**MULTI PARKING SITE**

**A PROJECT REPORT**

***Submitted by***

**HARI KRISHNAN R (110817104022)**

## BHARATH S (110817104009)

## KRISHANT THARUN K (110817104033)

***in partial fulfillment for the award of the degree***

***of***

## BACHELOR OF ENGINEERING

***in***

COMPUTER SCIENCE AND ENGINEERING

**JAYA ENGINEERING COLLEGE, THIRUNINRAVUR**

**ANNA UNIVERSITY: CHENNAI 600 025**

APRIL2021

**ANNA UNIVERSITY : CHENNAI 600 025**

## BONAFIDE CERTIFICATE

Certified that this project report **“MULTI PARKING SITE”,** is the

bonafide work of **“ HARI KRISHNAN R (110817104022) , BHARATH**

**S (110817104009) , KRISHANT THARUN K (110817104033) ”,** who

carried out the project work under my supervision.

### Prof. M. KUMARAN Dr . J. Jebamalar tamilselvi

Head of the Department, Supervisor &Asst. Professor,

Department of CSE, Department of CSE,Jaya Engineering College, Jaya Engineering College,Thiruninravur - 602024. Thiruninravur - 602024.

**ANNA UNIVERSIT : CHENNAI 600 025**

## VIVA VOCE EXAMINATION

The viva-voce examination of the project work titled **“MULTI PARKING SITE”**, submitted by “**HARI KRISHNAN R (110817104022) , BHARATH S (110817104009) , KRISHANT THARUN (110817104033) ”**, held on 21.04.2021

**INTERNAL EXAMINER EXTERNAL EXAMINER**

# 

# ABSTRACT

* Multi parking-site is a digital parking site, which was works in cloud. All

the required information should be get from the vehicle parker for example

name, phone no, vehicle no to ‘check in’, then all information will be stored in

database and generate a unique id to identify their vehicle when the customer

returns to ‘check out’, we should get and enter the id it will identifies the

vehicle and calculate the time and generates pay bill. The tools have used in

front of site is Html-is a structure of a webpage, CSS-used to design a layout

of a webpage, Bootstrap-it response at any devices, Java script- is used to

popup window. The tools have used in backend of site is Php- Is used to store

data from the front-end to back-end , Postgres SQL- It has all the data.

## TABLE OF CONTENTS

**PAGENO**

[ABSTRACT iv](#_Toc58709)

[LIST OF TABLES vii](#_Toc58710)

[LIST OF FIGURES viii](#_Toc58711)

[LIST OF SYMBOLS, ABBREVIATIONS AND DEFINITIONS ix](#_Toc58712)

#### 1. INTRODUCTION 1

1.1 System overview 2

1.2 Objective 2

1.3 System Study 2

1.4 Proposed system 5

1.4 Organization of the Project 5

#### 2. SOFTWARE REQUIREMENTS SPECIFICATION 7

2.1 Over all Description 8

2.2 External Interface Requirements 9

2.3 System Features 10

2.3.1 Acquired Image 10

2.3.2 Image processing 10

2.4 Other Non-Functional Requirements 10

#### 3. SOFTWARE DESIGN 12

3.1 System Architecture 13

3.2 Decomposition Description 22

3.2.1 Login 22

3.2.2 Image pre-processing 23

3.2.3 Number Detected 23

3.3 Data Design 23

3.4 Component Design 24

3.5 Human Interface Design 25

|  |  |
| --- | --- |
| **4. IMPLEMENTATION** | **28** |
| **5. TEST PLAN AND TESTING** | **50** |
|  |  |
| **6. RESULTS** | **55** |
| **7. CONCLUSION AND FUTURE WORK** | **57** |
| **APPENDIX** | **59** |
| **REFERENCES** | **63** |

# LIST OF TABLES

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Table.No** | **Table Name** | **Page No** |
| 1 | 5.1 | Unit Testing for Loading an RGB image | 18 |
| 2 | 5.2 | Unit Testing for Grayscale  conversion | 18 |
| 3 | 5.3 | Unit Testing for binarization | 19 |
| 4 | 5.4 | Unit Testing for Edge detection | 53 |
| 5 | 5.5 | Unit Testing for Plate region  extraction | 44 |
| 6 | 5.6 | UnitTesting for character  segmentation | 44 |
| 7 | 5.7 | Unit Testing for Template  matching | 44 |
| 8 | 5.8 | Execution of integration testing | 51 |
| 9 | 5.9 | Validation Testing | 51 |
| 10 | 5.10 | System Testing | 51 |

# LIST OF FIGURES

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Figure No** | **Figure Name** | **Page**  **No** |
| 1 | 3.1 | System Architecture | 13 |
| 2 | 3.2 | Use Case Diagram | 14 |
| 3 | 3.3 | Class Diagram | 15 |
| 5 | 3.5 | Sequence Diagram | 16 |
| 6 | 3.6 | Flow Diagram | 17 |
| 7 | 3.7 | Data Flow Diagram | 20 |
| 8 | 3.8 | Check visual studio Installation | 26 |
| 9 | 3.9 | User Screen to number plate detection | 26 |
| 10 | 3.10 | Check NPD are Started | 27 |
| 11 | A1 | Run NPD in Local Host | 59 |
| 12 | A2 | Browse User car image | 59 |
| 13 | A3 | Viewing the Inserted Data | 60 |
| 14 | A4 | Car Details in Output data | 60 |
| 15 | A5 | Multi Analysis-Top5 countries | 61 |

# LIST OF SYMBOLS, ABBREVIATIONS AND DEFINITIONS

### SYMBOLS

: External Entity



: Decision

: Flow of Data

: Database

: User/Actor

: Use case

: State

: Initial state

: End state

### ABBREVIATIONS

**UML** : Unified Modelling Language

**DFD** : Data Flow Diagram

**SSH** : Secure Shell

**SRS**: Software Requirement Specification

**API**: Application Programming Interface

**OCR** : optical character recognition

**NPIA** : National Policing Improvement Agency

**DEFINITIONS**

### Number plate extraction

An image-processing technology and an important field of research that identifies vehicle by their number plates in which the number plate information is extracted from vehicle's image or from sequence of images without direct human intervention.

### Character segmentation

Character segmentation is an operation that seeks to decompose an image of a sequence of characters into sub images of individual symbols. It is one of the decision processes in a system for optical character recognition (OCR).

### Template matching

Template matchingis a technique in digital image processing for finding small parts of an image which matcha template image. It can be used in manufacturing as a part of quality control, a way to navigate a mobile robot, or as a way to detect edges in images.

**CHAPTER 1 INTRODUCTION**

## 1. INTRODUCTION

### 1.1 SYSTEM OVERVIEW

We also demand a comfortable traveling life, be it private or public vehicle. With increasing number of vehicles every day, it is very difficult to keep track of every vehicle manually to keep check on law enforcement, traffic control, stolen cars etc. Using OCR technology, we can automate the manual work of noting down the license plate number. This system first will capture the image of the car, then it will pass on the image to OCR software which firstly recognizes the location of license plate in the image and then extracts the license plate from it. After extracting the license plate, we will do a number of image processing steps to enhance the image to get better results later on. Then we will perform character segmentation so that we can recognize each character individually. After getting the segmented character, we will recognize the characters using CNN ( Convolutional Neural Networks) which is trained on large number of data sets.

### 1.2 OBJECTIVE

The objective of this Number Plate detection system is to provide accurate vehicle registration number services on a continuous and free basis. The project is to find the most efficient way to recognize the registration information from the digital image (obtained from the camera).

### 1.3 SYSTEM STUDY

#### 1.3.1 Existing System

In existing system we consider that the high performance fibre optic sensors are used for detection of moving vehicles. A typical installation consists of an interface device with transmitter (LED), receiver (photo detector), and light guide connection cable (feeder) and fiber optic sensor. As the vehicle passes over the sensors there is a change in the signal levels obtained from the sensors. The output signals from the fiber optic sensors are fed into a signal processing and data evaluation unit which comprises of the algorithm, which computes axle count, axle spacing, vehicle lengths and vehicle classes based on time, distance formula, and amount of micro bending.

An IR curtain basically consists of an infra-red transmitter and receiver. These curtains create a clear profile of the vehicle as it passes through it. However the entire profile of the vehicle cannot be obtained by using just one strip of IR curtain due to varying speed of the vehicle that passes the gate. Thus it is important to know the speed of the vehicle. Using the distance between the curtains and the time, we calculate the speed of the vehicle. With the speed of the vehicle known and the frequency of pulses known we can determine the correct profile of the vehicle

#### 1.3.2 Literature Survey

**1.3.2.1 Car plate character extraction under complicated environment**  In this paper gives some of the trivial method used in the field of character recognition using various methods that made the system unreliable and time complex.

**1.3.2.2 Automatic License plate Recognition Using Artificial Neural**

### Network

As per the paper the proposed algorithm used feature extraction to extract the license plate from the given image. In the last stage, ANN was used for recognizing extracted characters.

#### 1.3.2.3 Extraction of License Plate Region in Automatic License Plate Recognition

The author of this paper has explained mathematical morphology concept was introduced for extracting plate region from the input image. Segmentation of license plate was done using digital image labelling and character recognition was done using template matching.

