of 77%, VGG16 shows an accuracy of 89%, VGG19 shows an accuracy of 49%, and YOLOV3 shows an accuracy of 78%. Among these, VGG16 exhibits precise license plate recognition. The dataset is collected from the public repository for the study. The pre-processing module receives the dataset before training and prepares the data. For training the model, CNN, VGG16, VGG19, and YOLO V3 algorithms are used. Train these algorithms separately on the pre-processed dataset and perform an evaluation on these models. The evaluation result shows that VGG16 performs well on this dataset with an accuracy of 89.6% i.e., VGG16 outperforms all other algorithms for number plate detection.

Neural Networks	Accuracy
CNN	77%
VGG16	89%
VGG19	49%
YOLOv3	78%

Table 2.1 Neural Networks Accuracy

“Deep Learning-Based System for License Plate Recognition In Somalia, Authors: Ahmed Nur Ali, Dr. Deepak K Sinha and Dr. Garima Sinha” proposed that the system incorporates two object detection networks, each serving a distinct task. The first network is dedicated to plate detection, utilizing the TensorFlow Object Detection API. Specifically, the pre-trained SSD MobileNet model is employed for the accurate identification and localization of license plates within images or video frames. The second network focuses on character recognition within the detected plates and utilizes the Tesseract OCR Engine for this purpose. During the training process, Google Colab was used as the platform. The dataset for car number plates comprises a total of 24,000 images, each accompanied by an XML file in the same directory. Within the XML file, crucial information regarding annotations is provided, including labels and the precise coordinates of the bounding box that encloses the objects of interest within the respective image. This project aims to automate the number plate detection system for enhanced security purposes. The data obtained from this system can be utilized for multiple purposes, including identifying stolen vehicles and capturing over-speeding vehicles. By replacing the current manual system, this project offers a more efficient and effective solution.

“Automatic License Plate Recognition using MSER and CNN, Authors: Abhipsa S Kalyanshetti, K Hemanth Ratna Kumar, Chethan R, Karthick R” a model for detecting license plate number from an image. The license plate consists of at least 4 characters to recognize. The image is first converted into gray-scale and filtered using a bilateral filter. Then, MSER is used to detect the license plate in the image and the plate region is extracted. Plate region is skew corrected if needed and each character is segmented. Individual characters are given as input to a CNN model to predict the character or class. The resulting license plate number is extracted. License plate recognition is the process of locating the region of interest (ROI) that is the license plate in the image. License plate segmentation is cropping out the located license plate from the image. Last step is the recognition of the segmented characters in the license plate. For locating the license plate MSER method is used. MSER is a technique used to detect the extremal regions in the image. Image processing techniques such as Sobel edge filtering and thresholding are applied to segment the region. After skew correction of the detected plate individual alphanumeric characters are segmented and recognized using convolutional neural network (CNN). The extracted numbers are filtered at last by using pattern matching technique. The system is implemented in Python.

“Long distance Automatic Number Plate Recognition under perspective distortion using zonal density and Support Vector Machine, Authors: Noprianto, Sunu Wibirama, Hanung Adi Nugroho” Automatic Number Plate Recognition (ANPR) is one of computer vision applications to extract information in vehicles plate number. Nevertheless, perspective distortion is unavoidable when taking pictures of the plate number. Another factor that causes inaccuracy is the distance of the camera from the plate number. To solve these problems, we propose a new method to automatically detect and recognize vehicle plate number with regards to perspective distortion and distance of capturing plate number. We used zonal density with Support Vector Machine (SVM) as a classifier. We tested our algorithm on 21 vehicles plate number with 1, 3, and 5 meter of capturing distance. Our method yields an accuracy of 89.77%, 82.86%, and 65.22% for 1, 3, and 5 meters capturing distance, respectively. Compared with previous work, our method is able to preserve high accuracy when segmenting characters of plate number taken from 5 meter distance.

“License Plate Detection with Machine Learning Without Using Number Recognition, Authors: Kazuo Ohzeki, Max Geigis, Stefan Alexander Schneider” In autonomous driving, detecting vehicles together with their parts, such as a license plate is important. Many methods with using deep learning detect the license plate based on number recognition. However, there is an idea that the method using deep learning is difficult to use for autonomous driving because of the complexity in realizing deterministic verification. Therefore, development of a method that does not use deep learning(DL) has become important again. Although the authors have made the world's best performance in 2018 for Caltech data with using DL, this concept has now turned to another research without using DL. The CT5L method is the latest type, that includes techniques of the continuity of vertical and horizontal black-and-white pixel values inside the plate, unique Hough transform, only vertical and horizontal lines are detected, the top five in the order of the number of votes to ensure good performance. In this paper, a method to determine the threshold value for binarizing input by machine learning is proposed, and good results are obtained. The detection rate is improved by about 20 points in percent as compared to the fixed case. It achieves the best performance among the conventional fixed threshold method, Otsu's method, and the conventional method of JavaANPR.

3.1 OVERVIEW:

Our proposed solution involves the usage of YOLOv5 and EasyOCR for Automatic License Plate Recognition (ALPR). YOLOv5 is a deep convolutional neural network that is used for vehicle recognition and license plate detection, while EasyOCR is used for character segmentation and recognition. The combination of these two tools forms a sequential pipeline for ALPR.

3.2 PROCESS:

The flow chart representation is as follows:

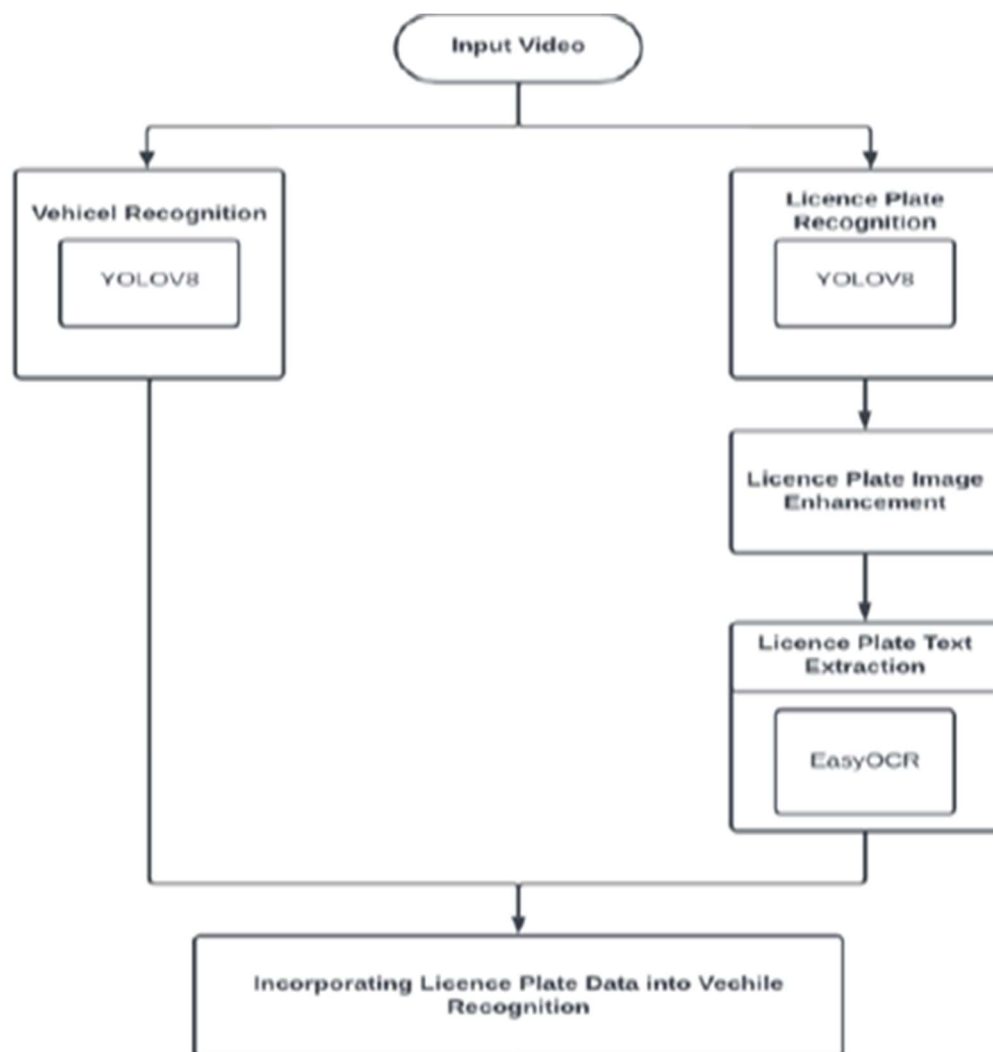


Fig:3.2 Flowchart of Proposed System

3.3 VEHICLE RECOGNITION

Vehicle recognition is a crucial component of modern computer vision systems, with applications ranging from traffic management to surveillance and autonomous vehicles. In this context, the YOLOv5 algorithm plays a pivotal role as a powerful and efficient object detection framework. Trained on the extensive COCO dataset, YOLOv5 exhibits the capability to detect a wide range of objects, including vehicles, in real-time video streams.

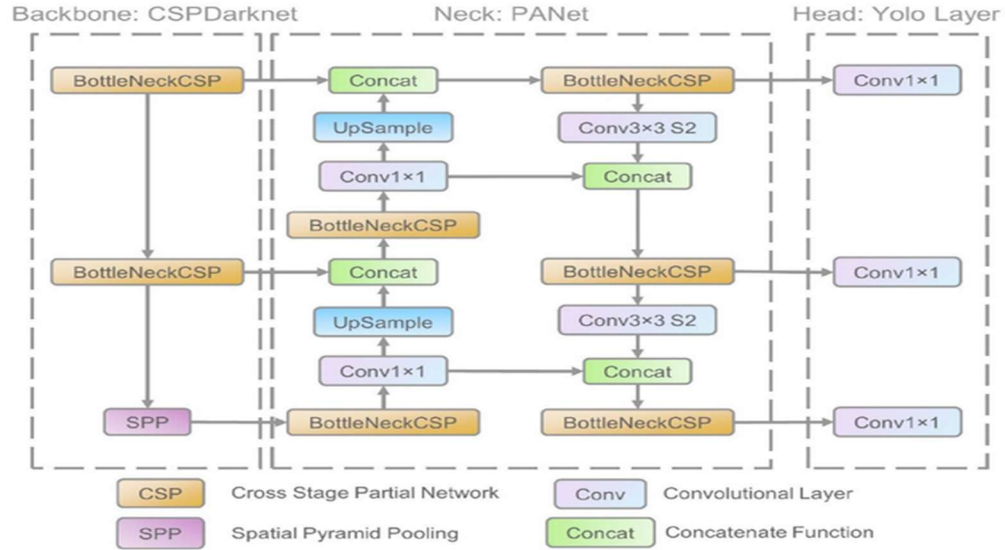


Fig. 3.3 YOLOv5 Architecture

The process of vehicle recognition begins with the input video stream, which is sequentially processed frame by frame. Each frame is analyzed by the YOLOv5 model, which identifies potential vehicles within the image. The algorithm returns bounding boxes around these detected vehicles, accompanied by confidence scores that reflect the model's confidence in its predictions. To ensure that only vehicles are considered for further analysis, the detected bounding boxes are filtered based on their associated class identifiers. Vehicles typically have specific class identifiers, making it possible to distinguish them from other objects that may be present in the scene. The resulting set of filtered bounding boxes, representing vehicles in the frame, forms the basis for subsequent analysis. These bounding boxes are then passed to the license plate recognition system, which focuses on the regions of interest (ROI) containing the license plates of the detected vehicles. This two-step process not only identifies vehicles within the video stream but also paves the way for detailed analysis of license plate information, such as recognition and extraction.

3.4 LICENSE PLATE RECOGNITION

3.4.1 Preparing the dataset

The workflow begins with the installation of the Roboflow library, a tool that streamlines data management and preprocessing for machine learning projects, including LPR. The library facilitates the handling of image datasets, making it easier to prepare the data for training. Within this context, a specific project and dataset are accessed using the Roboflow API. The chosen dataset likely contains a collection of images with labeled license plates, which serves as the training data for the LPR model. By leveraging this data, the system can learn to recognize and interpret license plates accurately.

