# Chapter 1
# INTRODUCTION

## 1.1 Background and Basics

Now a day, there is increase in traffic rule violation. For that the system currently used is not so efficient in detecting all the people who are breaking the rules. For one signal analysis there is a requirement of two people. Analysis of all the traffic signals is technically not possible, in this way, as the human intervention required is a lot, which is a part of concern.

Image for the camera at signals is given as information to the person checking for traffic violation in traffic control room and manually checks the registration number from number plate and enters the registration number of the vehicle manually and the challan for that vehicle is generated.

The system proposed by us involves automatic detection of vehicles that break the traffic rules at respective signals and registration number for every vehicle is recognized. The vehicle number detected is searched in the database for type of vehicle and owner’s information. This information is used to generate challan in the name of the person who owes the vehicle directly and instantly and send appropriate fine message to the owner.

## 1.2 Literature Survey

The work done in [1] Using Deep Learning and Applied the model of convolutional neural network(CNN), Recognized the license plate by converted object detection problem into a binary classification problem. The candidate regions are generated on the sliding window by using selective search algorithm, at last Support Vector Machine is used for classification.

In [2] This paper presents an approach which is based on characters of the number plate, for that an adaptive preprocessing method was used because not every time the illumination condition is constant, then segmentation of characters were done using vertical and horizontal projection of the extracted license plate. Finally, character recognition were done using K nearest neighbor classifier based on feature vector which was extracted from the boundary analysis of the character.

In paper [3], an automatic system for LP detection and recognition based on deep learning approach, which is divided into three parts: detection, segmentation, and character recognition. To detect an LP, many pretreatment steps should be made before applying the first Convolution Neural Network (CNN) model for the classification of plates / non-plates. Subsequently, we apply a few pre-processing steps to segment the LP and finally to recognize all the characters in upper case format (A-Z) and digits (0-9), using a second CNN model with 37 classes. The performance of the suggested system is tested on two datasets which contain images under various conditions, such as poor picture quality, image perspective distortion, bright day, night and complex environment.

## 1.3 Project Undertaken

### 1.3.1 Problem Definition

Detection of Vehicle Number Plate E-challan Generation on Rule Violation by Using Computer Vision.

### 1.3.2 Scope Statement

Purpose of our system is to automate e-challan generation when vehicles cross zebra crossing during traffic signal. The system is based on detection of the vehicles that have broken the rule, license plate detection of the vehicle breaking rule and effective e-challan generation. We will have a database server that has information of all the vehicle registered. After registration number is obtained database is searched for all owner information related to that number, after which e challan is generated in the name of that person who will receive a text message regarding the same. Above all, we hope to provide a smooth, easy and hassle-free system for the traffic authority.

## 1.4 Organization of Report

Project planning, management and the purpose of system to provide a complete solution for automatic challan generation is described in Chapter 2. Chapter 3 describes mathematical model, methodology along with analysis and design. Chapter 4 describes various test cases which are used for testing with their results.

Chapter 1 gives the Introduction to Background and Basics, Literature Survey conducted, about project to be undertaken, Problem definition and scope of the project.

Chapter 2 gives overview of project planning and management. It states the detailed Functional, non-functional requirements and other requirements. This project also specifies the Project process model used to develop this project.

Chapter 3 gives Analysis study and Designing of the project. It consists of Mathematical model and all UML diagrams specifying how to build the required system.

Chapter 4 gives the test cases identified for various types of testing.

# Chapter 2

# PROJECT PLANNING AND MANAGEMENT

## 2.1 Detail System Requirement Specification (SRS)

### 2.1.1 System Overview

The Product we are developing is supposed to be a project for the Government Officials. The system is a python software developed to automatically to detect rule violation at signals and generate a report regarding the very same. The system uses Computer Vision to capture image and system process the image required for registration number plate recognition.

The input to the system is the image captured by still frame camera when the signal at any junction goes red and later the image is per-processed to remove any noise if present. The processed image is use to identify all the number plates in the image and all those plates are used to recognize all registration number. The registration number is searched in database to get all user information to generate e-challan in the name of the user.

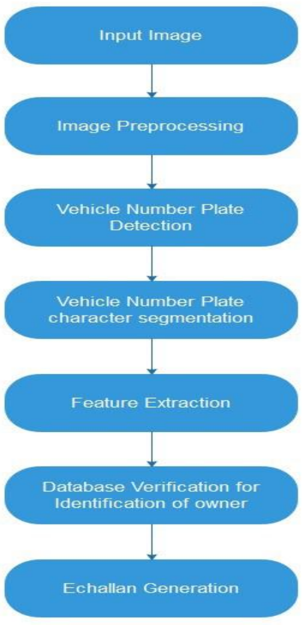

Input Image
Image Preprocessing
Vehicle Number Plate Detection
Vehicle Number Plate character segmentation
Feature Extraction
Database Verification for Identification of owner
Echallan Generation

**Figure. 2.1 System Overview**

### 2.1.2 Functional Requirements

The system being developed has two main features:

1) Detecting the vehicles that have violated the rule. 2) Generating e-challan for the vehicles that have broken the rule.

Detecting of rule violation

Description and Priority When the signal is red the vehicles that have violated the rule have to be detected, this is a high priority requirement.

Stimulus/Response Sequences

1) Image is taken when signal goes red.

2) The image taken is cropped from the zebra line.

3) Noise is filtered out of the image.

4) The vehicles in existing image have broken the rule.

5) Traffic authorities receive all the registration number of vehicles that have broken the rule.

Functional Requirements

REQ-1: A high resolution camera should be used to take the image, when signal goes red.

REQ-2: The image taken should be filtered to remove noise, and it should be cropped to detect vehicle.

Automatic E-challan Generation

Description and Priority Once the traffic authorities receive all the registration number, the system will automatically generate challan for all the vehicles, this is high priority.

Stimulus/Response Sequences

1) System will get all the registration number of vehicles that have violated the rule.

2) Using the government API for finding vehicle information, system will get owner information of all the registration numbers detected.

3) Once owner information is found, using way2SMS API message about rule violation and fine is recorded.

Functional Requirements

REQ-1: The registration number detected by system should be correctly detected by the system.

REQ-2: The government API for owner information should be running 24/7 and respond quickly for the system to be efficient.

REQ-3: The way2SMS API should be running 24/7 and send message to respective owner.

### 2.1.3 Non-Functional Requirements

Performance Requirement

All the vehicles that have violated the rule at particular signal have to be detected instantly and without any errors in normal climatic conditions is a particular hardware requirement that has to be meant.

Camera used should take image exactly when signal goes red. Image should be sent to system quickly and without any faults. System should recognize number correctly. Database should be running continuously to get vehicle information. The APIs necessary should be running constantly to send e-challan information.

Security Requirements

The user data obtained should be stored securely and should not be misused to generate wrong challans. The rules regarding the rules of API should not be violated.

Software Quality Attributes

The system should be robust as multiple request to process rule violation may occur at the same time in a particular region. The system must be reusable in different regions by making the system adapt to various different hardware interfaces. The system must be reliable and run 24/7 under any circumstances

### 2.1.4 Deployment Environment

Hardware Requirements

1. Processor speed of 2 GHz or above.
2. Hard Disk space of at least 20 GB.
3. Minimum 2GB RAM

Software Requirements

1. Python version 3 and above.
2. Windows 7 OR Linux Operating System.
3. Vehicle owner database verification.
4. API for message sending (Way2SMS)

### 2.1.5 External Interface Requirements

User Interfaces The traffic authority personal and administrators interact with the system through a web portal, an administrator should also be able to log in to the web-portal where he/she can administer the system.