#### 1.3.2.4 An Efficient Method for Indian Vehicle License Plate Extraction and Character Segmentation

As per the paper edge detection algorithm and vertical projection method was used for extracting the plate region. For segmentation, there were several steps of filtering, thinning, vertical and horizontal projection.

#### 1.3.2.5 Automatic Number Plate Recognition (ANPR) System for Indian conditions

As per the paper the system which was designed for Indian plates used feature based number plate localization for locating the license plate from image and for character segmentation image Scissoring technique was used and statistical feature extraction was used for character recognition.

#### 1.3.2.6 Automatic License Plate Location and Recognition Based on Feature Salience

As per the paper for locating the license plate within the image, salient features were used. Feature projection was used to segment the characters in the license plate.

### 1.4 Proposed System

Input to the system is a vehicle image that has been acquired through image acquisition device and output is the editable form of license number. The system consists of the following main steps:

1. Acquired Input Image
2. Image Pre-processing  RGB to gray scale conversion

* Noise removal by Iterative Bilateral Filtering
* Image Binarization
  1. License plate extraction using Sobel’s edge detection algorithm/Smearing

algorithm

* 1. Character Segmentation
  2. Recognition of License number using convolution neural network

### 1.5 ORGANIZATION OF THE REPORT

The Chapter 1 presents system study. The goal tended to be attained in the project is explained in objectives. The problem description tells the need for the system with the advantage of proposed system over existing system.

The Chapter 2 explains the system requirement for both, by feature and by functional hierarchy. The Chapter 3 describes the System Design which includes the decomposition description, dependency description and data design of modules

The Chapter 4 describes the implementation of the project, which includes the modules and components used in this project. The Chapter 5 deals with the test plan and testing of the project. The Chapter 6 describes the results of the implementations. The Chapter 7 contains the conclusion of the work done and also the extension of the work.

## CHAPTER 2

**SOFTWARE REQUIREMENTS SPECIFICATION**

### 2. SOFTWARE REQUIREMENTS SPECIFICATION

#### 2.1 OVERALL DESCRIPTION

##### 2.1.1 Product Perspective

The product is developed in a java platform to provide efficient travel route to the travellers using sensor based technology. The location tracking sensors first tracks the exact location of the user, then the efficient route to the destined location is determined using distance vector routing and the determined route will be displayed to the user such that the user can reach any destination efficiently.

##### 2.1.2 Product Function

This system helps to keep track of incoming and outgoing traffic.. This system helps to Environmental zones need categorization of traffic. This system helps to around airports the traffic is monitored to detect patterns as an antiterrorism measure . We had used an effective algorithm which will provide accurate result.

##### 2.1.3 User Classes and Characteristics

1. Load Images : The input image is loaded from the browse option from the list of images from the user.
2. Image Recognition: This image will then be processed to generate the number plate in string form as an output after recognition

##### 2.1.4 Operating Environment Software Requirements

* Operating System : Windows
* Front End : html , java , c ,c++.
* Back End : php , psql.

##### 2.1.5 Design and Implementation Constraints

Vehicle number plate recognition is divided into several steps including number plate extraction, image region which contains a number plate, character segmentation, and character recognition. Generally, in order to recognize a vehicle number plate, the region of the number plate should be extracted from a vehicle image. Accurate detection of the plate region is essential process to go over to the step of character recognition. There are two major methods to extract number plate region,

* Edge Detection
* Finding Rectangles in a Vehicle Image.

#### 2.2 EXTERNAL INTERFACE REQUIREMENTS

##### 2.2.1 User Interfaces

This contains a user input window where in the input image of vehicle whose number plate recognition is to be done is specified. The input image is loaded from the browse option from the list of images. This image will then be processed to generate the number plate in string form as an output after recognition.

##### 2.2.2 Software Interfaces

The GUI of this application is developed using Microsoft Visual Studio libraries and C# code.

**2.2.3 Hardware Interfaces**

A computer that has enough hard disk space to boot a windows operating system and enough processor speed to enable its function normally. A standard QWERTY keyboard, an optical mouse and display screen.

##### 2.2.4 Communication interface

This application could connect to any SQL Server database server provided all the connection string such as host name, port no, database name and username and password are available.

However, database connectivity is out of our scope. The system screen will act as communicating interface between the system and the user.

#### 2.3 SYSTEM FEATURES

##### 2.3.1 Acquired Image

This allows the user to acquire the image of the vehicle with license plate which in our case is taken from an already existing set of images been acquired with the help of various devices**.**

##### 2.3.2 Image Pre-Processing

Image pre-processing is an important step in any image analyzing system. Without a proper pre-processing, the recognition will be ineffective or may give improper results in later stages. The main motive of pre-processing is to enhance the quality of the image that will be processed for recognition. Various processes that we are going to apply are converting RGB image to grayscale, Grayscale, noise reduction and binarization of image.

#### 2.4 OTHER NON FUNCTIONAL REQUIREMENTS

##### 2.4.1 Performance Requirements

The product should be compatible with Microsoft Windows 7/8/10 and Linux and Ubuntu operating systems. The product report should be compatible with the Microsoft Word and Adobe Reader.

* The face recognition using MATLAB offers a rich library of functions for vector and matrix. Thus, it offers the quick prototyping.
* The elapsed time for the training process takes only a few seconds.
* The larger number of epochs improves the performance rate of the system.

##### 2.4.2 Safety Requirements

This system has no backup or recovery features. Hence the route searched by the user will not be saved in a database.

##### 2.4.3 Security Requirements

The Number plate detection as almost all the tollbooths employ cameras for security purposes, it was felt that the feasibility of a system using IP cameras should be tested.

##### 2.4.4 Software Quality Attribute

###### 2.4.4.1 User-friendliness

The proposed system will be user-friendly, designed to be easy to use through simple interface. The software could be used with a basic computer knowledge. The software is created by an easy look and feel concept.

###### 2.4.4.2 Portability

The software can run either on Microsoft windows operating systems or linux operating systems or Ubuntu operating systems.

###### 2.4.4.3 Maintainability

All code shall be fully documented. Each function shall be commented with pre and post conditions. All program files shall include comments concerning date of last change. The code should be modular, to permit future modification

**CHAPTER 3**

**SOFTWARE DESIGN**

## 3. SOFTWARE DESIGN

### 3.1 SYSTEM ARCHITECTURE

#### 3.1.1 Architectural Design

The Architecture diagram below mentioned explains the system

architectural design of Number plate detection

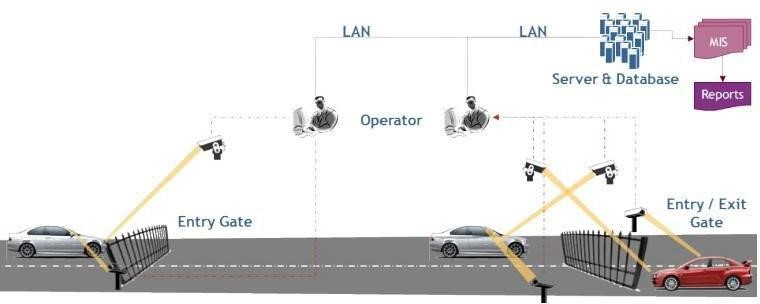


Fig 3.1 : Architectural diagram

#### 3.1.2 Use Case Diagram

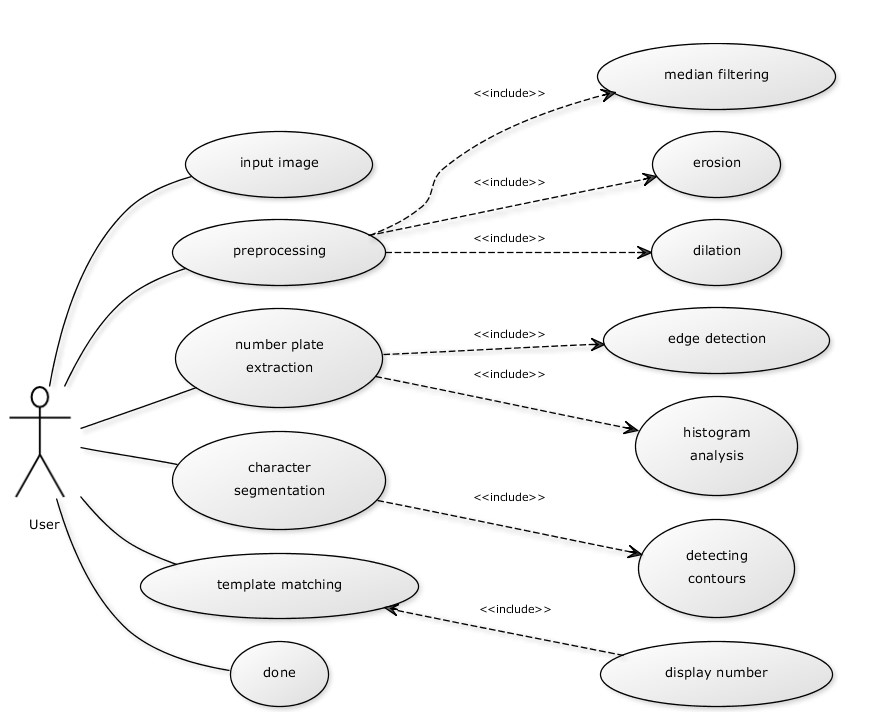


Fig 3.1.2 : Use Case diagram

Use case diagram are considered for high level requirement analysis of a system. It provides number of functions such as image upload, preprocessor, image processing, edge detection, template matching.