3.4.2 Train the Model

In our project, we utilized YOLOv5 as our chosen model and conducted training over 120 epochs, completing the process in a notably reduced time of 0.981 hours. Additionally, in Figure of our study, we present the outcomes of YOLOv5's training on the training dataset, showcasing the recognized labels, as well as providing precision, recall, and mAP (mean Average Precision) values. This performance assessment highlights the effectiveness of our chosen YOLOv5 model in object detection tasks.

```
all      64      68      0.859      0.883      0.887      0.574

Epoch  GPU_mem  box_loss  cls_loss  dfl_loss  Instances  Size
118/120  7.95G    0.252    0.1859   0.794     8          640: 100% 38/38 [00:19<00:00, 1.99it/s]
Class   Images  Instances  Box(P   R      mAP50  mAP50-95): 100% 2/2 [00:01<00:00, 1.02it/s]
all      64      68      0.939     0.721    0.882    0.565

Epoch  GPU_mem  box_loss  cls_loss  dfl_loss  Instances  Size
119/120  7.95G    0.251    0.1839   0.7847    11         640: 100% 38/38 [00:19<00:00, 1.96it/s]
Class   Images  Instances  Box(P   R      mAP50  mAP50-95): 100% 2/2 [00:01<00:00, 1.24it/s]
all      64      68      0.896     0.764    0.881    0.561

Epoch  GPU_mem  box_loss  cls_loss  dfl_loss  Instances  Size
120/120  7.95G    0.2501   0.1851   0.7893     8          640: 100% 38/38 [00:19<00:00, 1.98it/s]
Class   Images  Instances  Box(P   R      mAP50  mAP50-95): 100% 2/2 [00:02<00:00, 1.22s/it]
all      64      68      0.912     0.76    0.882    0.567

120 epochs completed in 0.981 hours.
Optimizer stripped from /content/Automatic_Number_Plate_Detection_Recognition_YOLOv8/runs/detect/train/weights/last.pt, 52.0MB
Optimizer stripped from /content/Automatic_Number_Plate_Detection_Recognition_YOLOv8/runs/detect/train/weights/best.pt, 52.0MB

Validating /content/Automatic_Number_Plate_Detection_Recognition_YOLOv8/runs/detect/train/weights/best.pt...
Ultralytics YOLOv8.0.3 Python-3.10.12 torch-2.0.1+cu118 CUDA:0 (Tesla T4, 15102MiB)
Fusing layers...
Model summary: 218 layers, 25840339 parameters, 0 gradients, 78.7 GFLOPs
Class   Images  Instances  Box(P   R      mAP50  mAP50-95): 100% 2/2 [00:02<00:00, 1.36s/it]
all      64      68      0.845     0.8    0.886    0.58

Speed: 0.2ms pre-process, 11.9ms inference, 0.0ms loss, 1.9ms post-process per image
Saving /content/Automatic_Number_Plate_Detection_Recognition_YOLOv8/runs/detect/train/predictions.json...
Results saved to /content/Automatic_Number_Plate_Detection_Recognition_YOLOv8/runs/detect/train
```

Fig. 3.4.2 YOLOv5 Model Training

3.5 LICENSE PLATE IMAGE ENHANCEMENT

Within the realm of LPR, a crucial step involves the initial processing of license plate images to enable precise character identification. The provided segment of the process emphasizes the significance of this preparatory phase, which entails transforming license plate images into grayscale and then applying thresholding. The conversion to grayscale simplifies the license plate image by eliminating color information, resulting in a single-channel image where pixel values represent varying degrees of brightness. This simplification reduces the intricacy of the image data, streamlining subsequent processing steps to concentrate exclusively on luminance data. Grayscale images prove particularly valuable for character recognition, as they remove any potential impact from color variations that may be present.



Fig. 3.5.1 GrayScale Image

Following the grayscale conversion, the technique of thresholding is applied. This process involves converting the grayscale image into a binary format, where pixel values are categorized as either black or white based on a predetermined threshold value. In this instance, a threshold value of 64 is employed. Pixels with values equal to or exceeding 64 are rendered as black (0), while those below this threshold are depicted as white (255). The utilization of the "THRESH_BINARY_INV" flag signifies the application of inversion, effectively swapping the foreground and background colors.



Fig. 3.5.2 GrayScale Conversion

The significance of this thresholding procedure lies in its role in separating characters on the license plate from the background. Through this transformation into a binary format, the characters usually become black against a white backdrop, resulting in heightened contrast and improved visibility for subsequent optical character recognition (OCR) techniques.

3.6 LICENSE PLATE TEXT EXTRACTION

Text extraction from license plates is a critical component of license plate recognition (LPR) systems, offering valuable insights into the alphanumeric characters displayed on license plates. The process is facilitated by Optical Character Recognition (OCR) technology, which plays a pivotal role in accurately and swiftly converting visual characters into machine-readable text.

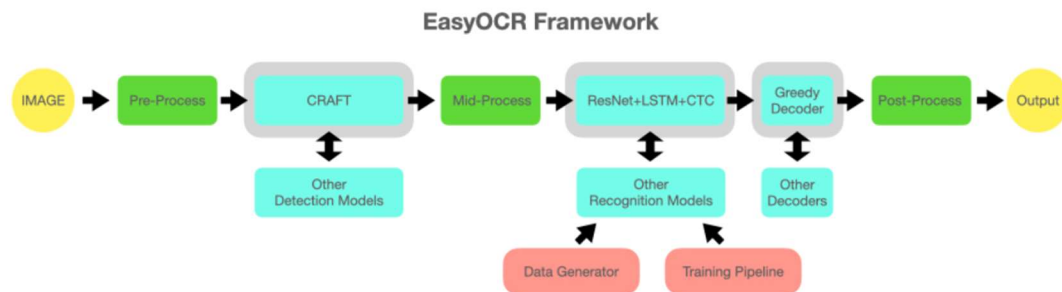


Fig. 3.6 EasyOCR Framework

EasyOCR is an OCR library that excels in recognizing text in images. It provides robust support for various languages, making it a versatile tool for text extraction tasks. In your provided code, EasyOCR is employed to recognize and extract text from license plates in English. One key aspect of the text extraction process involves formatting the extracted text to ensure consistency and accuracy. This is particularly important in license plate recognition, where license plates may exhibit variations in character styles and formats. The `'format_license'` function is responsible for this task.

Within the `'format_license'` function, character mapping dictionaries are used to handle character conversions. This is essential because license plates often include a mix of letters and numbers, and variations in character rendering can lead to recognition errors. The mapping dictionaries help standardize the characters, ensuring that the extracted text adheres to a predefined format. The OCR process itself relies on advanced image processing techniques to detect and recognize characters within the license plate region. EasyOCR employs deep learning models and neural networks to achieve high accuracy in character recognition.

4.1 INTRODUCTION

Requirement analysis is a crucial phase in software development that involves gathering, documenting, and analyzing the needs and constraints of stakeholders to define the scope of a project. It helps in understanding what the system should do and how it should behave to meet the objectives of the stakeholders. This process typically results in the identification of functional and non-functional requirements.

4.2 FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

FUNCTIONAL REQUIREMENTS:

Functional requirements specify what the system should do and describe the specific behaviors and functionalities it must possess to meet user needs and business objectives. These requirements outline the features, capabilities, and tasks that the system should support, including user interactions, data processing, and system outputs. Functional requirements are typically expressed through use case scenarios, user stories, or feature lists, and they serve as the foundation for system design, development, and testing. Some examples of functional requirements could include:

1. **Vehicle Registration Number Extraction:** The system must accurately extract registration numbers from vehicle plates using image processing techniques.
2. **User Access and Permissions:** The system should provide role-based user access and permissions to ensure data security and limit information exposure. Controlled access prevents unauthorized users from tampering with data and maintains the confidentiality of sensitive information.
3. **Logging and Auditing:** Log all vehicle activities, user interactions, and security events for auditing and compliance purposes. Records all the logging activities of the security personnel.
4. **Cloud-Based Data Transfer:** Captured vehicle data must be transferred to a cloud-based platform for storage and analysis. Cloud integration facilitates remote access, data backup.
5. **Anomaly Detection:** The system should identify anomalies or unusual patterns in vehicle movements and show them to the authorized user.
6. **Authentication and Verification:** The system should provide means for verifying the authenticity of vehicles.

7. **Reporting:** The system must display all the vehicle activities and use that for better decision making and provide insights into operations.

NON-FUNCTIONAL REQUIREMENTS:

Non-functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviours. They may relate to emergent system properties. **Non-functional requirements** are requirements that are not specifically concerned with the functionality of a system. They normally place restrictions on the product being developed and the development process. Non-functional requirements may be regarded as parameters of functionality in that they determine how quickly, how accurately, how reliably, how securely, etc., functions must operate. Some of the non-functional requirements are as follows:

1. **Security:** The system must ensure the security and integrity of captured data to prevent unauthorized access or tampering.
2. **Performance:** The system must provide real-time tracking and responsive user interface even under high loads. Performance is ensured through efficient database design, optimization of data retrieval and processing.
3. **Reliability:** The system should be available and operational at all times, with minimal downtime.
4. **Usability:** The system should offer an intuitive and user-friendly interface for various user roles. Achieved by conducting user-centered design practices, incorporating user feedback, and providing clear and concise user guides and documentation.
5. **Data Integrity:** The system must maintain the accuracy and consistency of data throughout its lifecycle.
6. **Scalability:** The system should be able to handle increasing numbers of vehicles and data without performance degradation.
7. **Maintainability:** The system should be easily maintainable, with the ability to apply updates and modifications efficiently. Maintainability is achieved by following modular design principles, using clean and well-documented code.

4.3 HARDWARE REQUIREMENTS

1. **Operating System:** Windows 10,11
2. **RAM:** 4 GB
3. **Storage:** An SSD with at least 500 GB of storage space
4. **Graphic-Card:** 1 GB
5. **Net-Bandwidth:** 10 Mbps (for Cloud Computings or AWS)

4.4 SOFTWARE REQUIREMENTS

1. **Development Software:** Python 3.12
2. **Programming Language:** Python
3. **Integrated Development Environment(IDE):** Visual Studio Code
4. **Front End Technologies:** HTML5, CSS3, Java Script
5. **Back End Technologies or Framework:** Django
6. **Database Language:** SQL
7. **Database(RDBMS):** MySQL
8. **Database Software:** WAMP or XAMPP Server
9. **Web Server or Deployment Server:** Django Application Development Server
10. **Design/Modelling:** Rational Rose

5.1 INTRODUCTION TO INPUT DESIGN

Input design is a fundamental aspect of software development, especially in applications like a medical emergency system, where accuracy, efficiency, and ease of use are paramount. Input design refers to the process of creating user interfaces and mechanisms through which users interact with the system to input data or commands. In the context of a medical emergency application, input design plays a crucial role in enabling users to register, book appointments, procure medicine, view medical reports, and perform other essential tasks seamlessly and accurately.

Effective input design is essential for ensuring that users can input data quickly, accurately, and intuitively, without encountering errors or confusion. It involves designing user-friendly interfaces, incorporating validation checks, providing feedback mechanisms, and optimizing accessibility for diverse users. By focusing on input design, the medical emergency application can enhance user experience, streamline data entry processes, minimize errors, and ultimately improve the overall efficiency and effectiveness of the system.

5.2 UML DIAGRAMS

UML is simply another graphical representation of a common semantic model. UML provides a comprehensive notation for the full life cycle of object oriented development. Unified Modelling Language is a general purpose developmental, modelling language in the field of software engineering that is intended to provide a standard way to visualize the design of the system. UML is a standard language for specifying, visualizing, and documenting of software systems and created by Object Management Group (OMG) in 1997. There are three important type of UML modelling are Structural model, Behavioral model, and Architecture model. To model a system the most important aspect is to capture the dynamic behavior which has some internal or external factors for making the interaction. These internal or external agents are known as actors. It consists of actors, use cases and their relationships. In this fig we represent the Use Case diagram for our project. The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modelling 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.

5.2.1 USE CASE DIAGRAM

Use Case Diagrams are used to depict the functionality of a system or a part of a system. They are widely used to illustrate the functional requirements of the system and its interaction with external agents(actors). A use case is basically a diagram representing different scenarios where the system can be used. A use case diagram gives us a high level view of what the system or a part of the system does without going into implementation details.



Fig. 5.2.1 Use Case Diagram

Admin begin by logging in, ensuring secure access. Following the successful login, the admin can access the mining companies, can access the security personnel, and the vehicle

management. Finally, admin has the logout option at any point concluding their interaction with the application.

5.2.2 CLASS DIAGRAM

The most widely use UML diagram is the class diagram. It is the building block of all object oriented software systems. We use class diagrams to depict the static structure of a system by showing system's classes, their methods and attributes. Class diagrams also help us identify relationship between different classes or objects. Class notation is a graphical representation used to depict classes and their relationships in object-oriented modeling.