Vehicle No: Submit

---

Vehicle No : XXXXXXXXXXX

Challan 1:

Challan No : XXXXX Date Time Fine: Rs. XXX

Challan 2:

Challan No : XXXXX Date Time Fine: Rs. XXX

Total Fine : Rs. XXX

---

After Clicking on Particular Challan:

Challan No : XXXXX

Date Time

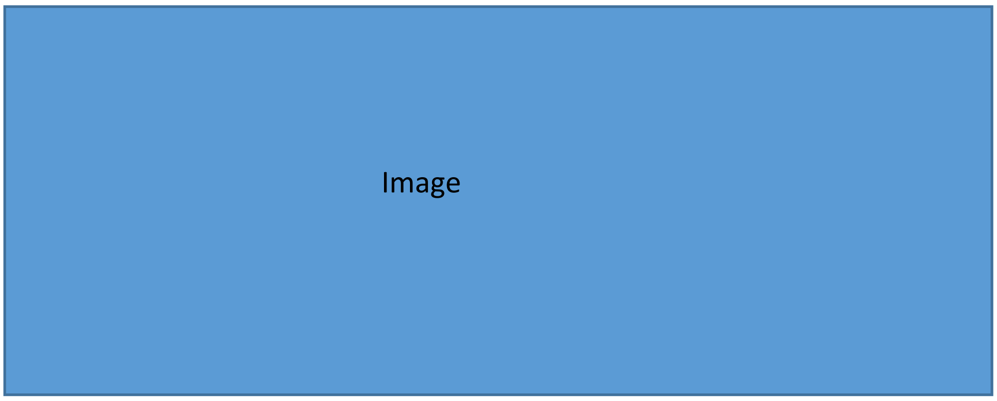
Fine: Rs. XXX

Vehicle No: XXXXXX

Mobile No : XXXXX

Place: XXXXXXX

Image of Rule Violation Proof:



2. Hardware Interfaces

Traffic Camera will be used for capturing Image.

3. Software Interfaces

| Software Used | Description |
|---|---|
| Operating System | We have chosen windows OS for its best support and user friendliness. |
| Python | To implement the project we have chosen python language for its more interactive support and its features. |
| Vehicle database | Database of vehicle |
| API for SMS | way2SMS |

Table 2.1 Software Interfaces

4. Communications Interfaces

Database of vehicle details is used to get owner information of the vehicle from its registration number

API for sending SMS(way2SMS) This API is used for sending SMS to owner regarding rule violation and fine generated.

## 2.2 Project Process Modelling

The Waterfall Model is the process model which we are following for the development of our software. In this model, the whole process of software development is divided into separate phases. The outcome of one phase acts as the input for the next phase sequentially.

This means that any phase in the development process begins only if the previous phase is complete. The waterfall model is a sequential design process in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance. As the Waterfall Model illustrates the software development process in a linear sequential flow; hence it is also referred to as a Linear-Sequential Life Cycle Model.

Sequential Phases in Waterfall Model:

- REQUIREMENTS
- SYSTEM DESIGN
- IMPLEMENTATION
- INTEGRATION AND TESTING
- DEPLOYMENT OF SYSTEM
- • MAINTENANCE

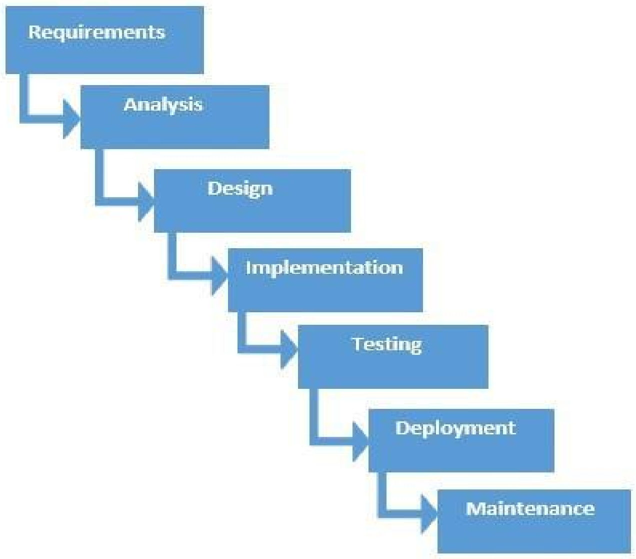


Figure 2.2 Waterfall Model

Waterfall model for our System.

**Requirements** – Video Data gathering from traffic signal, User data i.e. all information about user such as Name, Phone Number, License Number, Vehicle Model. API to send SMS once the challan has been generated. Various python packages to process video and image type of data. Tesseract Software for Optical Character Recognition.

**Analysis & Design**– Video data used in mp4 format and database used will be MySql database. The API used is Way2SMS that used python code to send message based on user key generated them. Tesseract is Software by Google used for Optical Character Recognition and used Connected Component Analysis to detect the characters and numbers in the image.

**Implementation** – Initial Module designed for video processing, it used the video to generate frames to check the vehicles that have violated the rules and send those images to the next module. The next module is the preprocessing module that takes the image and cleans it using various methods, after which various license plate detection methods like contours and bounding rectangle are used. After license plate detection the license image is preprocessed and the cleaned image is given to Tesseract which is OCR (Optical Character Recognition) to detect all the characters from the number plate. Once number is obtained database verification is done, and e-challan is generated and SMS is send using Way2SMS API.

**Testing** – Various Images of Vehicles were taken in various lighting conditions to test the image preprocessing module for license plate detection, and later given to Tesseract to check Optical Character Recognition System. The Way2SMS API is used to check that message is being sent from various different numbers to the vehicle owners and quickly and efficiently.

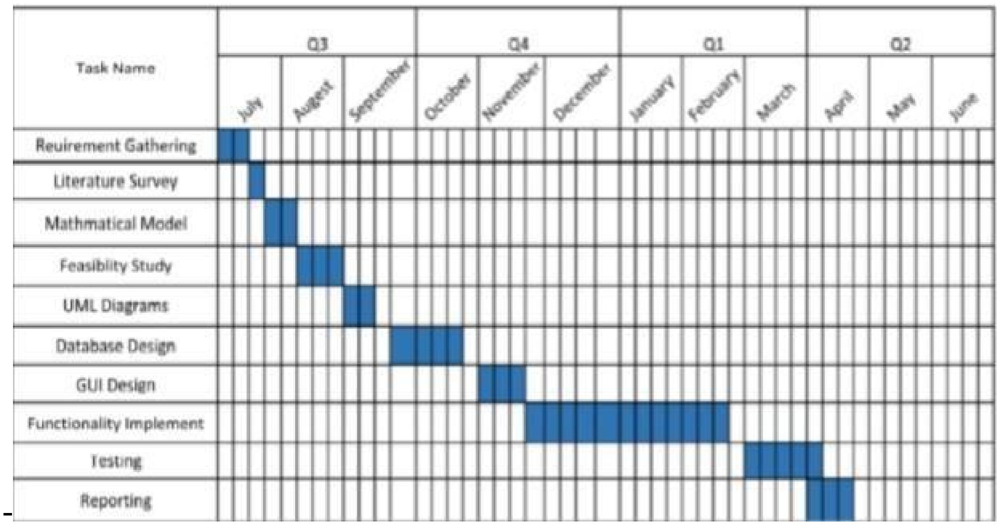
**Deployment** – The System is designed to detect traffic violation, so it will be used and deployed by the traffic authorities personal.