#### 3.1.3 Class Diagram

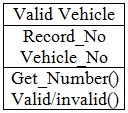
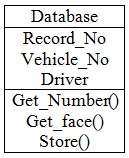
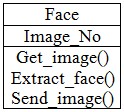
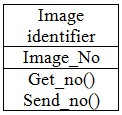
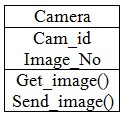


Fig 3.1.3 : Class diagram

A Class diagram is used for general conceptual modelling of the structure of the application and for detailed modelling translating the models into programming code. Class diagrams can also be used for data modelling. The classes in a class diagram represent both the main elements, interactions in the application and the classes to be programmed.

#### 3.1.4 Sequence Diagram

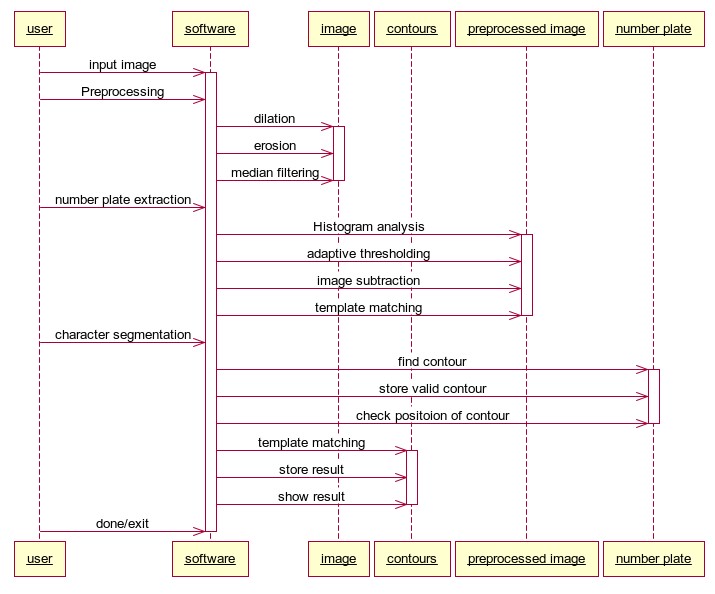
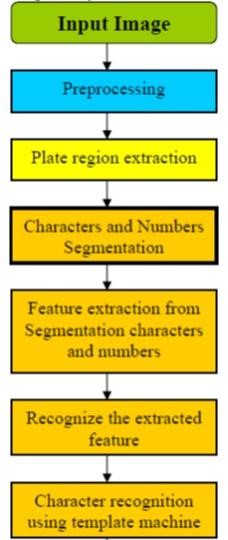


Fig 3.1.4 : Sequence diagram

A Sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. Sequence diagrams are sometimes called Event-trace diagrams, event scenarios and timing diagrams.

#### 3.1.5 Flow diagram



3.1.5 Flow Diagram

1. Number Plate Extraction.

2.Character Segmentation.

3.Template Matching

However, these steps are further divided into a series of other steps whose working is as followed:

#### 1. Loading an RGB image

The image whose number plate recognition is to be done is loaded.



Fig.3.3 Loading an image

#### 2. Grayscale conversion

This RGB image is converted to grayscale image



Fig.3.4 Grayscale Image

#### 3. Binarization

This image is then converted to binary using adaptive thresholding.



Fig.3.5 Binarized Image

#### 4. Edge detection

Dilated image is subtracted from the original image to get the edges.



Fig.3.6 Edge Detected Image

#### 5. Plate region extraction

Plate region is found out by passing a rectangular image over the previous using Match Template() function.



Fig.3.7 Extracted Plate Region

#### 6. Character segmentation

Characters are segmented from the number plate image which is then used for template matching.



Fig.3.8 Segmented Character

#### 7. Template matching

Segmented characters are template matched with the templates of each character and the number plate is identified as a string.

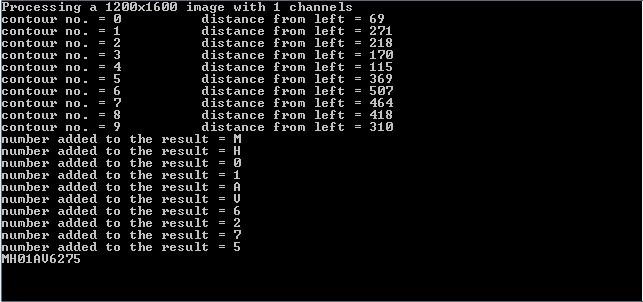


fig 3.9 Output in string form



Fig.3.10 Output after Template Matching

##### 3.1.6 Data Flow Diagram

Data flow diagram is a graphical representation of the flow of the data through an information system. The process starts by providing the source and destination location. The data is processed and the location of the user or the object is determined using the particle filtering algorithm. The extracted location can be displayed to the user in the form of maps with the help of the user segment.

###### 3.1.6.1 Data Flow Diagram : Level 0

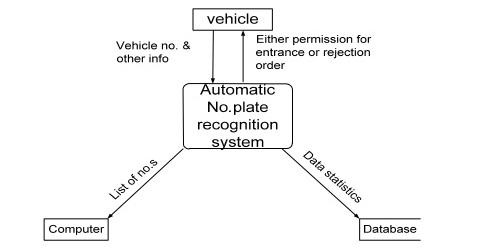
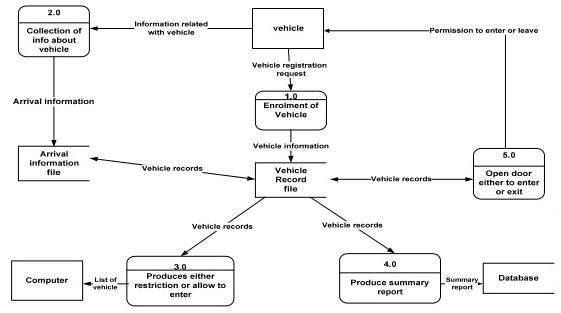


Fig 3.1.6.1 : Data flow diagram – level 0 **3.1.6.2 Data Flow Diagram : Level1**



### fig 3.1.6.2:Data flow diagram-level 1

#### 3.1.7 Entity Relationship Diagram

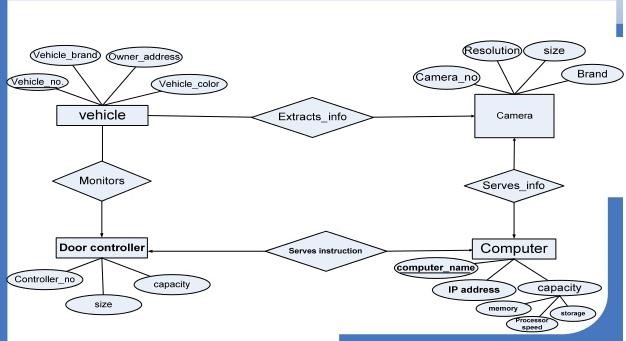


Fig 3.1.7 : Entity Relationship diagram

### 3.2 DECOMPOSITION DIAGRAM

#### 3.2.1 Module 1: Login

This module consists of the homepage of the project. It requires user id and password for the respective user. It checks whether the login id and the password was correct or not. It also verifies that the login was an authorized user who can access the cloud database. The new users have to register before login into the cloud to enable the access. The register dialog box consists of user name, user id, and password. Finally user can either register or login. These details can be stored and verifies for login to the cloud to access the cloud database

#### 3.2.2 Module 2 : Image Processing

The module involves the communication between the user and the system and image pre-processing. We can upload the image. The image whose number plate recognition is to be done is loaded. This image is converted to grayscale image. After that image is subtracted from the original image to get the edges.

#### 3.2.3 Module 3 : Number Detected

This module involves the number detected from segmented character. Characters are segmented from the number plate image which is then used for template matching. Segmented characters are template matched with the templates of each character and the number plate is identified as a string.

### 3.3 DATA DESIGN

#### 3.3.1 Data Description

The description about the login table is provided in order to store all the details about the user. The details are user name and password. In the registration it contains further more detail for the user.

#### 3.3.2 'Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Constrains** | **Description** |
| User Name | Varchar(45) | Primary Key | Login User Name |
| Password | Varchar(45) | Primary key | Login Password |

Table

#### 3.3.3 Register

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Constraints** | **Description** |
| User Name | Varchar(45) | Primary Key | Login User Name |
| Password | Varchar(45) | Primary key | Login password |
| Email Id | Varchar(45) | Primary key | Email Id |

Table

### 3.4 COMPONENT DESIGN

The major component involved in the project is described. The user interacts with the interface by providing the keywords.

#### 3.4.1 Register

The new users have to register before login to the cloud to access the data. The register dialog box consists of user name, password and the email id. Finally user can either register or login.

START

PRINT User\_Name, Password, Email\_Id;

GET User\_Name, Password, Email\_Id;

IF(User\_Name|| Password|| Email\_Id==NULL)

{ SHOW “Insufficient Data” }

ELSE

{ DISPLAY “Registered Successfully” } END

#### 3.4.2 Uploading the Image

The server will check the existence for the tag in the cloud database. Then the encrypted block image is stored in the database.