1. **Class Name:** The name of the class is typically written in the top compartment of the class box and is centred and bold.
2. **Attributes:** Attributes, also known as properties or fields, represent the data members of the class. They are listed in the second compartment of the class box and often include the visibility (e.g., public, private) and the data type of each attribute.
3. **Methods:** Methods, also known as functions or operations, represent the behavior or functionality of the class. They are listed in the third compartment of the class box and include the visibility (e.g., public, private), return type, and parameters of each method.

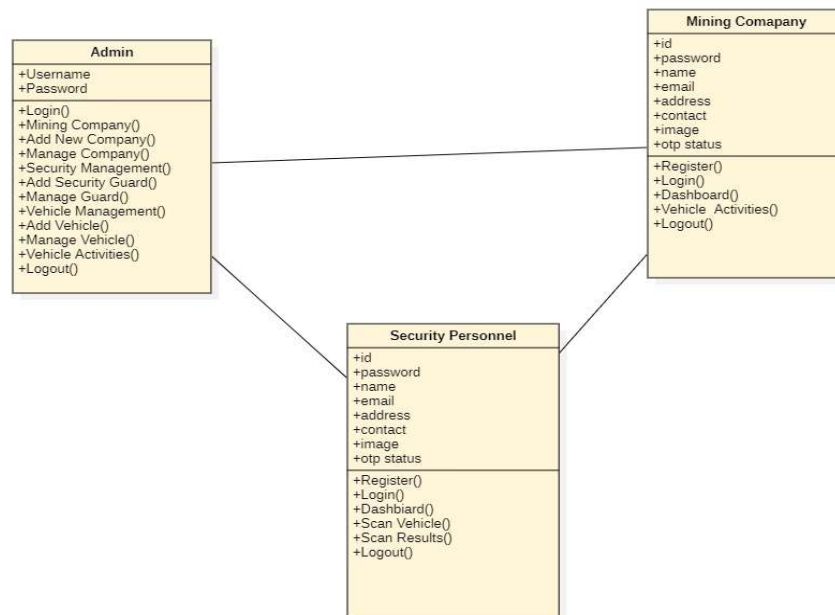


Fig. 5.2.2 Class Diagram

5.2.3 ACTIVITY DIAGRAM

Activity Diagrams is used to illustrate the flow of control in a system. We can also use an activity diagram to refer to the steps involved in the execution of a use case.

- We model sequential and concurrent activities using activity diagrams. So, we basically depict workflows visually using an activity diagram.
- An activity diagram focuses on condition of flow and the sequence in which it happens.
- We describe or depict what causes a particular event using an activity diagram.

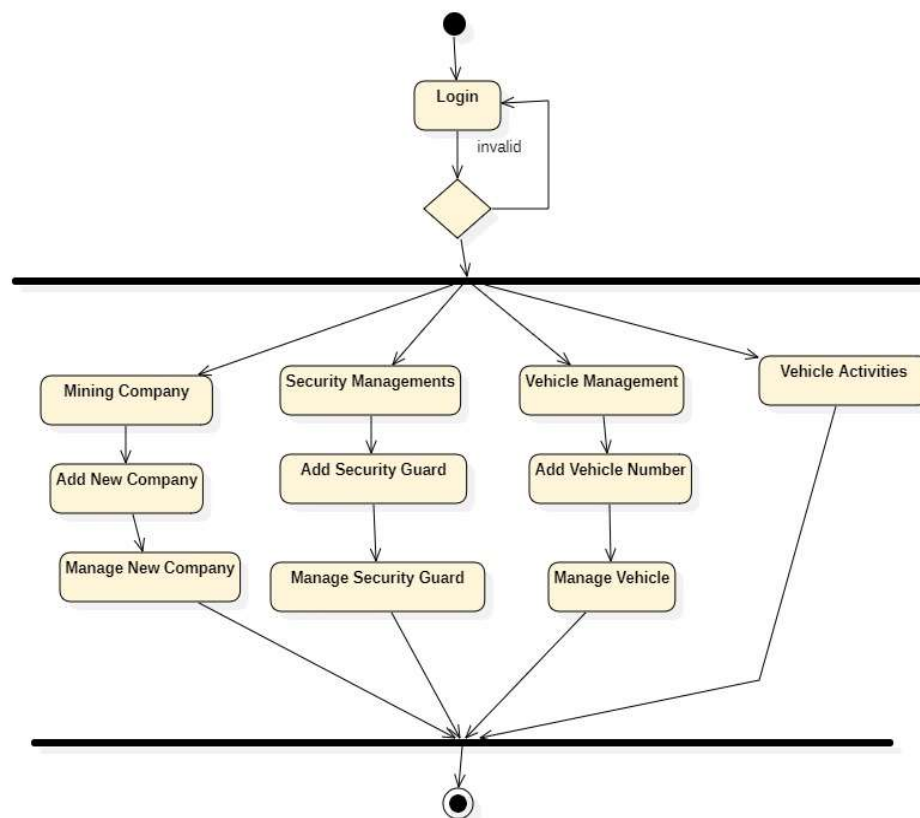


Fig. 5.2.3.1 Activity Diagram For Admin

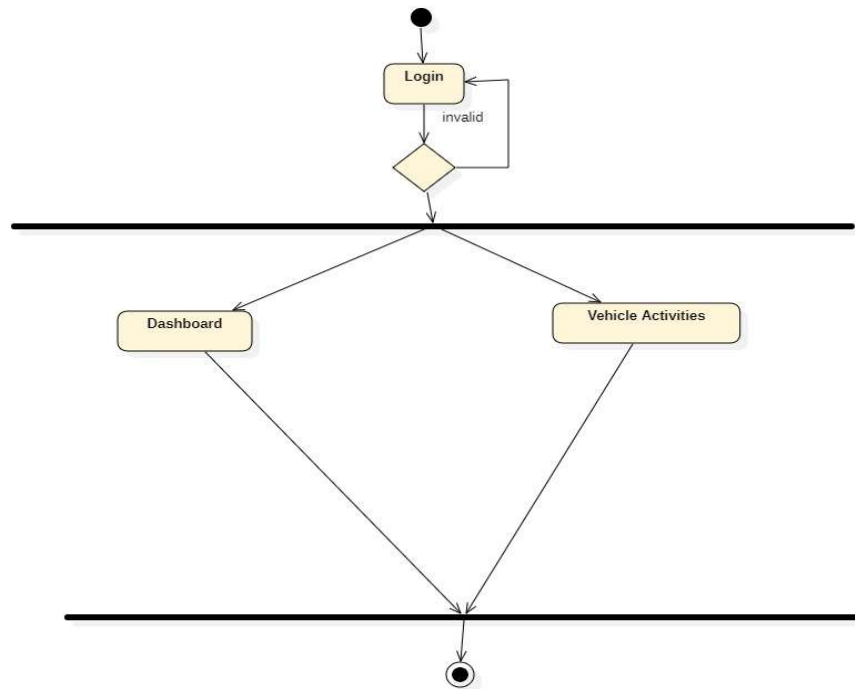


Fig. 5.2.3.2 Activity Diagram For Mining Company

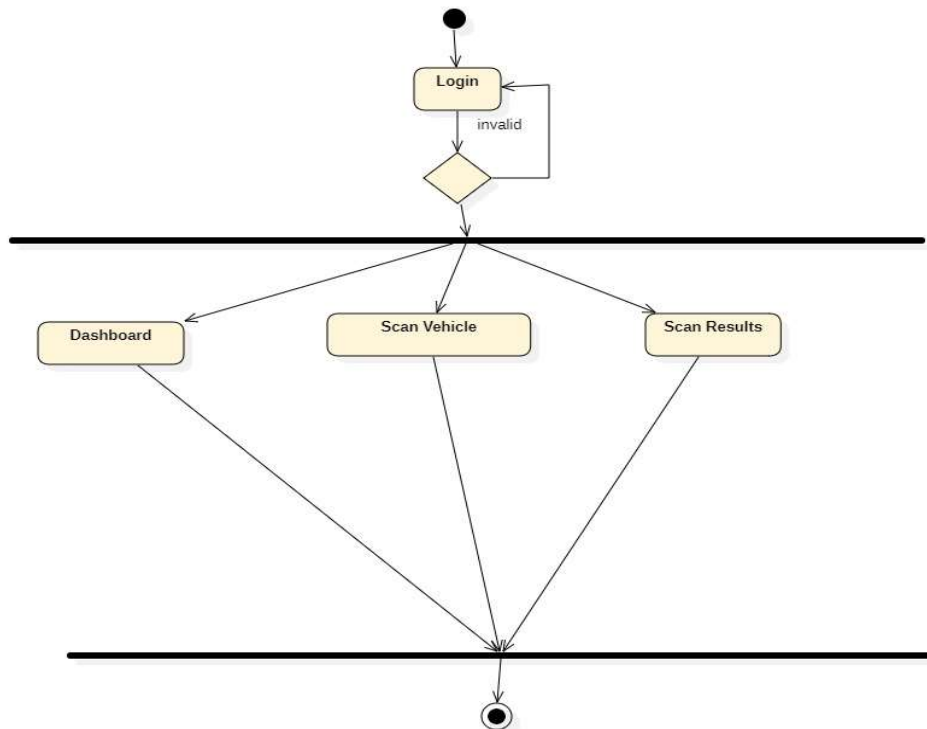


Fig. 5.2.3.3 Activity Diagram For Security Personnel

5.2.4 DATA FLOW DIAGRAM

DFD is the abbreviation for Data Flow Diagram. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present. Specific operations depending on the type of data can be explained by a flowchart. It is a graphical tool, useful for communicating with users, managers and other personnel. It is useful for analyzing existing as well as proposed system. It provides an overview of

- What data is system processes.
- What transformation are performed.
- What data are stored.
- What results are produced , etc.

Data Flow Diagram can be represented in several ways. The DFD belongs to structured-analysis modelling tools. Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes.

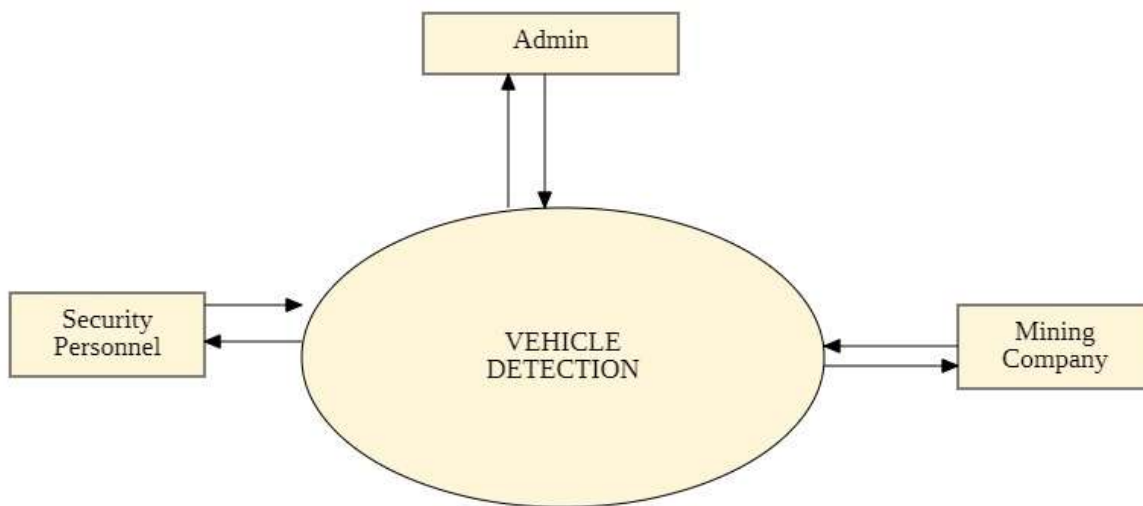


Fig. 5.2.4 Data Flow Diagram

5.2.5 SEQUENCE DIAGRAM

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

1. A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place.
2. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram.
3. Sequence diagrams describe how and in what order the objects in a system function.
4. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