**Maintenance** - The Database has to be updated frequently to keep updating as new vehicles are registered so that they can be detected. The API code changes from time to time so that has to be kept updated regularly for the system to work smoothly. Tesseract Software by Google for Optical Character Recognition has to be regularly updated as the newer versions are included with better recognition capabilities and faster processing.

## 2.3 Project Scheduling

### 2.3.1 Time Line Chart

Project Scheduling communicates what activities need to perform and in what time frame.



-

**Figure 2.3 Time Line Chart**

## 2.4 Cost and Effort Estimate

To measure the cost of our project we have used COCOMO model. Cocomo (Constructive Cost Model) is a regression model based on LOC, i.e. number of Lines of Code. It is a procedural cost estimate model for software projects and often used as a process of reliably predicting the various parameters associated with ma-king a project such as size, effort, cost, time and quality. COCOMO model has hierarchy of three different types of models to calculate the cost and effort estimate of any project. Any of these three models can be used according to requirement. For our project we have used the basic type of COCOMO model to calculate the cost.

- **Basic COCOMO Model:**
  - Project Class: We have determined our project as a Semi-detached level of system.
  - Number of Lines of Code: Estimate of the source instructions delivered by our project will be 10 KLOC.

So, the Basic COCOMO model equations are as follows:

Efforts applied (E) = $a(KLOC)^b$[Man Months]

Development Time (D) = $c(E)^d$ [months]

People Required (P) = E/D

The coefficient *a, b, c, d* are predetermined according to the project.

For this project values will be:

Efforts Applied (E) = $2.8(10)^{(1.03)}$[Man months]

E = 30.002 man months

Development Time (D) = $2.4\ (3.41)^{(0.6)}$ [Months]

D = 7.88 months.

People Required (P) = 30.002/7.88 count

P = 3.8 People

# Chapter 3

# ANALYSIS & DESIGN

This chapter covers the analysis and design of the system as a whole. The chapter deals with analysis of each module separately as well as the cumulative analysis of the system.

## 3.1 Mathematical Model

Let S represent the events of the system.

S= {I,O,F,DD,NDD,Success,Failure}

Where I is the set of inputs O is the set of outputs, s is the initial state the set of functions, e is the final state

F= {F1, F2, F3, F4}

F1= Capture*image*()

Steps in F1:

1. Video input from camera (.mp4)
2. Extract frames from video

F2 = Pre-processing()

Steps in F2 :

1. Captured image is cropped
2. Cropped image is converted to grey scale image.
3. Grey scale image is blurred using Gaussian blur.
4. Adaptive threshold is applied.
5. Contours are found using bounding rectangle.
6. Plate detected and extracted.

F3= Character*recognization*()

Steps in F3:

1. Tesseract is used to recognize characters.
2. License number obtained

F4= E-challan*generation*()

1. Owner Information is extracted from database
2. Message is sent is using way2SMS API.

I= {Video Stream, Configuration}

DD = Deterministic Data

NDD = Non Deterministic Data

Success (S1,S2)

S1= {The License plate Detected Successfully}

S2 = {License plate recognize Successfully}

S3= {E-challan Generated successfully}

Failure= {F1,F2}

F1={Failed to recognize characters}

F2-{Failed to produce output}

## 3.2 UML Diagrams

### 3.2.1 Use Case Diagram

The Use Case Diagram of the project depicts the user's interaction with the system