START

COMPARE Tag in Memory;

IF( Tag is not Present)

THEN

STORE the image;

ELSE

UPDATE Meta\_Data;

DISPLAY"ImageisuploadedSuccess"

#### 3.5 HUMAN INTERFACE DESIGN

##### 3.5.1 Overview of User Interface

The user interface of this includes a search page where the user need to provide the source location and the destination location in the respective text boxes and then select the search button provided below. The user will be given an option to provide either the source location in the text box or provide permission to automatically access the source location of the user

|  |  |
| --- | --- |
| **Purpose** | To find the vehicle plate number |
| **Inputs** | Image Source and Destination |
| **Processing** | Identifies the Number to the specified Vehicle. |
| **Outputs** | Displays the correct number and description of vehicle. |

Table 3.2 : Overview of User Interface

##### 3.5.2 Screen Image



Home page

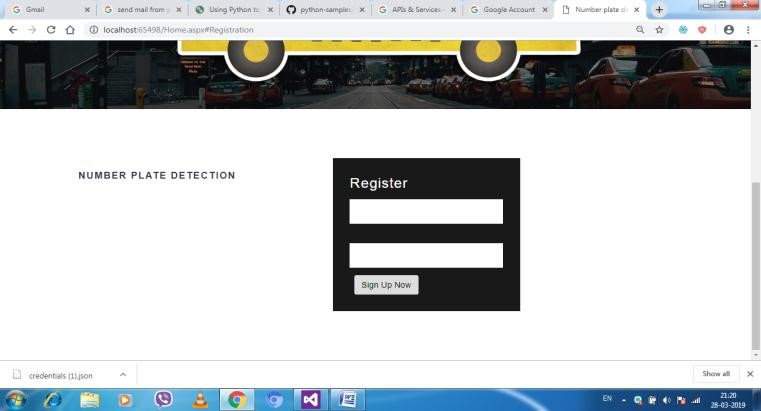


Fig. 3.5.2 Screen Image-Registration

##### 3.5.3 User Login

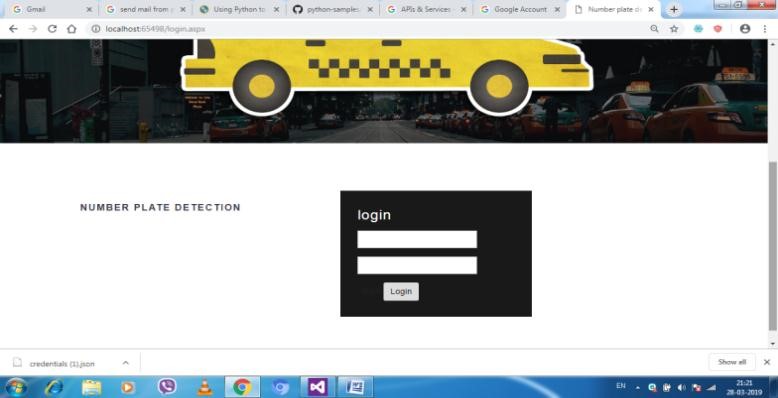
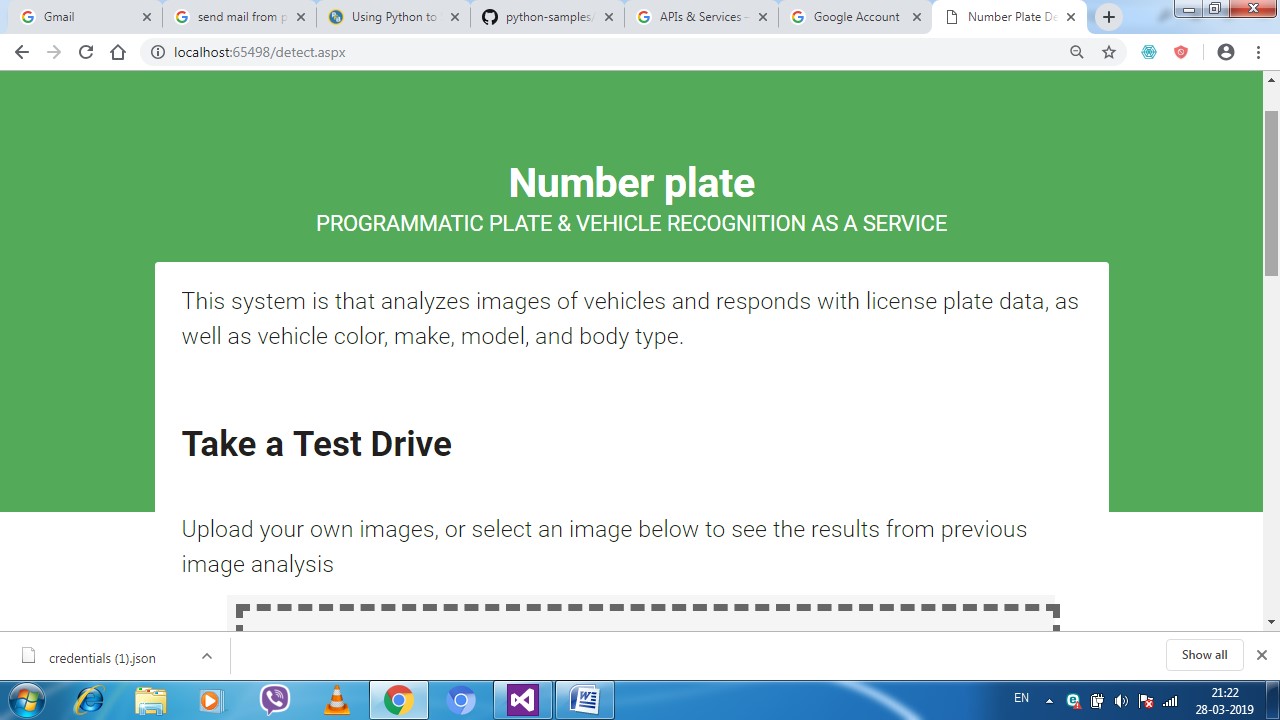


Fig. 3.5.3: Screen Image-Login

##### 3.5.4 Image Upload

 fig 3.5.4 Image upload

#### 3.6 Screen Actions

1. **Login**- The user want to login to the cloud before accessing the data on the cloud. It requires user id and password for login into the cloud.
2. **Register**- The user will register if he/she to create a new account in the cloud then only the user can upload the data in the cloud.
3. **Reset**- The reset is used to clear all the attributes in the text box.
4. **Home**- it is used to pass the query to perform the action in the database once after login into the website.
5. **Logout-** After the complete of the process the user need to log out their ID in order to prevent stealing of the user ideal.

3.5:

**CHAPTER 4**

**IMPLEMENTATION**

## 4. IMPLEMENTATION

The system design is targeted towards the implementation of the project.

This project is implementation using dot net. The implementation is dividing The task sheet divides the entire project into smaller and manageable components. This helps in planning, organizing and controlling the project and various activities involved in the project. It makes more systematic and the flow of task is determined. It helps us in analyzing our project progress as we track tasks in our project. The task sheet for this project is made in Microsoft Project 2003.The snapshot can be seen as shown.

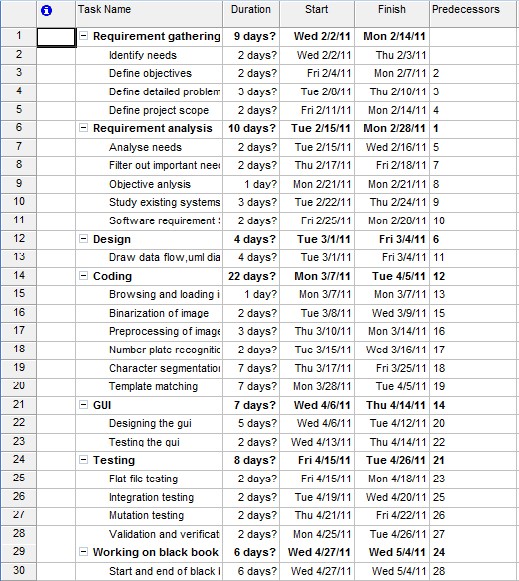


Fig. 4.1 : Task Sheet for Gantt Chart

**4.1 GANTT CHARTS:**

A Gantt chart is a popular type of bar chart that illustrates the project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of the project under consideration. The Gantt charts also show the dependency relationship between the activities. Thus a Gantt chart is a graphical representation of the duration of the task against the progression of time. It is a useful tool for planning and scheduling projects.