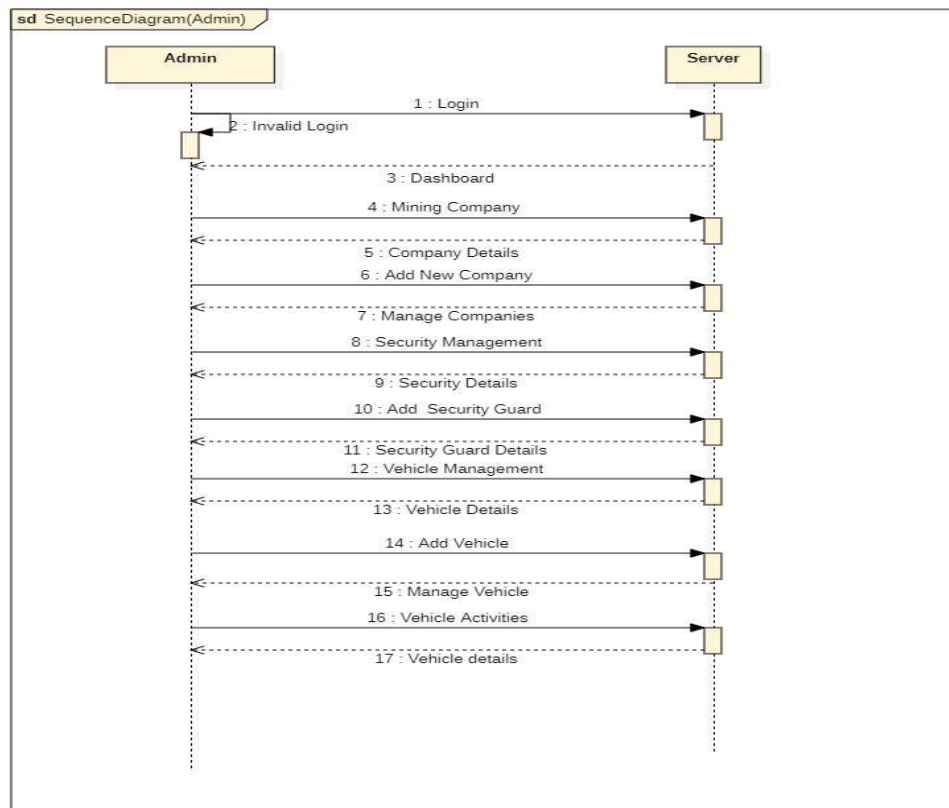


Fig. 5.2.5.1 Sequence Diagram For Admin

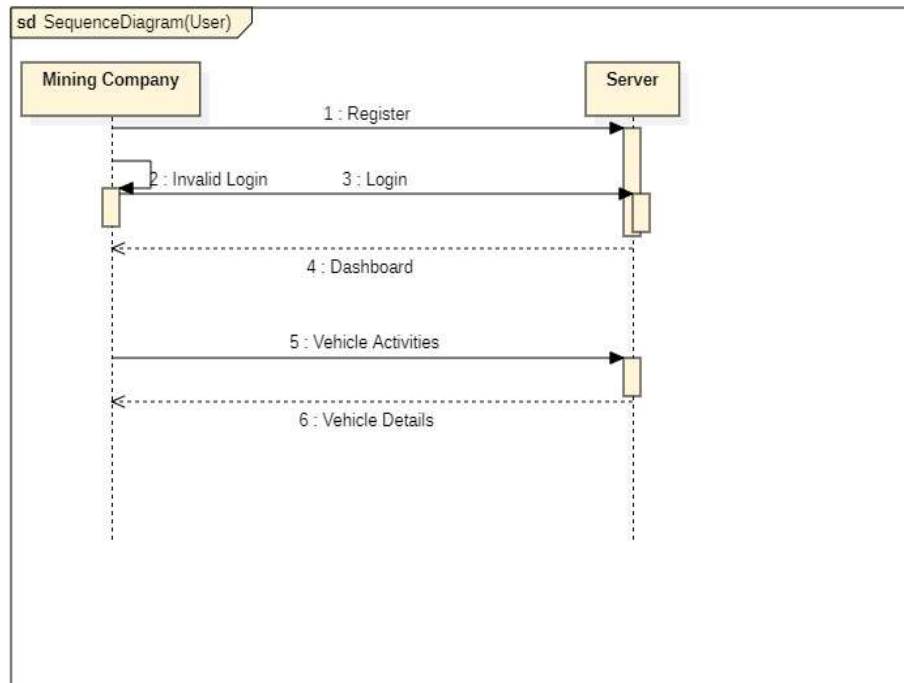


Fig. 5.2.5.2 Sequence Diagram For Mining Company

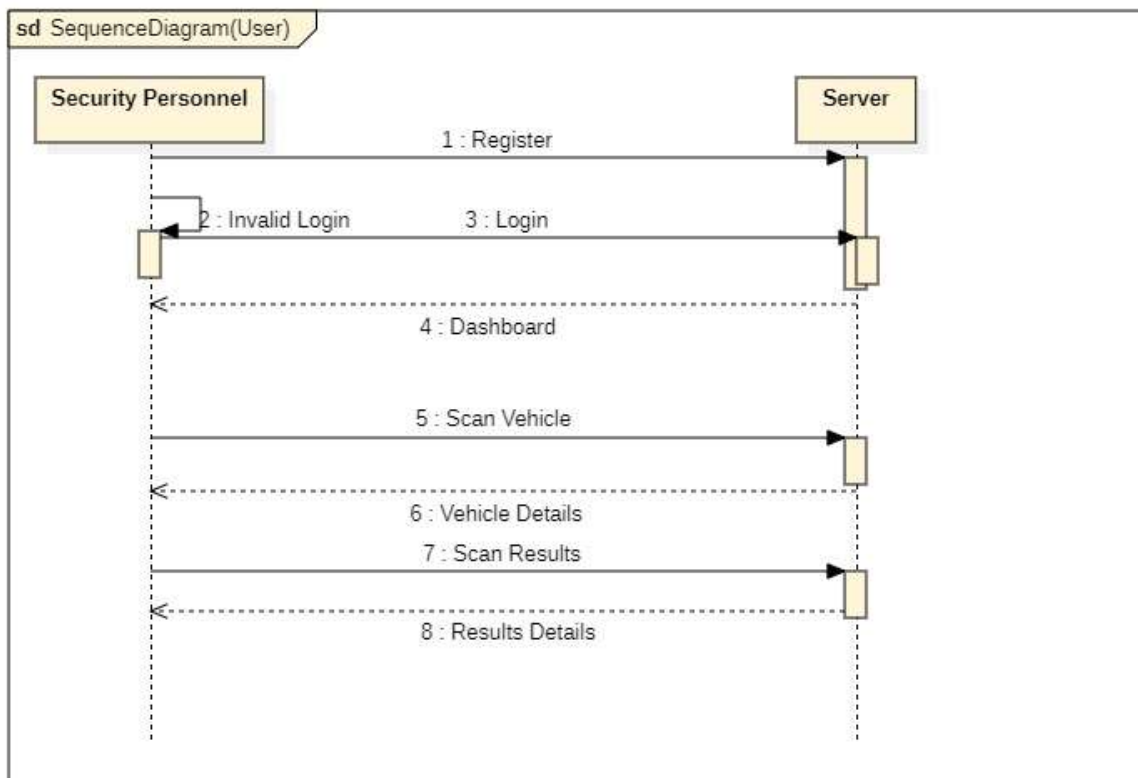


Fig. 5.2.5.3 Sequence Diagram For Security Personnel

5.2.6 COMPONENT DIAGRAM

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node. It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, and it necessitates an interface to execute a function. It is like a black box whose behavior is explained by the provided and required interfaces.

The component diagram is helpful in representing the physical aspects of a system, which are files, executables, libraries, etc. The main purpose of a component diagram is different from that of other diagrams. It is utilized in the implementation phase of any application.

Once the system is designed employing different UML diagrams, and the artifacts are prepared, the component diagram is used to get an idea of implementation. It plays an essential role in implementing applications efficiently.

Following are some artifacts that are needed to be identified before drawing a component diagram:

1. What files are used inside the system?
2. What is the application of relevant libraries and artifacts?
3. What is the relationship between the artifacts?

Following are some points that are needed to be kept in mind after the artifacts are identified:

1. Using a meaningful name to ascertain the component for which the diagram is about to be drawn.
2. Before producing the required tools, a mental layout is to be made.
3. To clarify the important points, notes can be incorporated.

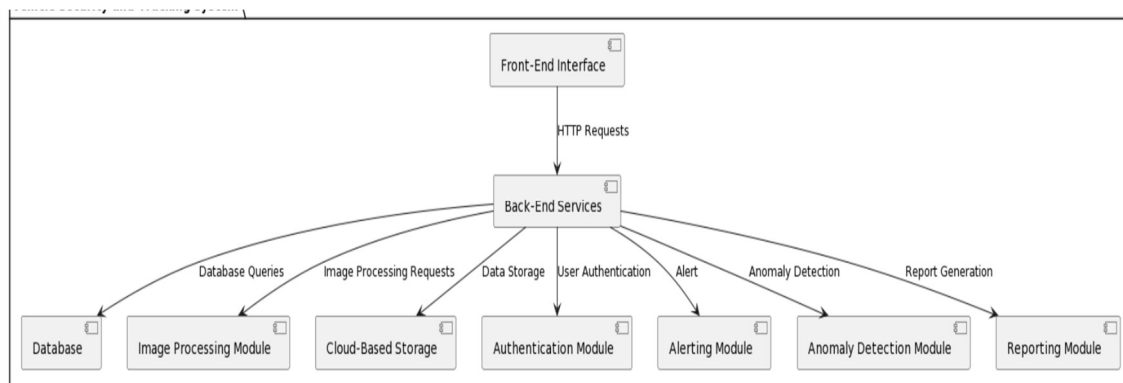


Fig. 5.2.6 Component Diagram

5.2.7 DEPLOYMENT DIAGRAM

The deployment diagram visualizes the physical hardware on which the software will be deployed. It portrays the static deployment view of a system. It involves the nodes and their relationships.

It ascertains how software is deployed on the hardware. It maps the software architecture created in design to the physical system architecture, where the software will be executed as a node. Since it involves many nodes, the relationship is shown by utilizing communication paths.

The main purpose of the deployment diagram is to represent how software is installed on the hardware component. It depicts in what manner a software interacts with hardware to perform its execution.

Both the deployment diagram and the component diagram are closely interrelated to each other as they focus on software and hardware components. The component diagram represents the components of a system, whereas the deployment diagram describes how they are actually deployed on the hardware.

The deployment diagram does not focus on the logical components of the system, but it put its attention on the hardware topology.

Following are the purposes of deployment diagram enlisted below:

1. To envision the hardware topology of the system.
2. To represent the hardware components on which the software components are installed.
3. To describe the processing of nodes at the runtime.

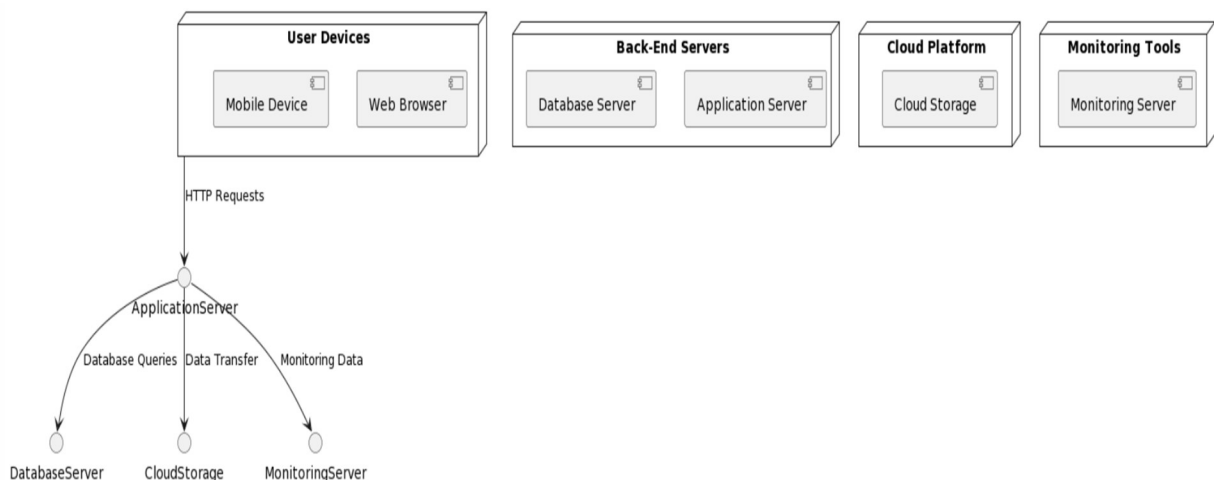


Fig. 5.2.7 Deployment Diagram

6.1 MODULES

6.1.1 Data Pre-Processing

The entries are present in the dataset. The null values are removed using `df = df.dropna()` where `df` is the data frame. The categorical attributes (Date, High, Low, Close, Adj value) are converted into numeric using Label Encoder. The date attribute is splitted into new attributes like total which can be used as feature for the model.