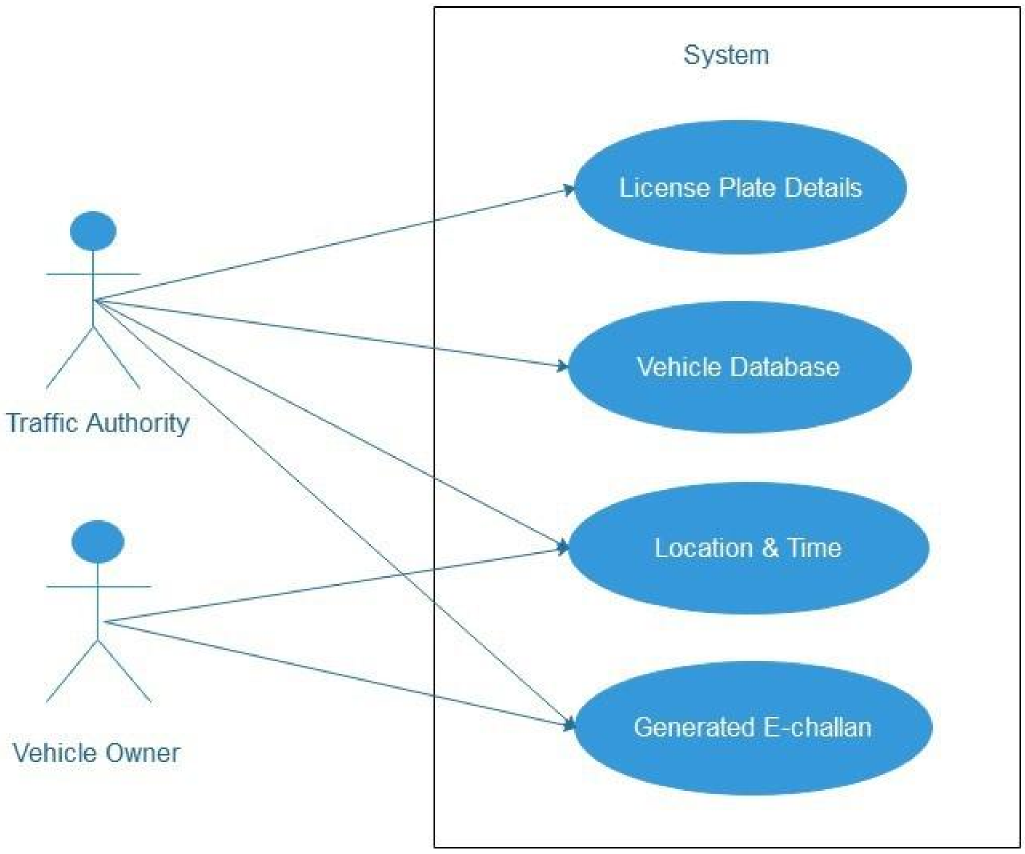


**Figure 3.1 Use Case Diagram**

### 3.2.2 Activity Diagram

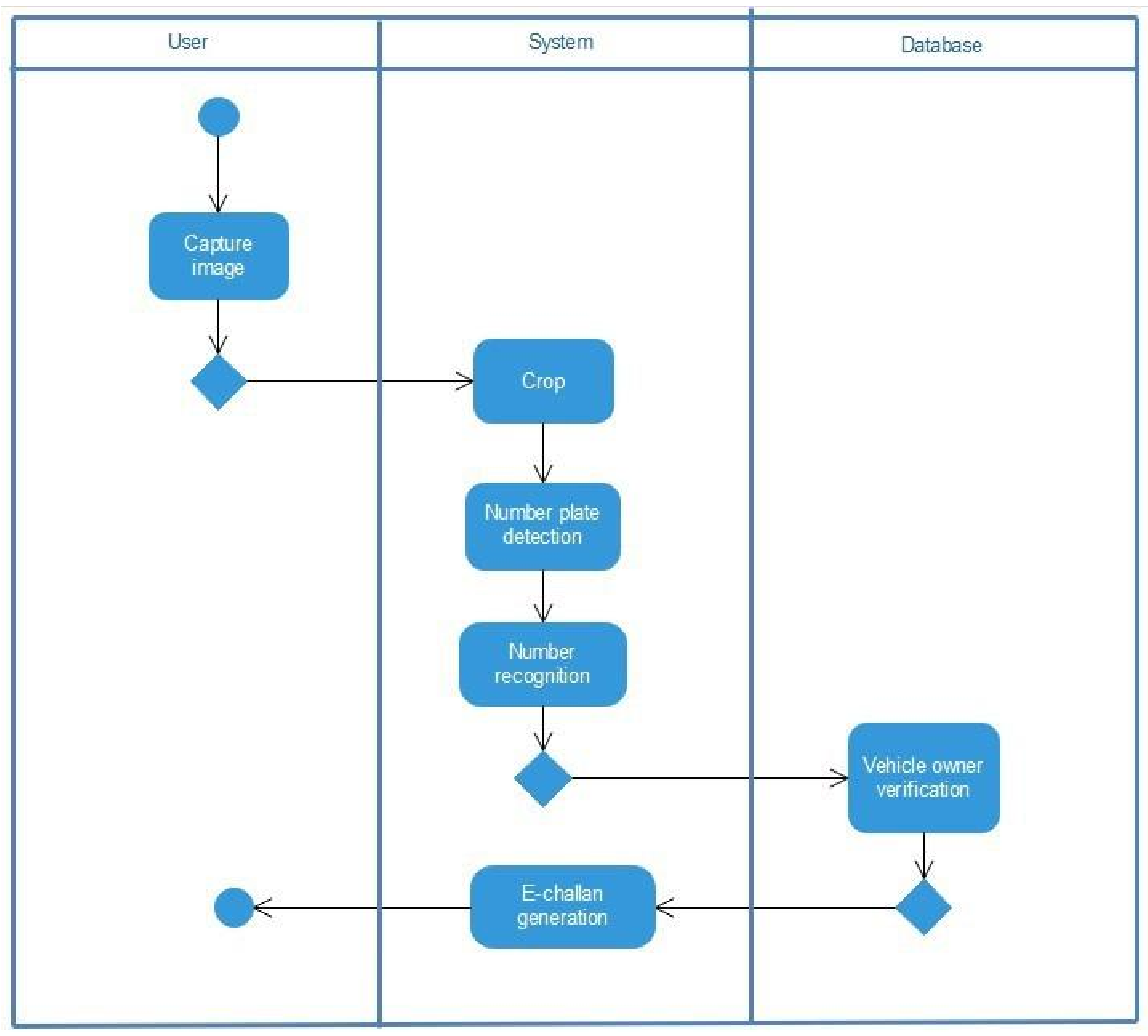


**Figure 3.2 Activity Diagram**

### 3.2.3 Data Flow Diagram

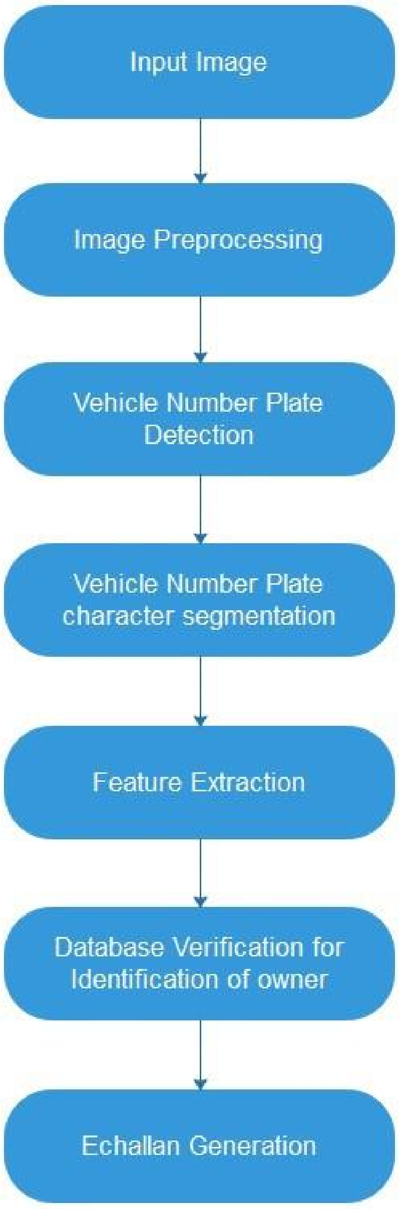

Input Image
Image Preprocessing
Vehicle Number Plate Detection
Vehicle Number Plate character segmentation
Feature Extraction
Database Verification for Identification of owner
Echallan Generation