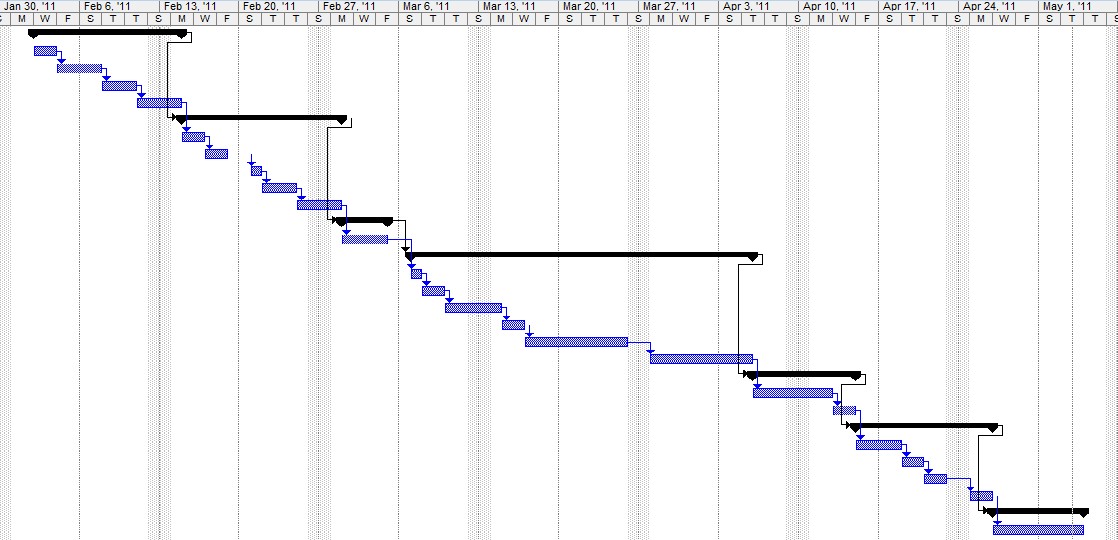


Fig 4.1. : Gantt Chart

**4.2** **WORK BREAKDOWN STRUCTURE:**

A complex project is made manageable by first breaking it into individual components in a hierarchical structure, known as the work breakdown structure, or WBS. Such a structure defines tasks that can be completed independently of the other tasks, facilitating resource allocation, assignment of responsibilities, and measurement and control of the project.

A Work Break down structure is a result oriented family tree that captures all the work of a project in an organized way. It is often portrayed graphically as a hierarchical tree; however, it can also be a tabular list of element categories and task or the intended tasks list that appears in a Gantt chart schedule.

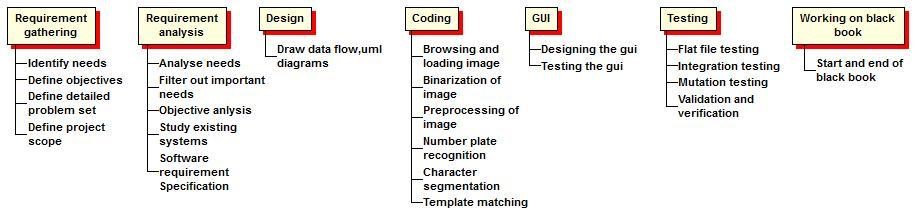


Fig 4.2 : Work Breakdown Structure

### 4.3 Code with reference to design with proper comments and brief description

**4.3.1 Code histogram** public void DrawHistogram(long[] Values)

{ myValues = new long[Values.Length]; Values.CopyTo(myValues,0); myMaxValue = getMaxim(myValues); myIsDrawing = true; ComputeXYUnitValues();

this.Refresh();

} private void ComputeXYUnitValues()

{ myYUnit = (float) (this.Height - (2 \* myOffset)) / myMaxValue; myXUnit = (float) (this.Width - (2 \* myOffset)) / (myValues.Length-1);

} if (myIsDrawing)

{

Graphics g = e.Graphics;

Pen myPen = new Pen(new SolidBrush(myColor),myXUnit);

//The width of the pen is given by the XUnit for the control.

for (inti=0;i<myValues.Length;i++)

{

//We draw each line

g.DrawLine(myPen,newPointF(myOffset + (i\*myXUnit), this.Height - myOffset),new PointF(myOffset + (i\*myXUnit), this.Height - myOffset - myValues[i] \* myYUnit));

//We plot the coresponding index for the maximum value.

if (myValues[i]==myMaxValue)

{

SizeFmySize = g.MeasureString(i.ToString(),myFont);

g.DrawString(i.ToString(),myFont, new SolidBrush(myColor), new PointF(myOffset + (i\*myXUnit) - (mySize.Width/2), this.Height - myFont.Height),

System.Drawing.StringFormat.GenericDefault);

}

}

//We draw the indexes for 0 and for

//the length of the array beeing plotted

g.DrawString("0",myFont, new SolidBrush(myColor), new PointF(myOffset,this.Height - myFont.Height),

System.Drawing.StringFormat.GenericDefault);

g.DrawString((myValues.Length-1).ToString(),myFont, new SolidBrush(myColor), new PointF(myOffset + (myValues.Length \* myXUnit)

- g.MeasureString((myValues.Length-1).ToString(),myFont).Width, this.Height - myFont.Height),

System.Drawing.StringFormat.GenericDefault); //We draw a rectangle surrounding the control.

g.DrawRectangle(newSystem.Drawing.Pen(new SolidBrush(Color.Black),1),0,0, this.Width-1,this.Height-1);

}

#### 4.3.2 Code Edge Detection namespace Image

{ public partial class FormImageEdge : Form

{ public FormImageEdge()

{

InitializeComponent();

} private void Form1\_Load(object sender, EventArgs e)

{

Bitmap image = new Bitmap(Image.Properties.Resources.sassafras); pictureBoxPreview.Image = image;

} private void buttonTest\_Click(object sender, EventArgs e)

{

OpenFileDialogOpenFileDialog = new OpenFileDialog();

OpenFileDialog.Title = "Select image ";

OpenFileDialog.Filter = "Png Images(\*.png)|\*.png|Jpeg Images(\*.jpg)|\*.jpg"; OpenFileDialog.Filter += "|Bitmap Images(\*.bmp)|\*.bmp"; if(OpenFileDialog.ShowDialog() ==System.Windows.Forms.DialogResult.OK)

{

StreamReaderstreamReader = new StreamReader(OpenFileDialog.FileName); Bitmap image = new Bitmap(streamReader.BaseStream); streamReader.Close(); pictureBoxPreview.Image = image;

} } private void buttonApplyFilters\_Click(object sender, EventArgs e) {

Try { if(listBoxXFilter.SelectedItem.ToString().Length>0&& listBoxYFilter.SelectedItem.ToString().Length > 0)

{ filter(listBoxXFilter.SelectedItem.ToString(), listBoxYFilter.SelectedItem.ToString());

ConvertToXYCoord(pictureBoxResult);

}

Else

{ labelErrors.Text = "2 filters must be selected";

} } catch (Exception)

{ labelErrors.Text = "2 filters must be selected";

} } public void filter(string xfilter, string yfilter)

{ double[,] xFilterMatrix; double[,] yFilterMatrix; switch (xfilter) { case "Laplacian3x3":

xFilterMatrix = FilterMatrix.Laplacian3x3; break; case "Laplacian5x5":

xFilterMatrix = FilterMatrix.Laplacian5x5; break; case "LaplacianOfGaussian":

xFilterMatrix = FilterMatrix.LaplacianOfGaussian; break; case "Gaussian3x3":

xFilterMatrix = FilterMatrix.Gaussian3x3; break; case "Gaussian5x5Type1":

xFilterMatrix = FilterMatrix.Gaussian5x5Type1; break; case "Gaussian5x5Type2":

xFilterMatrix = FilterMatrix.Gaussian5x5Type2; break; case "Sobel3x3Horizontal":

xFilterMatrix = FilterMatrix.Sobel3x3Horizontal; break; case "Sobel3x3Vertical":

xFilterMatrix = FilterMatrix.Sobel3x3Vertical; break; case "Prewitt3x3Horizontal":

xFilterMatrix = FilterMatrix.Prewitt3x3Horizontal; break; case "Prewitt3x3Vertical":

xFilterMatrix = FilterMatrix.Prewitt3x3Vertical; break; case "Kirsch3x3Horizontal":

xFilterMatrix = FilterMatrix.Kirsch3x3Horizontal; break; case "Kirsch3x3Vertical":

xFilterMatrix = FilterMatrix.Kirsch3x3Vertical; break; default: xFilterMatrix = FilterMatrix.Laplacian3x3; break; } switch (yfilter) { case "Laplacian3x3":

yFilterMatrix = FilterMatrix.Laplacian3x3; break; case "Laplacian5x5":

yFilterMatrix = FilterMatrix.Laplacian5x5; break; case "LaplacianOfGaussian":

yFilterMatrix = FilterMatrix.LaplacianOfGaussian; break;

case "Gaussian3x3":

yFilterMatrix = FilterMatrix.Gaussian3x3; break; case "Gaussian5x5Type1":

yFilterMatrix = FilterMatrix.Gaussian5x5Type1; break; case "Gaussian5x5Type2":

yFilterMatrix = FilterMatrix.Gaussian5x5Type2; break; case "Sobel3x3Horizontal":

yFilterMatrix = FilterMatrix.Sobel3x3Horizontal; break; case "Sobel3x3Vertical":

yFilterMatrix = FilterMatrix.Sobel3x3Vertical; break; case "Prewitt3x3Horizontal":

yFilterMatrix = FilterMatrix.Prewitt3x3Horizontal; break; case "Prewitt3x3Vertical":

yFilterMatrix = FilterMatrix.Prewitt3x3Vertical; break; case "Kirsch3x3Horizontal":

yFilterMatrix = FilterMatrix.Kirsch3x3Horizontal; break; case "Kirsch3x3Vertical":

yFilterMatrix = FilterMatrix.Kirsch3x3Vertical; break; default: yFilterMatrix = FilterMatrix.Laplacian3x3; break; } if (pictureBoxPreview.Image.Size.Height> 0)