1.DataCleaning:

The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc.

2.DataTransformation:

This step is taken in order to transform the data in appropriate forms suitable for mining process.

3.DataReduction:

Since data mining is a technique that is used to handle huge amount of data. While working with huge volume of data, analysis became harder in such cases. In order to get rid of this, we use data reduction technique. It aims to increase the storage efficiency and reduce data storage and analysis costs.

6.1.2 Number Plate Extraction

The images are given to the model and these images are stored in the database. The path is specified to where these images are stored. And the date and time of the image uploaded is also recorded in the database. Now, we make the prediction of the image using YOLOv5 assuming it is a function for object detection. Now crop the predicted image.

6.1.3 Character Segmentation

The character segmentation part further segments the character individually from the extracted number plate. From input image the first process will be to crop out the number plate characters from starting to the ending point leaving all the extra wide spaces from top to below and from right to left as it is. Characters are equally fit in the plate region. For easy comparison of the input character with the character in the data base the result is normalized into the character set as the size of the images in the database.

6.1.4 Optical Character Recognition

The optical character recognition is a recognition method in which the input is an image and the output is string of character. OCR is a process which separates the different characters from each other taken from an image. Template matching is one of the approaches of OCR. The cropped image is compared with the template data stored in database. OCR automatically identifies and recognizes the characters without any indirect input. The characters on the number plate have uniform fonts then the OCR for number plate recognition is less complex as compared to other methods. We use easyOCR to extract the text from the image.

6.2 SOURCE CODE

6.2.1 Implementation of Scanning the Image of vehicle

```
BEST_WEIGHTS_PATH = r"yolov5\runs\train\my_yolov5_model_s\weights\best.pt"
```

```
# Uploading the best
```

```
yolov5 = torch.hub.load(
```

```
    repo_or_dir='yolov5',
```

```
    model='custom',
```

```
    path=BEST_WEIGHTS_PATH,
```

```
    source='local'
```

```
)
```

```
from django.shortcuts import render,redirect
```

```
from django.conf import settings
```

```
from django.contrib import messages
```

```
from django.contrib.auth import authenticate, login
```

```
from adminapp.models import *
```

```
import os
```

```
import easyocr
```

```
import torch
```

```

from django.core.files.storage import default_storage

from userapp.models import *

from datetime import datetime

def user_scan(request):

    if request.method == 'POST' and request.FILES.get('scan'):

        number_plate_file = request.FILES['scan']

        path = default_storage.save(number_plate_file.name, number_plate_file)

        url = default_storage.url(path)

        print(url)

        current_datetime = datetime.now()

        current_date = current_datetime.date()

        current_time = current_datetime.time()

        print("Path of the saved image:", path)

        print("Current date:", current_date)

        print("Current time:", current_time)

        with open('temp_image.png', 'wb') as f:

            for chunk in number_plate_file.chunks():

                f.write(chunk)

        prediction = yolov5('temp_image.png') # Assuming 'yolov5' is a function for object
detection

        cropped = prediction.crop()

        car_numbers = []

        status_list = [] # List to store status of each car number

        for pred in cropped:

            if pred['conf'] < 0.7:

```

```

        continue

# Retrieve the cropped out image

img_cropped = pred['im']

# Use EasyOCR to extract text from the image

reader = easyocr.Reader(['en'])

results = reader.readtext(img_cropped)

# Iterate over the extracted results and concatenate the car numbers

plate_number = ""

for result in results:

    car_number = result[1]

    plate_number += car_number + ' ' # Concatenate the car number with a space

car_numbers.append(plate_number.strip()) # Remove trailing space and append to the
list

print(car_numbers,"iouiyutyrtterwewrty")

car_numbers_str = ''.join(car_numbers)

print(car_numbers_str,"uythgdsfddhfyu")

# Check status of the car number and append to status_list

car = CarNumber.objects.get(number=car_numbers_str)

car_number = car.number

car_status = car.user_status

# status_list.append(car.user_status)

print(car.user_status,"gdagdgdgajgd")

try:

    pass

except CarNumber.DoesNotExist:

```

```

        status_list.append('Unauthorized')

    scanned_vehicle = ScannedVehicle.objects.create(

        image=path,

        time=current_time,

        date=current_date,

        number_plate=car_numbers_str, # You'll need to update this with the actual number
plate detected

        status=car_status # You'll need to update this with the actual status

    )

    scanned_vehicle.save()


    # Pass the extracted car numbers and their status to the template

    return render(request, "user/result.html", {'car_numbers': car_number, 'car_status':
car_status})

else:

    return render(request, "user/scan_vehicle.html")

```

OUTPUT SCREENS

This web application combines the most recent technology to create a robust and user-friendly experience. The client side of the application boasts a striking design, made using the latest combination of HTML5, CSS3, and JavaScript. By ensuring that the user experience is seamless across devices, these technologies establish a foundation for engaging interactions. Django, a Python-based framework that is known for its flexibility and effectiveness, is utilized by the application on the server side. Django adeptly manages the application's logic, guaranteeing seamless functionality and data exchange. To ensure the integrity and accessibility of the data in the application, we've implemented a dual database strategy: MySQL with strong transactional capabilities and MongoDB with flexible handling of different data types. With care, this carefully selected tech stacks produce an enthralling web app that is both user-friendly and reliable.

Vehicle Registration Plate Detection

In the context of our project, we harnessed the capabilities of YOLOv5 and EasyOCR as our core models. YOLOv5, specifically the YOLOv5m variant, played a pivotal role in our pursuit of license plate detection. Through meticulous training on our custom dataset, this model demonstrated exceptional proficiency in identifying license plates within images. Operating at an image resolution of 640 pixels, it proved to be an optimal choice for the task, balancing accuracy and computational efficiency. We fine-tuned the model through an extensive training regimen spanning 150 epochs, optimizing its performance further with a batch size of 5.

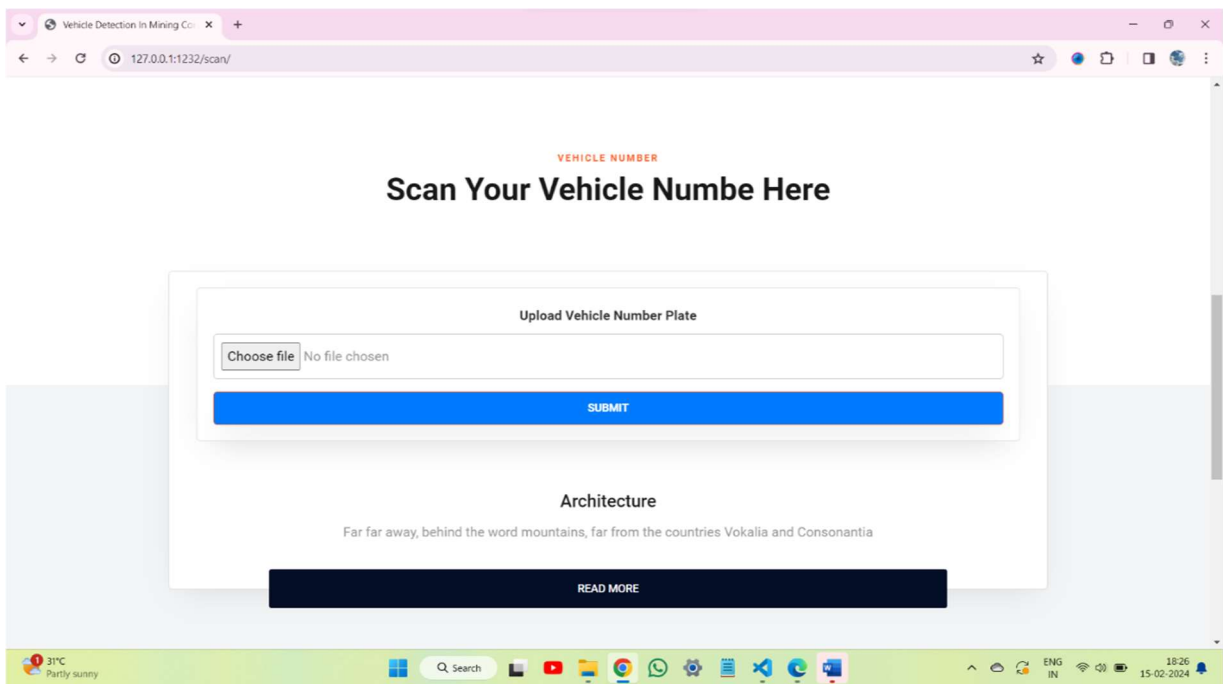


Fig. 7.1 Upload the Vehicle Image

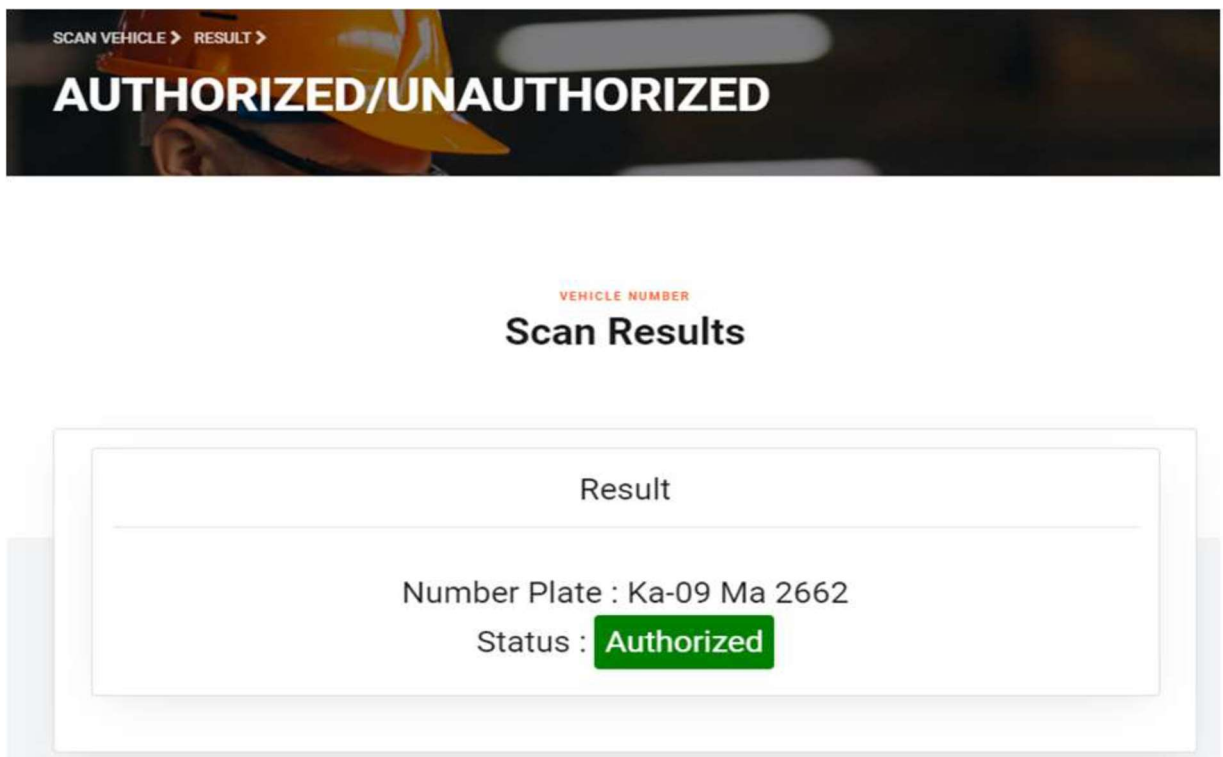


Fig. 7.2 Results of the Uploaded Vehicle Image

Vehicle Activities					
Page 1 of 2.					
ID	Vehicle	Time	Vehicle Number	Date	Vehicle status
1		3:35 p.m.	HR 26 BC SS14	March 11, 2024	Authorized
2		3:47 p.m.	Ka-09 Ma 2662	March 11, 2024	Authorized
3		7:21 p.m.	Ka-09 Ma 2662	March 12, 2024	Authorized
4		12:09 p.m.	Ka-09 Ma 2662	March 13, 2024	Authorized
5		12:08 p.m.	Ka-09 Ma 2662	March 16, 2024	Authorized