**Figure 3.3 Data Flow Diagram**

### 3.2.4 Sequence Diagram

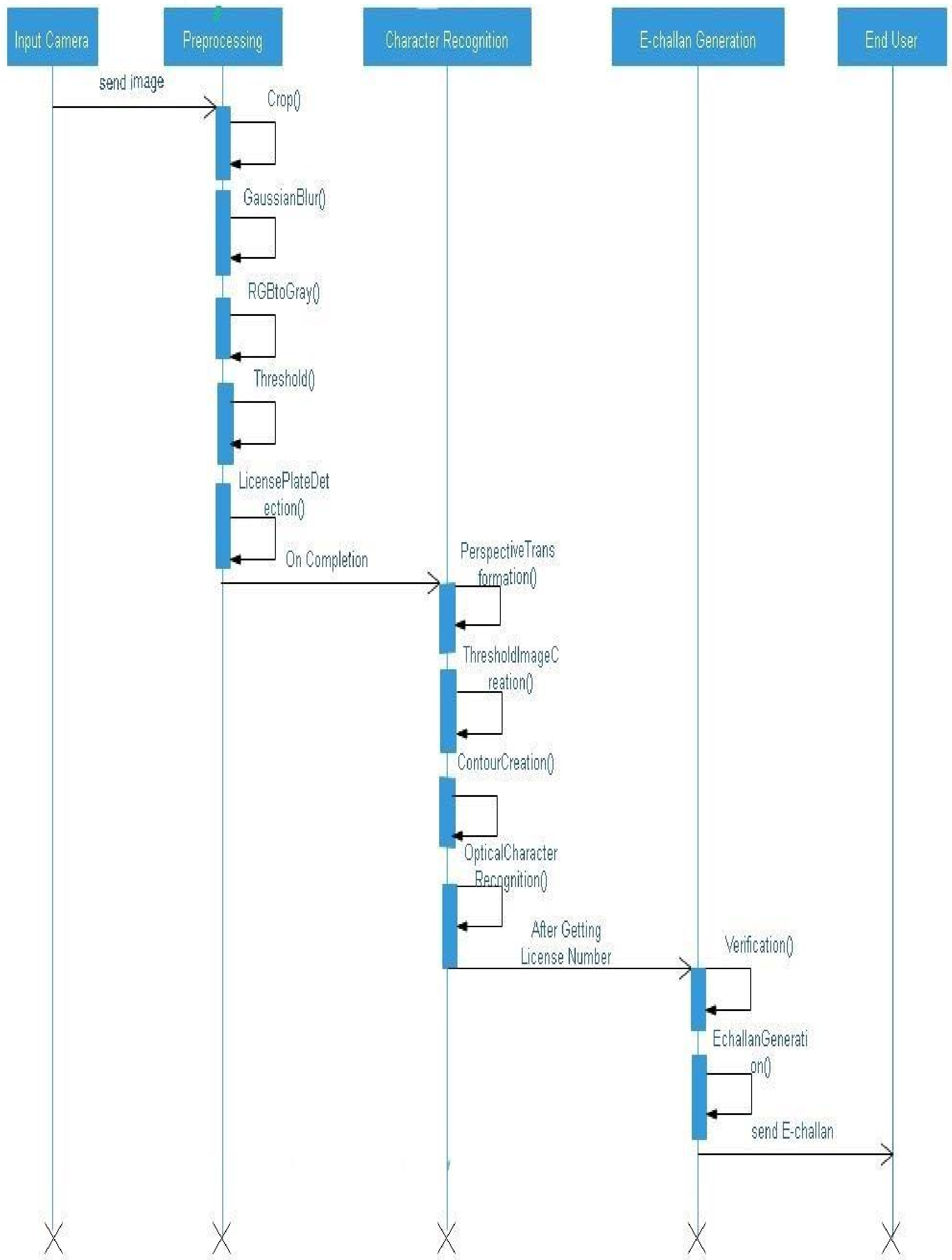


**Figure 3.4 Sequence Diagram**

### 3.2.5 Deployment Diagram

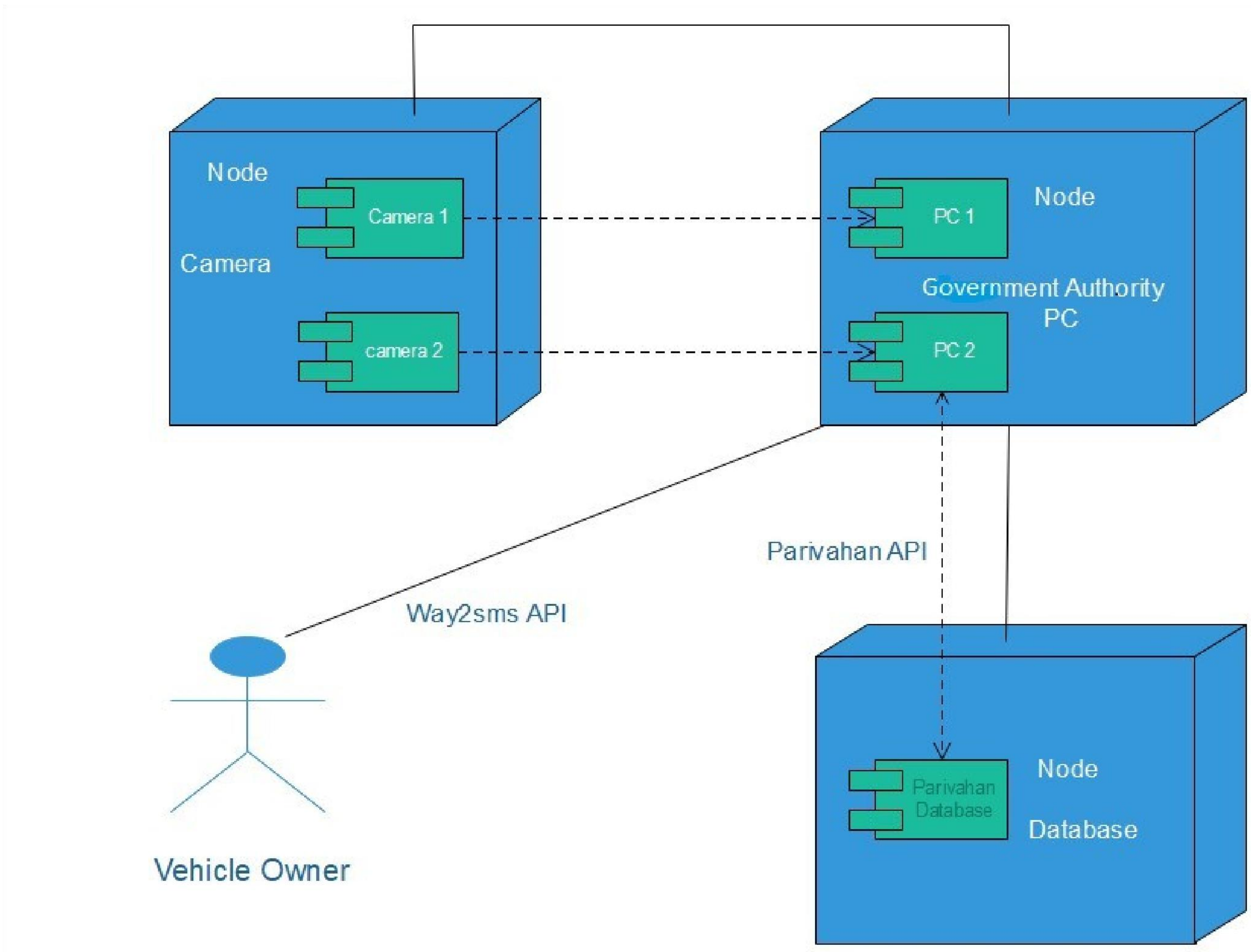


**Figure 3.5 Deployment Diagram**

### 3.2.6 Class Diagram

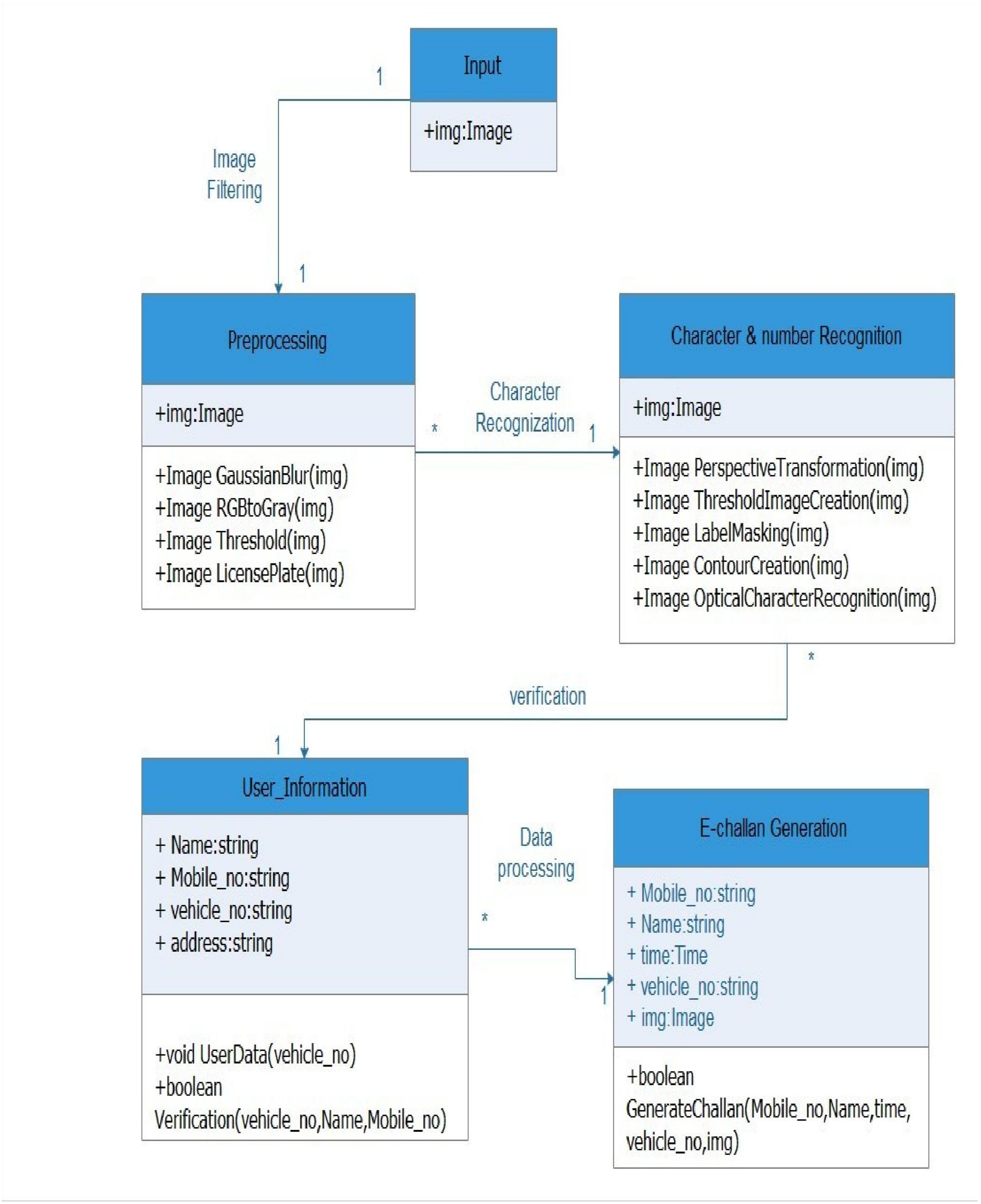


**Figure 3.6 Class Diagram**

# Chapter 4

# IMPLEMENTATION AND CODING

## 4.1 Introduction

In the previous chapters we have seen about the detailed design of the system using various UML Diagrams. With reference to these UML Diagrams we will now see the actual implementation and coding part of the system. This chapter focuses on the implementation of all the key classes of our system i.e. number plate detection, extracting registration number, generating e-challan, sending SMS to vehicle owner, etc. This chapter also contains the database architecture used in our system. Also, some screenshots of the running system are provided in this chapter.

## 4.2 Database Schema

Our database contains two tables. One table contains information about vehicle  vehicles and another table contains information about vehicles who have violated the rule. Table one has attributes challan no, name, vehicle no, offence committed, mobile no, vehicle model, total fine. Second table has attributes challan no, name, vehicle no, offence committed and fine.

## 4.3 Operational Details

### 4.3.1 Input

Input for the project is taken in the form of video. The video represents the camera installed at any given traffic signal. The camera is installed such that is captures the entire width of the zebra crossing. The video taken in previous step is divided into frames i.e. separate images so that we can process them. The frames that are generated later in the video are taken as they best represent that some vehicle might have broken the rule.

### 4.3.2 Preprocessing - Cropping and Reshaping

The image taken is cropped from the top to only get the bottom section so that the number plates of only the vehicles that have broken the rule remain in the image and rest are lost. The model developed requires a less resolution image so the cropped image is re-sized to a resolution of 1280 x 670.