{

Bitmap newbitmap = new Bitmap(pictureBoxPreview.Image); BitmapDatanewbitmapData = new BitmapData(); newbitmapData = newbitmap.LockBits(new Rectangle(0, 0, newbitmap.Width, newbitmap.Height),

ImageLockMode.ReadOnly, PixelFormat.Format32bppPArgb); byte[] pixelbuff = new byte[newbitmapData.Stride \* newbitmapData.Height]; byte[] resultbuff = new byte[newbitmapData.Stride \* newbitmapData.Height]; Marshal.Copy(newbitmapData.Scan0, pixelbuff, 0, pixelbuff.Length); newbitmap.UnlockBits(newbitmapData); double blue = 0.0;

double green = 0.0; double red = 0.0;

//intfilterWidth = filterMatrix.GetLength(1);

//intfilterHeight = filterMatrix.GetLength(0);

//intfilterOffset = (filterWidth - 1) / 2;

//intcalcOffset = 0; //intbyteOffset = 0; double blueX = 0.0; double greenX = 0.0; double redX = 0.0; double blueY = 0.0; double greenY = 0.0; double redY = 0.0; double blueTotal = 0.0; double greenTotal = 0.0; double redTotal = 0.0; intfilterOffset = 1; intcalcOffset = 0; intbyteOffset = 0; for (intoffsetY = filterOffset; offsetY< newbitmap.Height - filterOffset; offsetY++)

{ for (intoffsetX = filterOffset; offsetX< newbitmap.Width - filterOffset; offsetX++)

{ blueX = greenX = redX = 0; blueY = greenY = redY = 0; blueTotal = greenTotal = redTotal = 0.0; byteOffset = offsetY \*newbitmapData.Stride +offsetX \* 4; for (intfilterY = -filterOffset; filterY<= filterOffset; filterY++)

{ for (intfilterX = -filterOffset; filterX<= filterOffset; filterX++)

{ calcOffset = byteOffset +(filterX \* 4) +(filterY \* newbitmapData.Stride); blueX += (double)(pixelbuff[calcOffset]) \*xFilterMatrix[filterY + filterOffset,filterX + filterOffset];

greenX += (double)(pixelbuff[calcOffset + 1]) \*xFilterMatrix[filterY + filterOffset,filterX + filterOffset];

redX += (double)(pixelbuff[calcOffset + 2]) \*xFilterMatrix[filterY + filterOffset,filterX + filterOffset];

blueY += (double)(pixelbuff[calcOffset]) \*yFilterMatrix[filterY + filterOffset,filterX + filterOffset];

greenY += (double)(pixelbuff[calcOffset + 1]) \*yFilterMatrix[filterY + filterOffset,filterX + filterOffset];

redY += (double)(pixelbuff[calcOffset + 2]) \*yFilterMatrix[filterY + filterOffset,filterX + filterOffset];

}

}

//blueTotal = Math.Sqrt((blueX \* blueX) + (blueY \* blueY)); blueTotal = 0; greenTotal = Math.Sqrt((greenX \* greenX) + (greenY \* greenY));

//redTotal = Math.Sqrt((redX \* redX) + (redY \* redY)); redTotal = 0; if (blueTotal> 255)

{ blueTotal = 255;

} else if (blueTotal< 0) { blueTotal = 0; } if (greenTotal> 255) { greenTotal = 255; } else if (greenTotal< 0) { greenTotal = 0; } try { if (greenTotal< Convert.ToInt32(trackBarThreshold.Value))

{ greenTotal = 0;

}

Else { greenTotal = 255;

} } catch (Exception)

{ throw; } if (redTotal> 255) { redTotal = 255; } else if (redTotal< 0) { redTotal = 0; resultbuff[byteOffset] = (byte)(blueTotal); resultbuff[byteOffset + 1] = (byte)(greenTotal); resultbuff[byteOffset + 2] = (byte)(redTotal); resultbuff[byteOffset + 3] = 255;

}

}

Bitmap resultbitmap = new Bitmap(newbitmap.Width, newbitmap.Height);

BitmapDataresultData = resultbitmap.LockBits(new Rectangle(0,

0,resultbitmap.Width, resultbitmap.Height),ImageLockMode.WriteOnly,

PixelFormat.Format32bppArgb);

Marshal.Copy(resultbuff, 0, resultData.Scan0, resultbuff.Length); resultbitmap.UnlockBits(resultData); pictureBoxResult.Image = resultbitmap;

} else { labelErrors.Text = "You must load an image";

}

} public void ConvertToXYCoord(PictureBoxpictureBoxelem)

{ string coord = ""; int width = pictureBoxelem.Image.Width; int height = pictureBoxelem.Image.Height;

System.Drawing.Size size = new System.Drawing.Size(width, height); Bitmap bitmapIMG = new Bitmap(pictureBoxResult.Image, width, height);

int x = 0; int y = 0; for (x = 0; x < width; x++)

{ for (y = 0; y < height; y++)

{

Color pixelColor = Color.FromArgb(bitmapIMG.GetPixel(x, y).ToArgb()); if (pixelColor.Name != "ff000000" &&pixelColor.Name != "0")

{ coord = coord + x.ToString() + "," + y.ToString() + "|";

}

}

}

textBoxData.Text = coord;

//chart1.Series.Add("plot");

//chart1.Series["plot"].ChartType =

System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Point;

////chartNew.Series["plot"].LabelAngle = 90;

////chartNew.ChartAreas[0].AxisX.Maximum = width;

////chartNew.ChartAreas[0].AxisY.Maximum = height;

//string[] topCoord = coord.Split('|');

//for (inti = 0; i<topCoord.Length; i++)

//{

// string[] bottomCoord = topCoord[i].Split(',');

// chart1.Series["plot"].Points.AddXY(Convert.ToDouble(bottomCoord[0]),

Convert.ToDouble(bottomCoord[1]));

//}

}

|  |  |  |
| --- | --- | --- |
| private void  MouseEventArgse)  {  } | pictureBoxPreview\_MouseDown(object | sender, |
| private void | pictureBoxPreview\_MouseMove(object | sender, |

MouseEventArgse)

{

//pictureBoxPreview.Refresh();

//Rectangle rect = new Rectangle(e.X, e.Y, 200, 250);

//using (Pen pen = new Pen(Color.Green, 5))

//{

// pictureBoxPreview.CreateGraphics().DrawRectangle(pen, rect);

//}

//textBoxPos.Text = e.X.ToString() + "," + e.Y.ToString();

} private void buttonSaveAs\_Click(object sender, EventArgs e)

{

//pictureBoxResult.Image.Save(Application.StartupPath + "\\Result.png",

ImageFormat.Png);

}

}

}

**CHAPTER 5**

**TEST PLAN AND TESTING**

## 5. TEST PLAN AND TESTING

Testing presents an interesting anomaly for the software engineer. During earlier software attempts to build a software from an abstract concept to a tangible product. Testing is the one step in the software process that could be viewed as destructive rather than constructive.

Testing is the process of analysing the system for any possible errors, and thereby providing solutions to correct them. This process is done y arriving at various test cases that are applied to the system and the resulting messages being noted. There are different types of testing namely,

* Unit Testing
* Integration Testing
* Validation Testing
* System Testing

### Testing Objectives

Testing is the process of process of executing a program with the intent of finding an error. A good test case is the one that has a high probability of finding an-yet-undiscovered error. A successful test case is one that discovers an-yet-undiscovered error. There are various types of testing among which unit testing, Integration testing and validation testing are discussed.

### 5.1 UNIT TESTING

This involves the test carried out on modules that make up the system. It is also called as program testing. The units in a system are the modules and routines that are assembled and integrated to perform a specific function.

### 5.2 INTEGRATION TESTING

When the whole system is split into different modules, data can be across and interface, one module can advertent effect another, sub functions were combined may be configured to unacceptable levels, global data structures present problem etc. Integration testing is a systematic technique for constructing the program structure, while at the same tie conducting test to uncover the errors associated with interfacing. In other words, testing is complete testing of the modules, which makes up the construct a product. All modules are combined in advance. The entire program is tested as a whole sets errors are encountered. Corrections are difficult because isolation of cause completed by the vast expanse of the entire program.

### 5.3 VALIDATION TESTING

Finally the validation test is carried out. Here the vulnerability of the project towards various inputs is tested. The behaviour of all those inputs on the system is checked. The details about the validation testing on our project are listed below.

### 5.4 SYSTEM TESTING

The process of performing various of tests on a system to explore functionality or to identify problems. System testing is usually required before and after a system is put in place. A series of s systematic procedures are referred to while testing is being performed. These procedures tell the tester how the system should perform and where common mistakes may be found. Testers usually try to break the system by entering data that may cause the system to malfunction or return incorrect information.