Fig. 7.3 Logging All the Vehicle Details

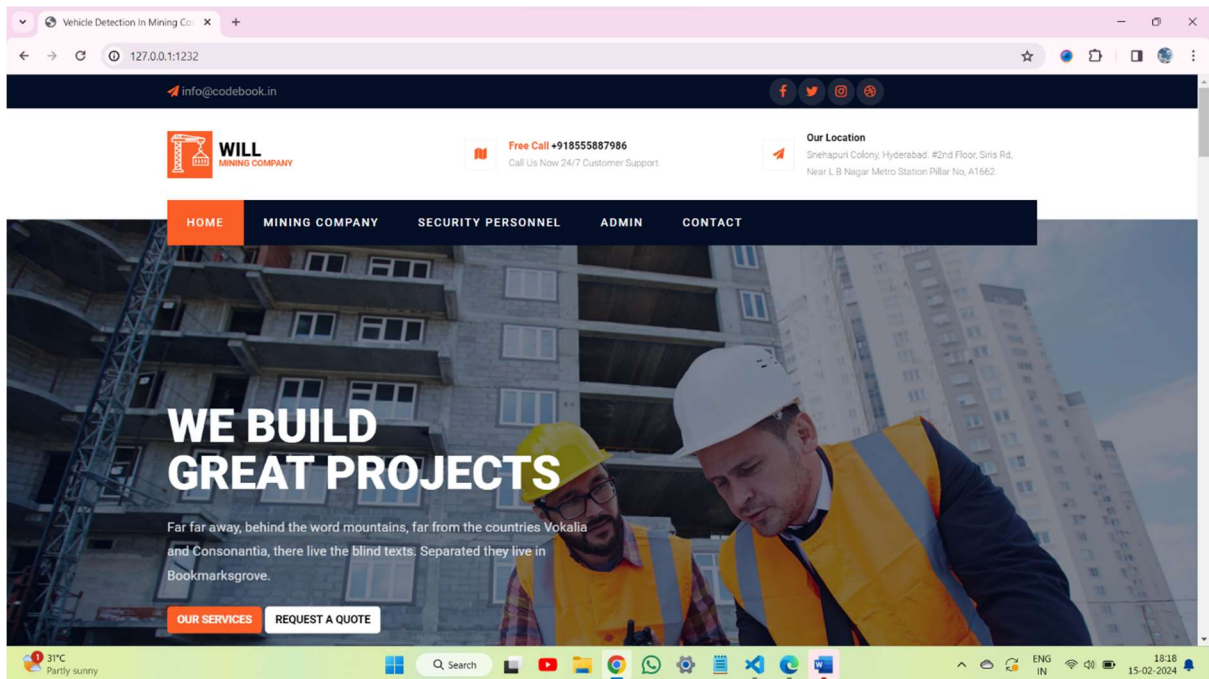


Fig. 7.4 Home Page

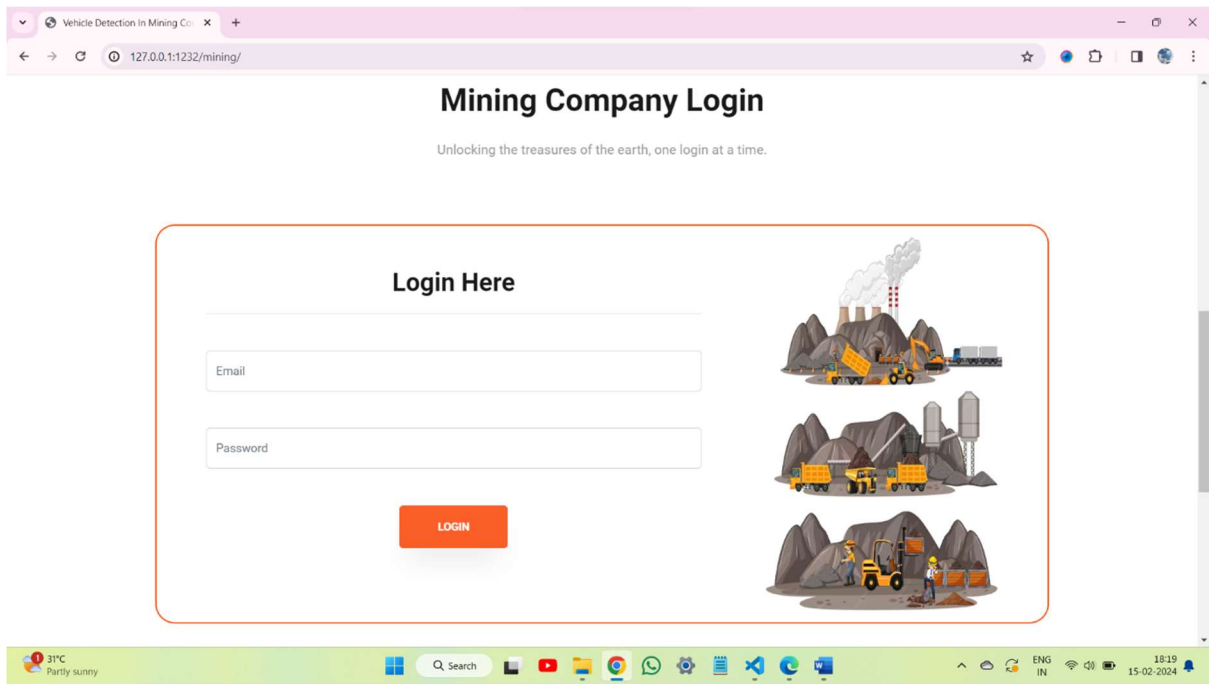


Fig. 7.5 Mining Company Login Page

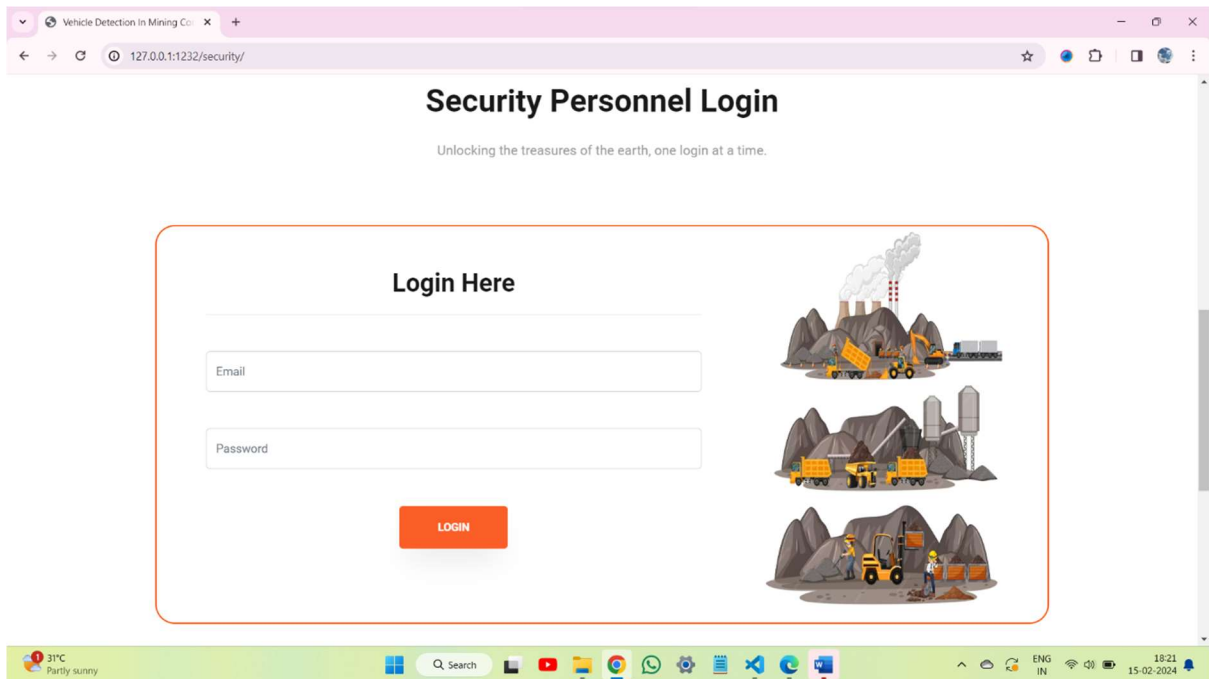


Fig. 7.6 Security Personnel Login Page

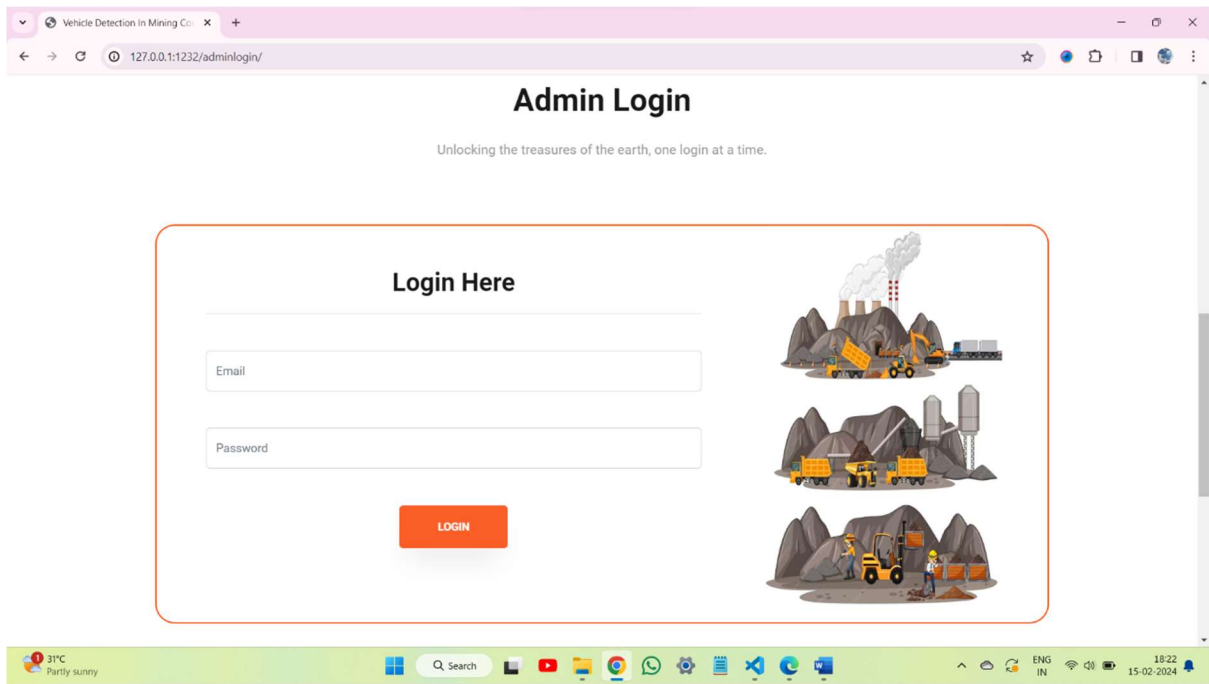


Fig. 7.7 Admin Login Page

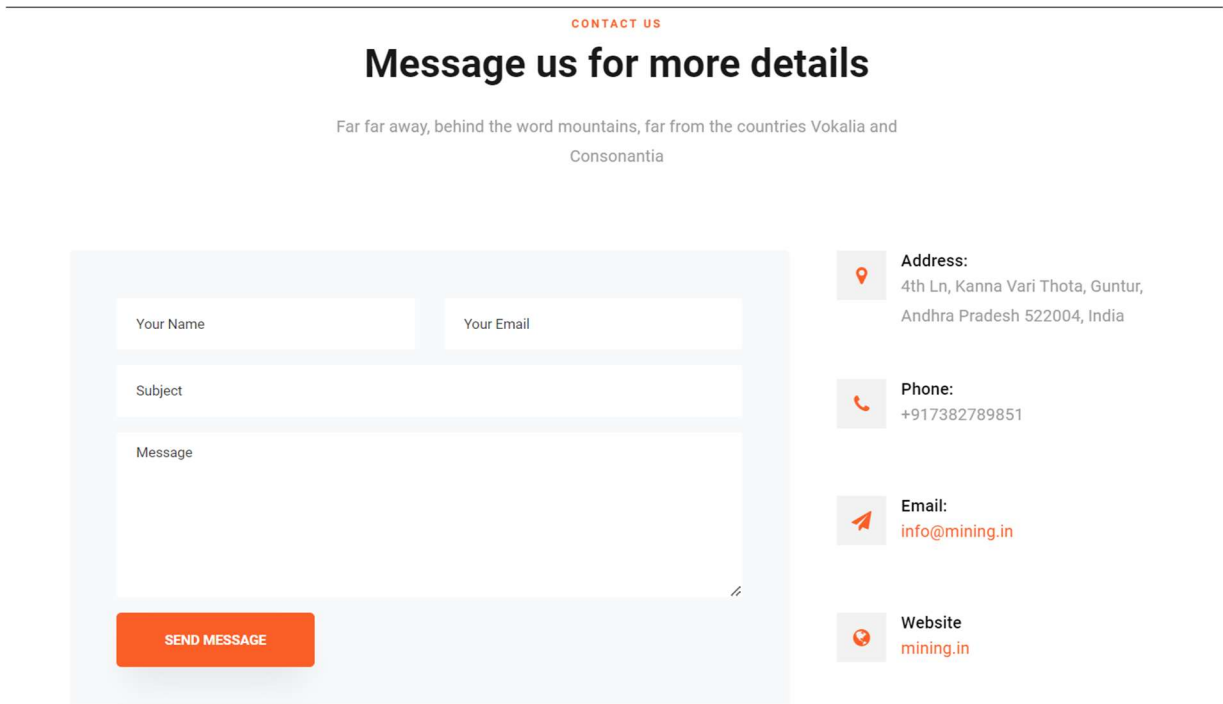


Fig. 7.8 Contact Us Page

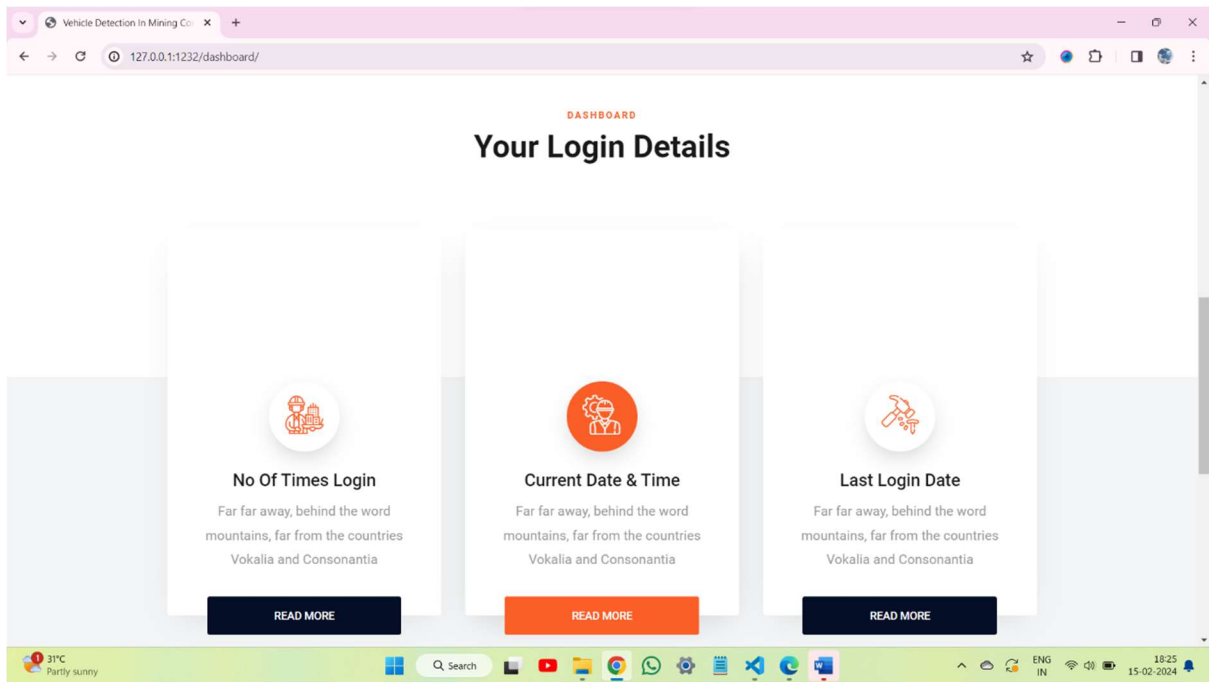


Fig. 7.9 Security Personnel Dashboard

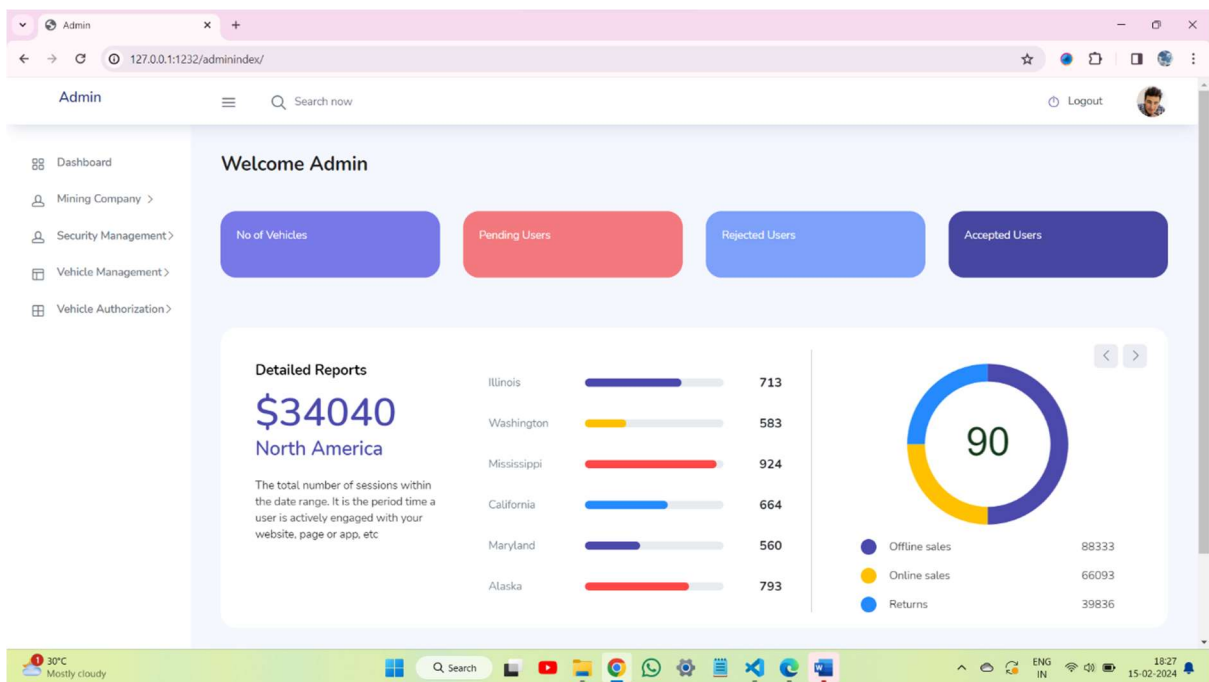


Fig. 7.10 Admin Dashboard

Add Company

Company Name

E-Mail

Location

Employee

Document Proof

Submit

Fig. 7.11 Adding the Mining Company

Manage Companies

ID	Document Proof	Company Name	Company E-mail	Company Location	Employee Count	Password	Delete Company
1		mining	penumallijothendra@gmail.com	Hyd	200	crTRYsVh	Delete
2		Miner	sampathkumarmuthayalapati@gmail.com	hyderabad	300	zbA1KAUD	Delete
3		Miner	penumallijothendra@gmail.com	hyderabad	300	5o9J7PPe	Delete

Page 1 of 1.

Copyright © mining.in All rights reserved. Created by Mining Developer 2023

Fig. 7.12 Managing the Mining Companies

Add New User

Username

E-Mail

Contact

Password

[Submit](#)

Fig. 7.13 Adding Security Personnel Details

Manage Security Personnels

ID	Document Proof	Username	E-mail	Contact	Password	Delete Guard
1		Jyo	penumallijyothendra@gmail.com	7382789851	b7Qm	Delete

Page 1 of 1.