### 4.3.3 Plate Detection

The re-sized image goes through a no. of different preprocessing techniques like Gaussian Blur, Sobel, Threshold etc. A rectangular structuring element representing the rectangular number plate is taken. After taking threshold all contours are obtained from that structuring element. All the obtained contours are checked for their height, width, and brightness are checked to detect correct rectangle from all contours. The obtained image plate is further preprocessed by all the above techniques. The count of all the alphabet contours are taken entire number plate is blacked except the numbers.

### 4.3.4 Extracting Registration Number

The cleaned number plate is given to the Tesseract OCR Engine program by Google. Tesseract OCR engine uses Connected Component Analysis to detect all the characters from the plate. Recognition passes through 2 passes. Each word that is recognized is given to the adaptive classifier to generate training data. The second pass is used to further recognize words better.

### 4.3.5 Get Vehicle Owner Information

The number obtained is checked in the database to get all details regarding the owner of the vehicle and offence committed is given in the table. The GUI displays all the information and the image of the vehicle from where the offence is committed.

### 4.3.6 Sending E-Challan Regarding Offence

Once all the user details are obtained the application operator can approve to send the details of the offence committed via SMS.

## 4.4 Major Classes

**Video**

This class takes the input in the form of video and generates all the frames in the image and resize them accordingly for the system to work.

**GUI**

It is the visual representation of the system the user will see, where all the details of the offence committed and user can take action regarding offence.

**Plate-Detection**

This entire class is dedicated to number plate detection, entire pre-processing, registration number extraction is taken care by this class.

**Send SMS**

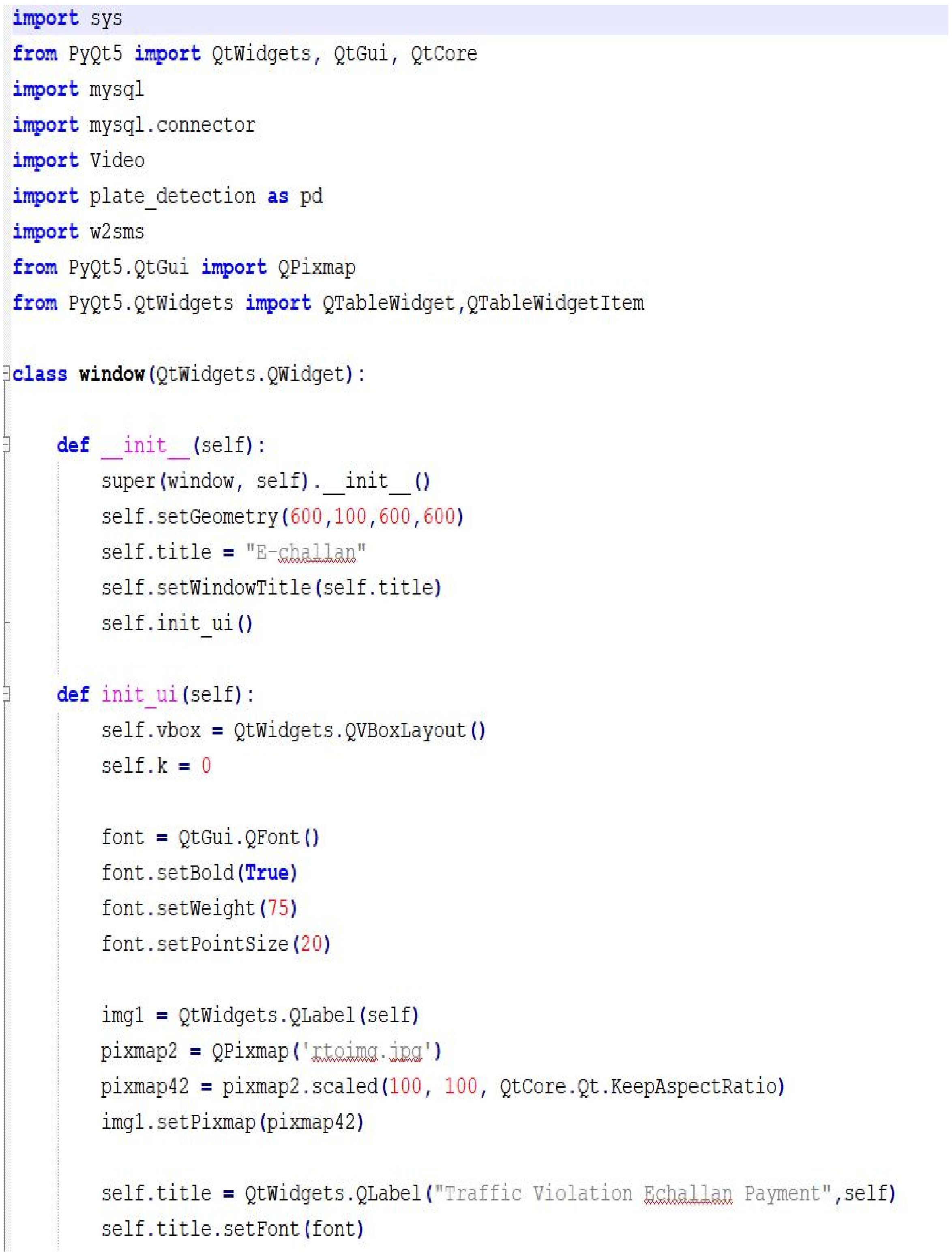
This class uses the way-2-SMS API to send message to the owner of the vehicle who has committed the offence.