System testing is performed on the entire system in the context of a Functional

Requirement Specification (FRS) and a System Requirement Specification

(SRS). System testing is an investigatory testing phase, where the focus is to have almost a destructive attitude and tests mot only the design, but also the behaviour and even the experiments of the customer**.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Input** | **Actual Output** | **Expected Output** | | **Is actual output is same as**  **expected output?** |
| 1. | Process without  loading an image | nothing | It will show error  message – “please load a valid  image” | It will show error  message – “please load a valid  image” | | Yes |
| 2. | Load an image | .jpg image | Gray scale image is  loaded | Gray scale image | | Yes |
| 3. | Load an image  without number plate | .jpg image | It will show error  message  “number plate not present” | It will show error  message  “number plate not present” | | Yes |
| 4. | Load an image with noisy  number plate | .jpg image | It will show error  message  “number plate is noisy” | It will show error  message  “number plate is noisy” | | Yes |
| 5. | Load an image with language  other than english | .jpg image | It will show error message “templates cannot be matched” | | It will show error message “templates cannot be matched” | Yes |
| 6. | Load an image with number plate of  different font | .jpg image | It will show error message “templates cannot be matched” | | It will show error message “templates cannot be matched” | Yes |

Table 5.1 : System Testing

**CHAPTER 6**

**RESULTS**

## 6. RESULTS

Our proposed system gave an excellent results as far as the training and recognition of the character is concerned. The below table shows the accuracy maintained while recognizing the character.

|  |  |  |  |
| --- | --- | --- | --- |
| **RECOGNI TION**  **ACCURA**  **CY** | **LICENSE**  **PLATE**  **RECOGNITIO**  **N** | **CHARACTER**  **SEGMENTATION** | **CHARACTER RECOGNITION** |
| STAGES  OUTPUT/T  OTAL | 94/100 | 96/100 | 980/1000 |
| PERCENT  AGE | 94% | 96% | 98% |

**CHAPTER 7**

**CONCLUSION AND FUTURE WORK**

## 7. CONCLUSION AND FUTURE WORK

### 7.1 CONCLUSION

Automatic license plate recognition is a wide field which can be implemented using many different algorithms and techniques. Every method has its own advantages and disadvantages. Our proposed methodology initially does the pre=processing steps which includes BGB to gray scale conversion, noise removal and binarization of the image. After which the license plate is extracted using Sobel’s edge detection algorithms. Then the characters are segmented using horizontal scanning which is given as input to the CNN in order to recognize the character correctly. Training our system with the help of ANN made our system more reliable and efficient in order to recognize the characters correctly. Although we can see that so may algorithms have been implemented in various previous projects, in order to make a robust system for automatic license plate recognition, there are still many loop holes left in the system which can be filled in order to make the system more future-proof and reliable. Our project however works on the simple font styles which is being used normally on license plates of the cars as per the rules made by the governing bodies of traffic department. But in order to handle the cases where people don’t follow these rules, It can be handled in future projects being implemented in this field of license plate recognition. Some of the various fields which can be explored in this project are as follows.

* Car model recognition
* Multi-lingual character recognition
* Fancy character recognition etc.

**APPENDIX**

## APPENDIX

### A1 HOME PAGE

User can register or login .

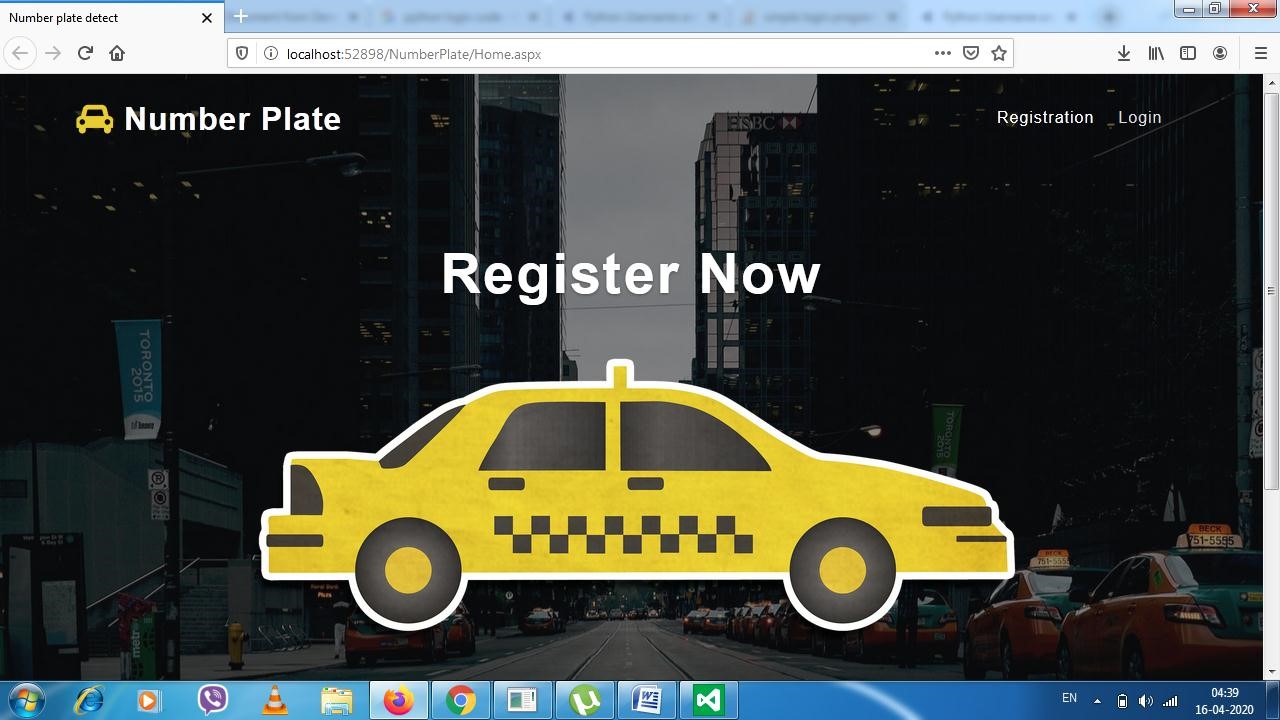


Fig A1 : Home Page

### A2 REGISTER

Initially the register with your credentials .

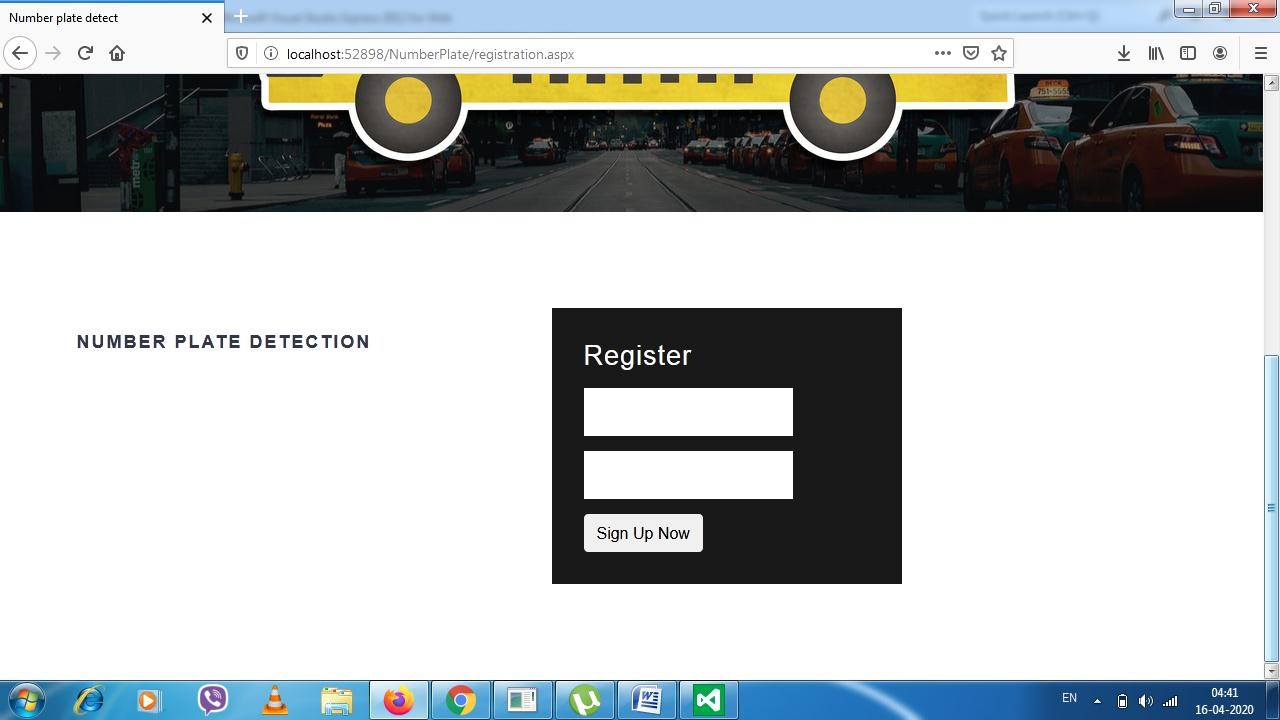
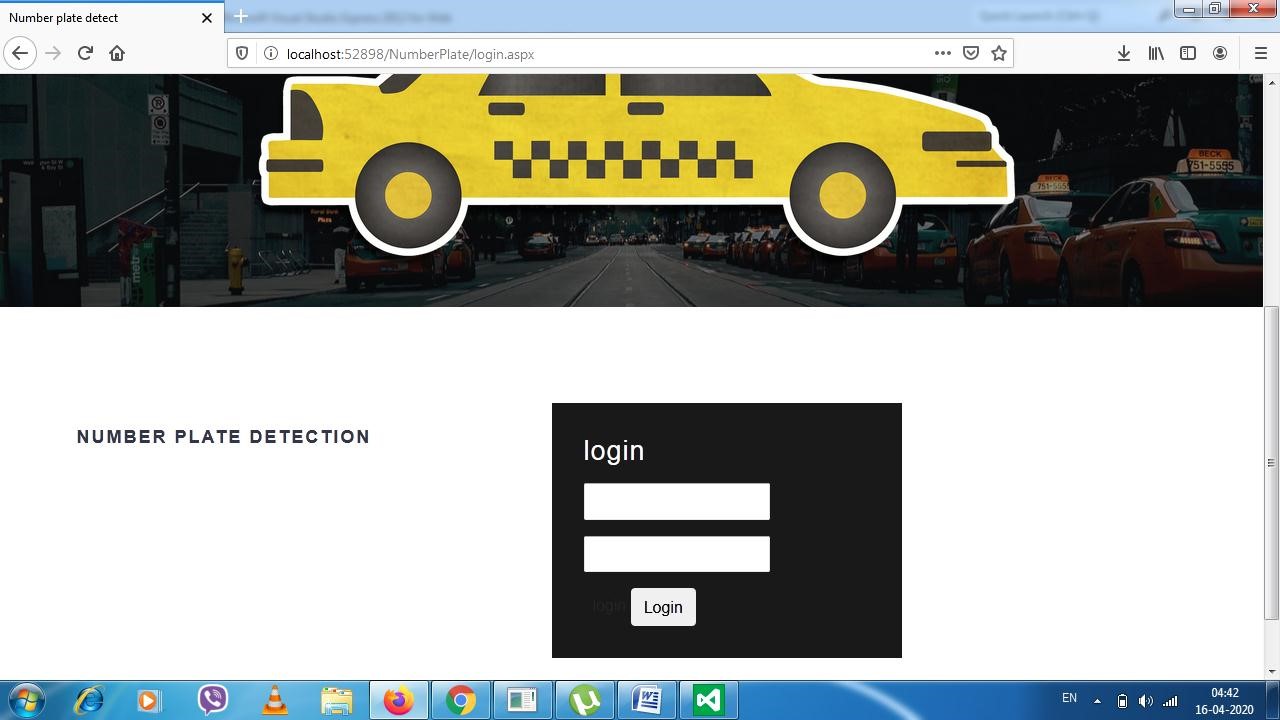