Copyright © mining.in All rights reserved. Created by Mining Developer 2023

Fig. 7.14 Managing Security Personnel

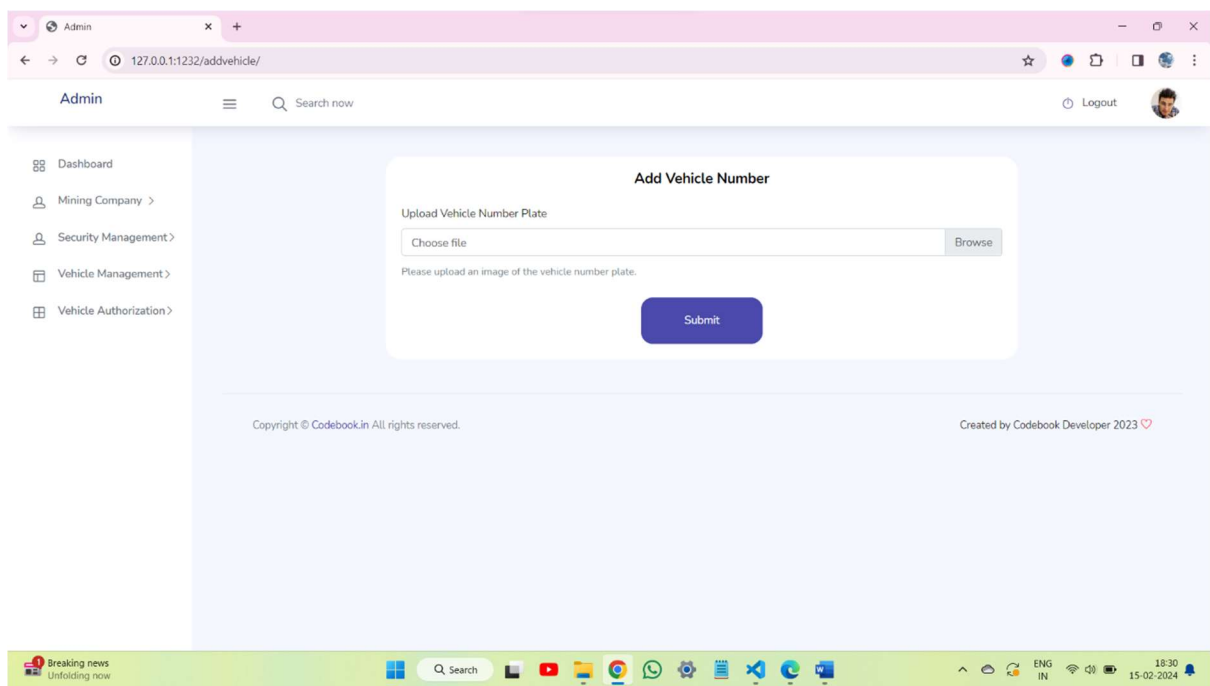


Fig. 7.15 Adding Vehicle to Admin Database

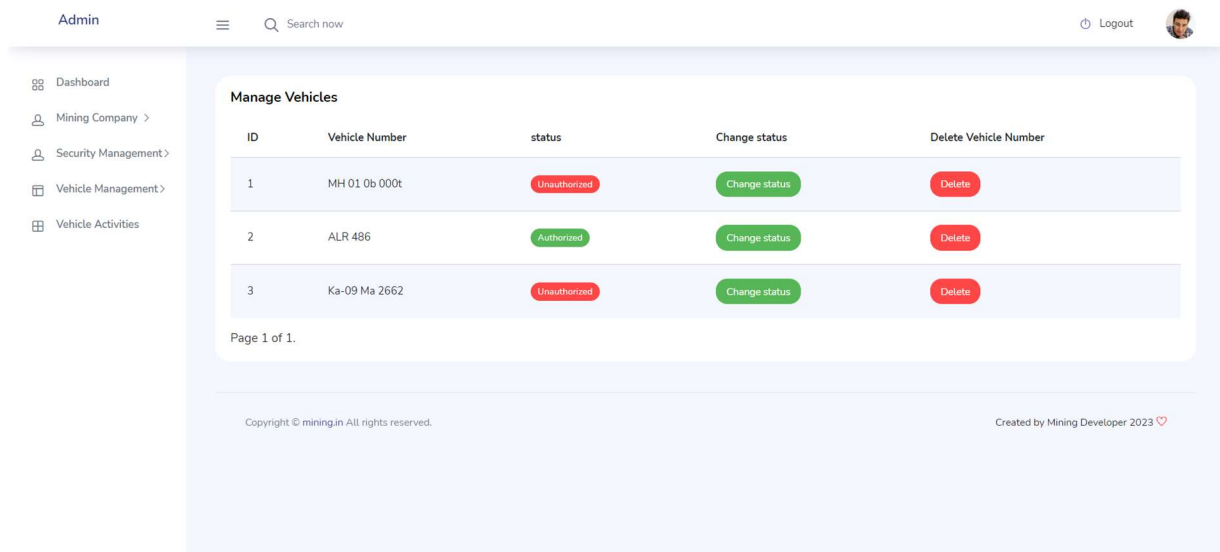


Fig. 7.16 Managing the Vehicles

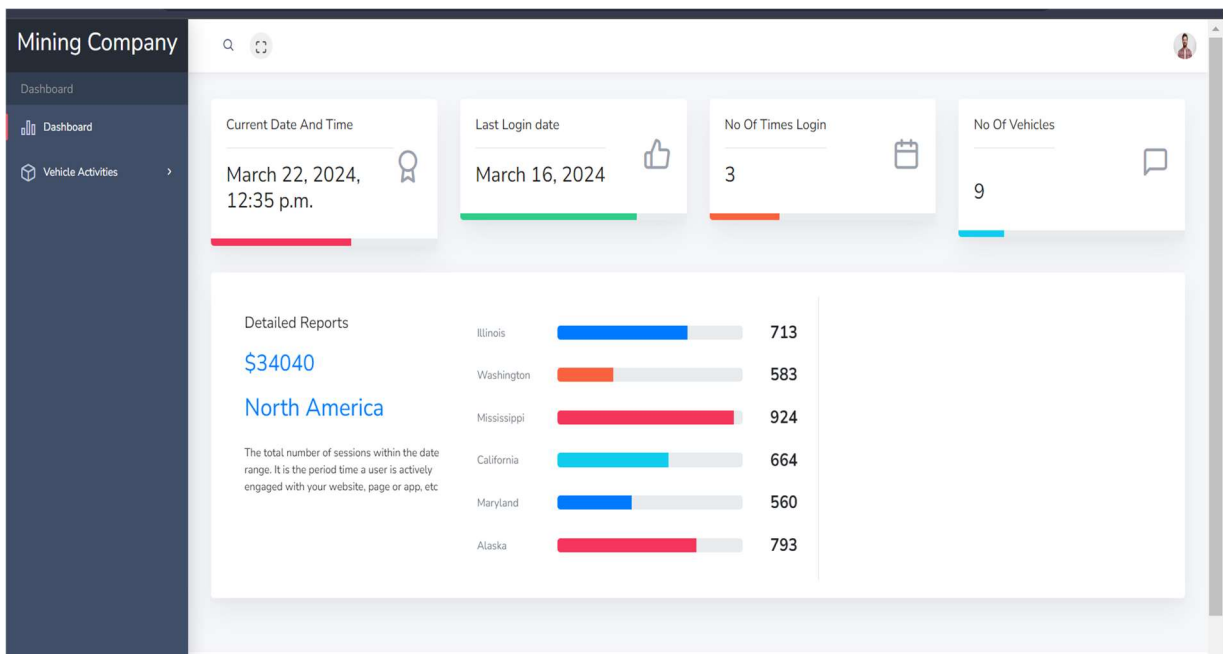


Fig. 7.17 Mining Company Dashboard

Page of .






ID	Vehicle	Vehicle Number	Time	Date	Vehicle status
1		3:35 p.m.	HR 26 BC SS14	March 11, 2024	Authorized
2		3:47 p.m.	Ka-09 Ma 2662	March 11, 2024	Authorized
3		7:21 p.m.	Ka-09 Ma 2662	March 12, 2024	Authorized
4		12:09 p.m.	Ka-09 Ma 2662	March 13, 2024	Authorized
5		12:08 p.m.	Ka-09 Ma 2662	March 16, 2024	Authorized

Fig. 7.18 Vehicle Activities in Mining Companies

Incorporating YOLOv5 and EasyOCR into our project, we have achieved real-time Automatic Number Plate Recognition (ANPR) capabilities. This integration harnesses the power of GPU acceleration to enhance the speed of both object detection and character recognition, rendering them well-suited for real-time applications. YOLOv5 has notably outperformed its predecessors in terms of speed and accuracy, making it a superior choice for object detection. Our YOLOv5 model, which has undergone successful training using a custom dataset tailored for object detection, demonstrates remarkable performance compared to previous YOLO versions. Additionally, we have achieved an impressive 95% accuracy in character recognition with EasyOCR, reinforcing its position as an excellent choice for text extraction tasks. Moreover, through the collaborative efforts of EasyOCR and YOLOv5, we have attained a commendable accuracy rate of 92%. This synergy between state-of-the-art object detection and character recognition technologies significantly enhances the overall effectiveness and reliability of our ANPR system, making it a promising solution for various real-world applications.

- [1] Shashirangana, Jithmi, et al. "Automated license plate recognition: a survey on methods and techniques." *IEEE Access* 9 (2020): 11203- 11225.
- [2] Jamtsho, Yonten, Panomkhawn Riyamongkol, and Rattapoom Waranusast. "Real-time license plate detection for non-helmeted motorcyclist using YOLO." *Ict Express* 7.1 (2021): 104-109.
- [3] R Shashidhar, A S Manjunath, R Santhosh Kumar, M Roopa, S B Puneeth. "Vehicle Number Plate Detection and Recognition using YOLO- V3 and OCR Method" , 2021 IEEE International Conference on Mobile Networks and Wireless Communications (ICMNBC), 2021 Publication
- [4] Pinto, Pedro F. A. et al. "PVBR-Recog: A YOLOv3-based Brazilian Automatic License Plate Recognition Tool." *Anais Estendidos do Simpósio Brasileiro de Sistemas Multimídia e Web (WebMedia)* (2019): n. pag
- [5] Prajwal M J., Tejas K B., Varshad V., Mahesh Madivalappa Murgod and Shashidhar R "Detection of Non-Helmet Riders and Extraction of License Plate Number using Yolo v2 and OCR Method" *International journal of Innovative Technology and Exploring Engineering (IJITEE)* Volume-9, Issue-2, December 2019 DOI: 10.35940/ijitee.B6527.129219.
- [6] Yuchao SUN, Qiao PENG, Dengyin ZHANG, "Light-YOLOv3: License Plate Detection in Multi-Vehicle Scenario: Regular Section," *IEICE Transactions on Information and Systems*, 2021, E104.D(5) : 723-728.
- [7] R. K. Varma P, S. Ganta, H. K. B, and P. Svrsk, "A Novel Method for Indian Vehicle Registration Number Plate Detection and Recognition using Image Processing Techniques," *Procedia Comput Sci*, vol. 167, pp. 2623–2633, 2020, doi: 10.1016/j.procs.2020.03.324.
- [8] C. Henry, S. Y. Ahn, and S. W. Lee, "Multinational License Plate Recognition Using Generalized Character Sequence Detection," *IEEE Access*, vol. 8, pp. 35185–35199, 2020, doi: 10.1109/ACCESS.2020.2974973.
- [9] K. Tejas, K. Ashok Reddy, D. Pradeep Reddy, K. P. Bharath, R. Karthik, and M. Rajesh Kumar, "Efficient license plate recognition system with smarter interpretation through IoT," *Advances in Intelligent Systems and Computing*, vol. 817, pp. 207–220, 2019, doi: 10.1007/978-981-13-1595-4_16/COVER.
- [10] M. Molina-Moreno, I. Gonzalez-Diaz, and F. Diaz-De-Maria, "Efficient Scale-Adaptive License Plate Detection System," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 6, pp. 2109–2121, Jun. 2019, doi: 10.1109/TITS.2018.2859035.
- [11] Y. Li, D. Niu, X. Chen, T. Li, Q. Li, and Y. Xue, "Vehicle license plate recognition combining MSER and support vector machine in a complex environment," *Chinese Control Conference, CCC*, vol. 2019-July, pp. 7045–7050, Jul. 2019, doi: 10.23919/CHICC.2019.8865171.
- [12] P. L. Kumari, R. Tharuni, I. V. S. Sai Vasanth, and M. Vinay Kumar, "Automatic license plate detection using KNN and convolutional neural network," in *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, 2022.
- [13] "heartexlabs/labelImg: LabelImg is now part of the Label Studio community. The popular image annotation tool created by Tzutalin is no longer actively being developed, but you can check out Label Studio, the open source data labeling tool for images, text, hypertext, audio, video and time-series data." <https://github.com/heartexlabs/labelImg>(accessed May 21, 2023).

- [14] J. Shashirangana, H. Padmasiri, D. Meedeniya, and C. Perera, “Automated license plate recognition: A survey on methods and techniques,” *IEEE Access*, vol. 9, pp. 11203–11225, 2021.
- [15] F. N. M. Ariff, A. S. A. Nasir, H. Jaafar, and A. N. Zulkifli, “Character segmentation for automatic vehicle license plate recognition based on fast K-means clustering,” in 2020 IEEE 10th International Conference on System Engineering and Technology (ICSET), 2020.
- [16] A. Mushthofa, A. Bejo, and R. Hidayat, “The improvement of character recognition on ANPR algorithm using CNN method with efficient grid size reduction,” in 2020 6th International Conference on Science and Technology (ICST), 2020.
- [17] Z. Xu, W. Yang, A. Meng, N. Lu, H. Huang, C. Ying, and L. Huang, “Towards end-to-end license plate detection and recognition: A large dataset and baseline,” in *Proc. Eur. Conf. Comput. Vis.*, in *Lecture Notes in Computer Science*, vol. 11217, 2018, pp. 261–277.
- [18] M. Valdeos, A. S. Vadillo Velazco, M. G. Perez Paredes, and R. M. Arias Velasquez, “Methodology for an automatic license plate recognition system using Convolutional Neural Networks for a Peruvian case study,” *IEEE Lat. Am. Trans.*, vol. 20, no. 6, pp. 1032–1039, 2022.
- [19] P. Huang and W. Wang, “Research and design of automatic license plate recognition system based on android platform,” in 2022 IEEE 6th Information Technology and Mechatronics Engineering Conference (ITOEC), 2022.
- [20] X. Fan and W. Zhao, “Improving robustness of license plates automatic recognition in natural scenes,” *IEEE Trans. Intell. Transp. Syst.*, pp. 1–10, 2022.
- [21] H. Xu, X.-D. Zhou, Z. Li, L. Liu, C. Li, and Y. Shi, “EILPR: Toward end-to-end irregular license plate recognition based on automatic perspective alignment,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2586–2595, 2022.
- [22] P. L. Kumari, R. Tharuni, I. V. S. Sai Vasanth, and M. Vinay Kumar, “Automatic license plate detection using KNN and convolutional neural network,” in 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), 2022.
- [23] Y. Wen, Y. Lu, J. Yan, Z. Zhou, K. M. von Deneen, and P. Shi, “An algorithm for license plate recognition applied to intelligent transportation system,” *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 830–845, Sep. 2011.
- [24] D. Wang, Y. Tian, W. Geng, L. Zhao, and C. Gong, “LPRNet: Recognizing Chinese license plate in complex environments,” *Pattern Recognit. Lett.*, vol. 130, pp. 148–156, Feb. 2020.
- [25] S. Azam and M. M. Islam, “Automatic license plate detection in hazardous condition,” *J. Vis. Commun. Image Represent.*, vol. 36, pp. 172–186, Apr. 2016.
- [26] A. Rio-Alvarez, J. de Andres-Suarez, M. Gonzalez-Rodriguez, D. Fernandez-Lanvin, and B. L. Pérez, “Effects of challenging weather and illumination on learning-based license plate detection in noncontrolled environments,” *Sci. Program.*, vol. 2019, pp. 1–16, Jun. 2019.
- [27] Y.-T. Chen, J.-H. Chuang, W.-C. Teng, H.-H. Lin, and H.-T. Chen, “Robust license plate detection in nighttime scenes using multiple intensity IR-illuminator,” in *Proc. IEEE Int. Symp. Ind. Electron.*, May 2012, pp. 893–898.

- [28] K. S. Raghunandan, P. Shivakumara, H. A. Jalab, R. W. Ibrahim, G. H. Kumar, U. Pal, and T. Lu, “Riesz fractional based model for enhancing license plate detection and recognition,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 9, pp. 2276–2288, Sep. 2018.
- [29] R. Laroca, E. Severo, L. A. Zanolensi, L. S. Oliveira, G. R. Goncalves, W. R. Schwartz, and D. Menotti, “A robust realtime automatic license plate recognition based on the YOLO detector,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–10.
- [30] G.-S. Hsu, A. Ambikapathi, S.-L. Chung, and C.-P. Su, “Robust license plate detection in the wild,” in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2017, pp. 1–6. [14] J. Shashirangana, H. Padmasiri