## 4.5 Code Listing

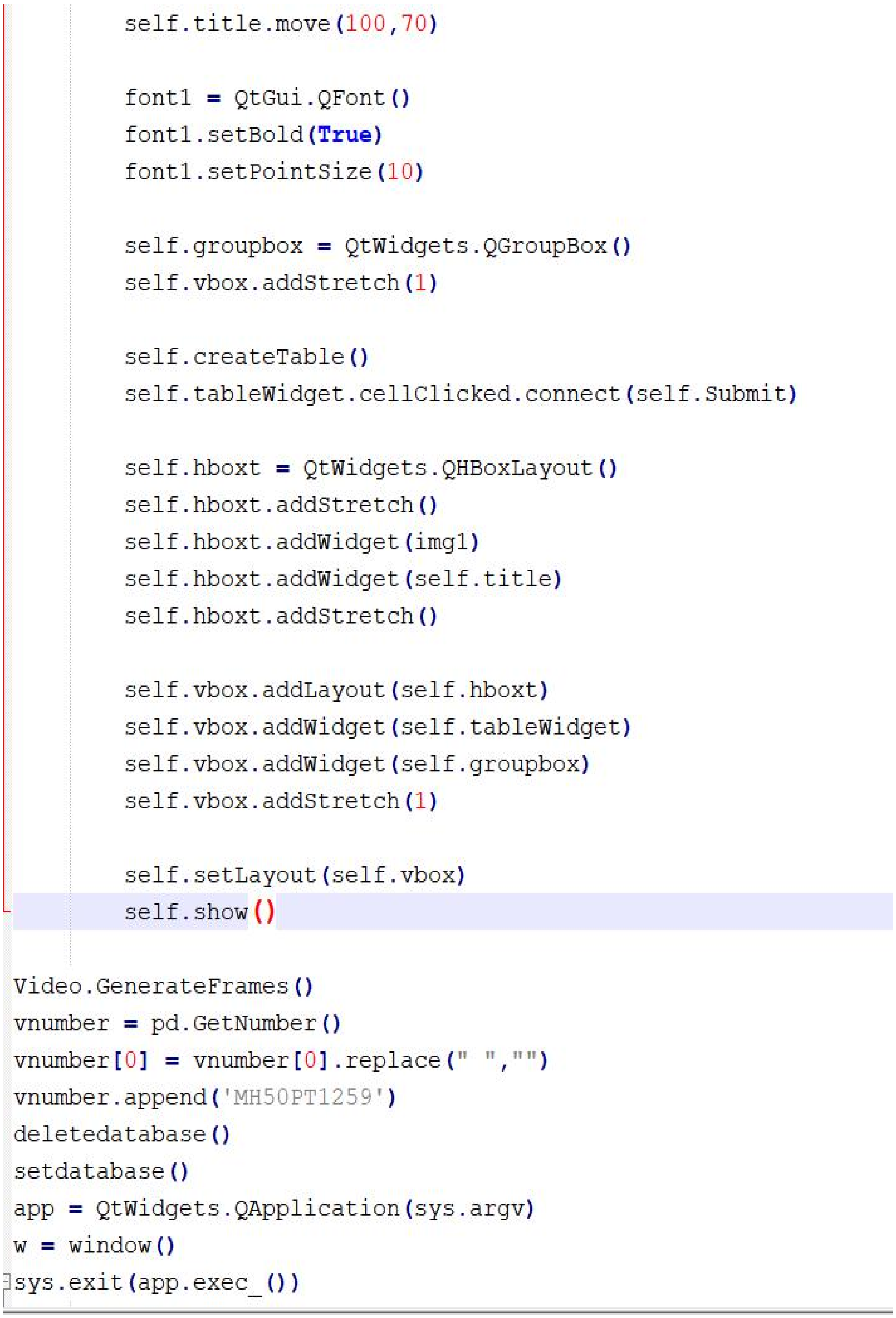
### Video

```
import cv2
import os
from PIL import Image

def GenerateFrames():
    cam = cv2.VideoCapture("opencv.mp4")
    fps = int(cam.get(cv2.CAP_PROP_FPS))
    print(fps)
    try:
        if not os.path.exists('data'):
            os.makedirs('data')
    except OSError:
        print ('Error: Creating directory of data')
    ret = True
    currentframe = 0
    while(ret):
        ret,frame = cam.read()
        if currentframe%(1*fps) == 0:
            name = './data/frame' + str(currentframe) + '.jpg'
            print ('Creating...' + name)
            cv2.imwrite(name, frame)
        currentframe += 1
    cam.release()
    cv2.destroyAllWindows()
    imageObject = Image.open("./data/frame368.jpg")
    cropped = imageObject.crop((1000,1480,3400,2150))
    cropped.save('Cropped.png','PNG')
    size = 1280, 670
    im = Image.open("Cropped.png")
    im_resized = im.resize(size, Image.ANTIALIAS)
    im_resized.save("Test.png", "PNG")
```