Fig A1 : Register

### A3 LOGIN



T

Fig A3 : Login page

### A4 Upload Image

The image is upload.

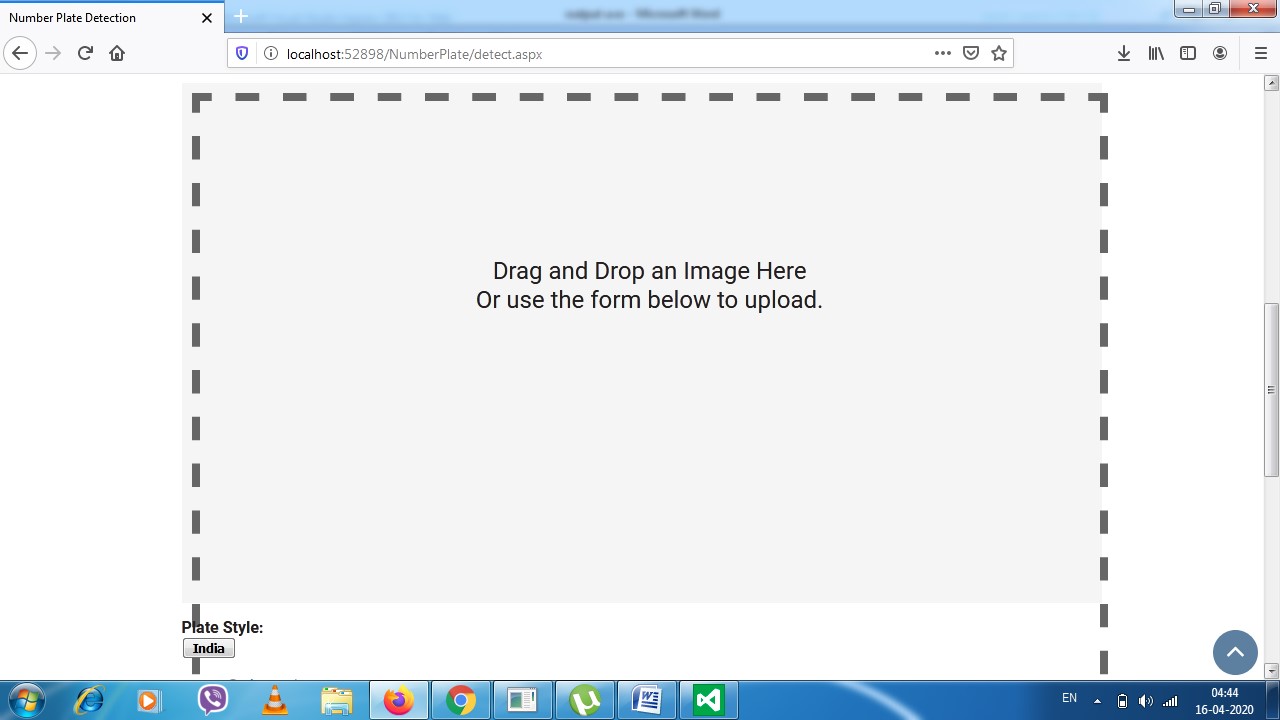


Fig A4 : Upload image

### A5 VEHICLE DETAILS

The license number and information about particular car.

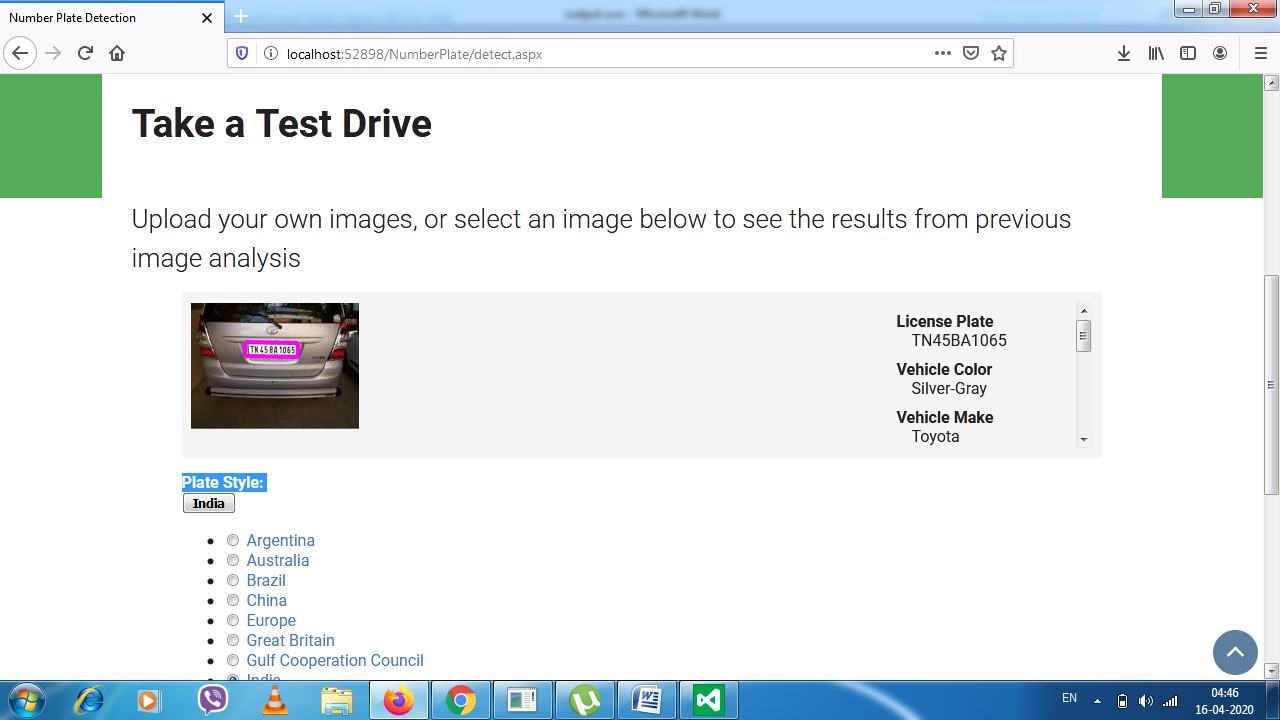


Fig A5 : Vehicle Details

**REFERENCES**

## REFERENCES

[1] Harish D. Kendre,Gaurav V. Talokar, Mohan Girhe, TejasPidkalwar, ***“The Automatic Number Plate Recognition System (ANPR)”***, INTERNATIONAL JOURNAL OF MATHEMATICS AND

COMPUTER RESEARCH, Volume 1 issue 3 April 2013 ISSN :2320-7167, pp. 99-102

[2]Amarjot Singh, KetanBacchuwar, and AkshayBhasin ***“A Survey of OCR Applications”***International Journal of Machine Learning and Computing, Vol. 2, No. 3; June 2012

[3]AniruddhPuranic , Deepak K. TUmadevi .V ***“Vehicle Number Plate Recognition System”*** A Literature Review and Implementation using Template Matching-