## GUI

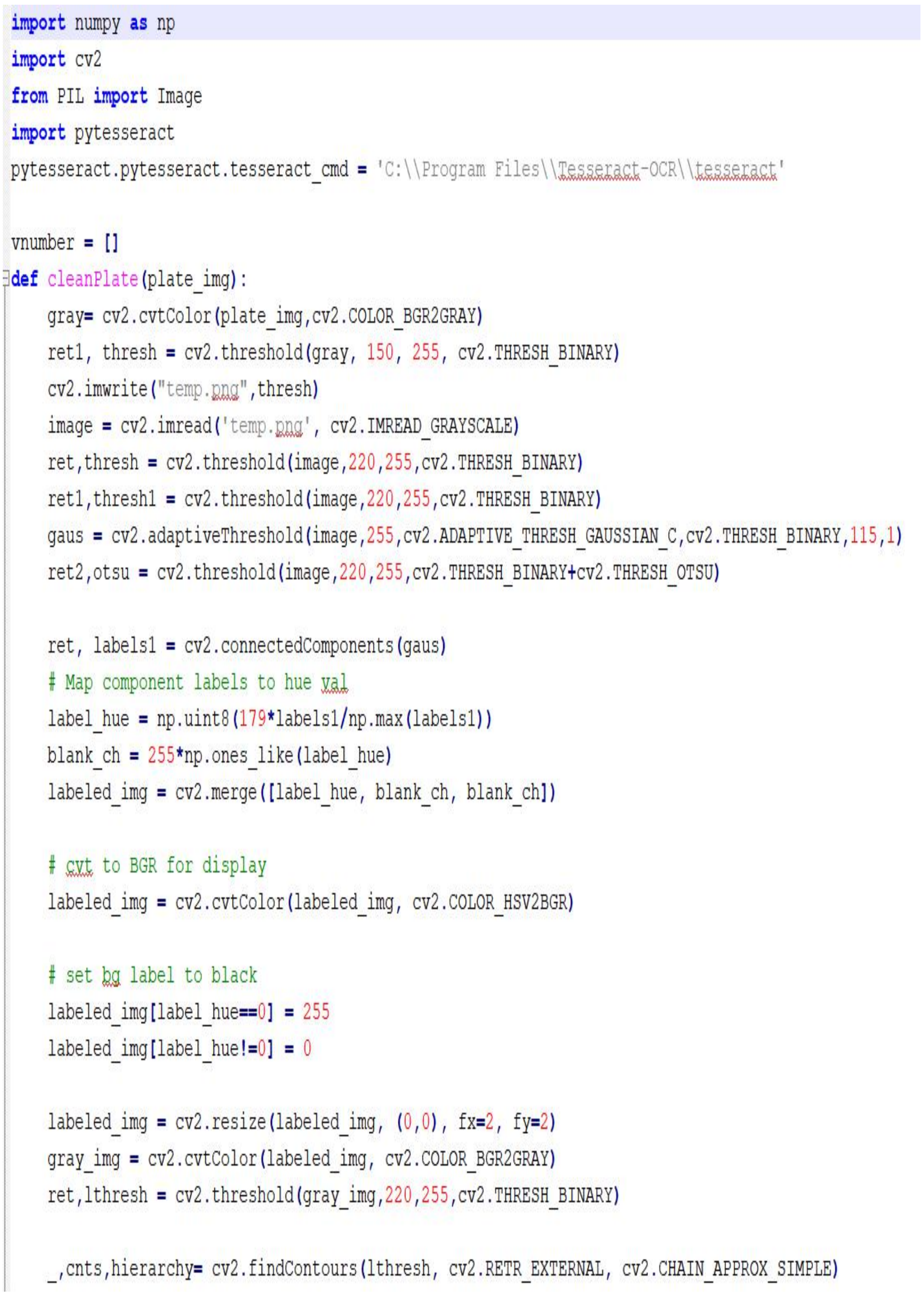


```
import sys
from PyQt5 import QtWidgets, QtGui, QtCore
import mysql
import mysql.connector
import Video
import plate_detection as pd
import w2sms
from PyQt5.QtGui import QPixmap
from PyQt5.QtWidgets import QTableWidget,QTableWidgetItem

class window(QtWidgets.QWidget):

    def __init__(self):
        super(window, self).__init__()
        self.setGeometry(600,100,600,600)
        self.title = "E-challan"
        self.setWindowTitle(self.title)
        self.init_ui()

    def init_ui(self):
        self.vbox = QtWidgets.QVBoxLayout()
        self.k = 0

        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        font.setPointSize(20)

        img1 = QtWidgets.QLabel(self)
        pixmap2 = QPixmap('rtoimg.jpg')
        pixmap42 = pixmap2.scaled(100, 100, QtCore.Qt.KeepAspectRatio)
        img1.setPixmap(pixmap42)

        self.title = QtWidgets.QLabel("Traffic Violation Echallan Payment",self)
        self.title.setFont(font)
```

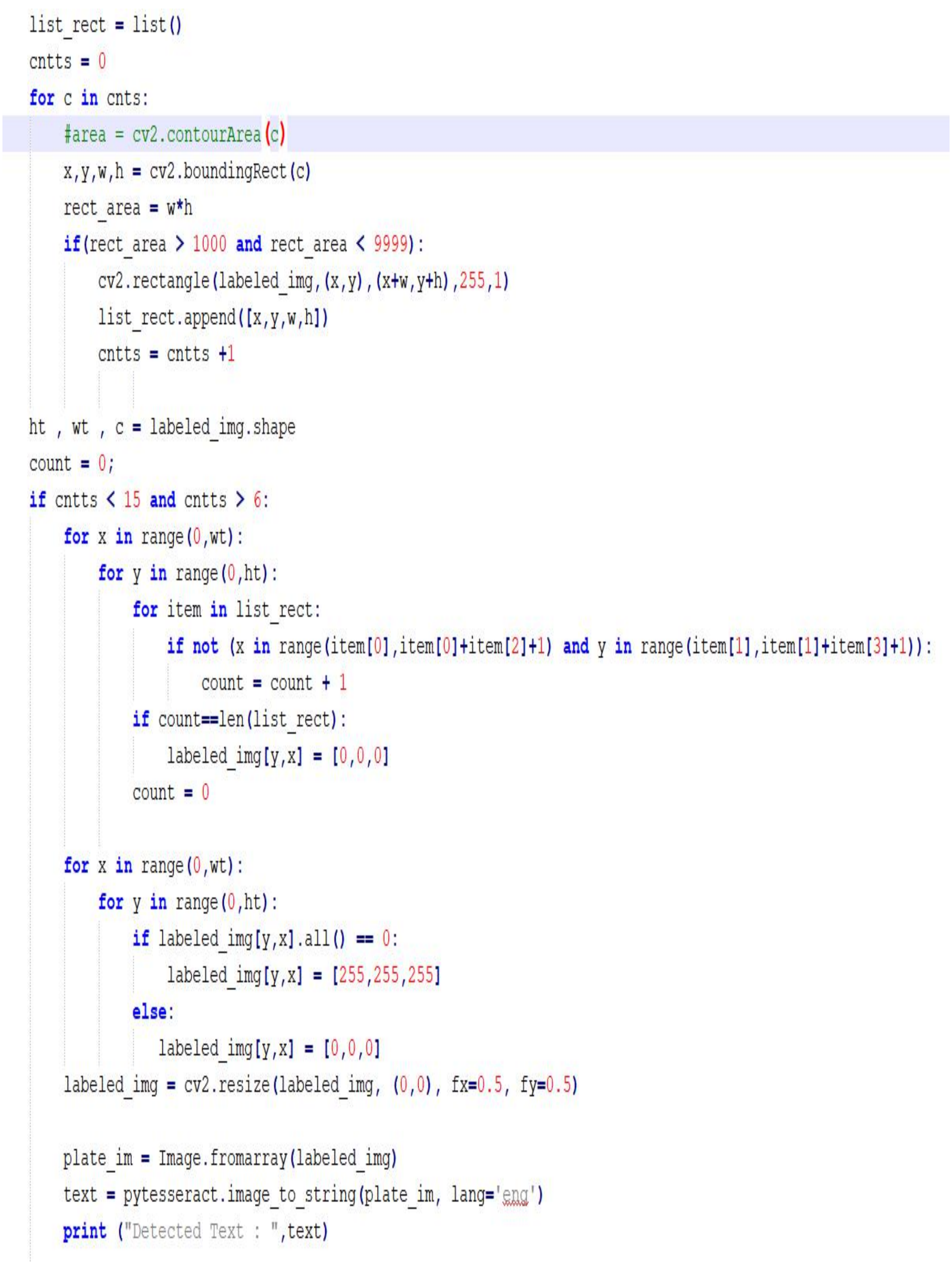


```
        self.title.move(100,70)

        font1 = QtGui.QFont()
        font1.setBold(True)
        font1.setPointSize(10)

        self.groupbox = QtWidgets.QGroupBox()
        self.vbox.addStretch(1)

        self.createTable()
        self.tableWidget.cellClicked.connect(self.Submit)

        self.hboxt = QtWidgets.QHBoxLayout()
        self.hboxt.addStretch()
        self.hboxt.addWidget(img1)
        self.hboxt.addWidget(self.title)
        self.hboxt.addStretch()

        self.vbox.addLayout(self.hboxt)
        self.vbox.addWidget(self.tableWidget)
        self.vbox.addWidget(self.groupbox)
        self.vbox.addStretch(1)

        self.setLayout(self.vbox)
        self.show()

Video.GenerateFrames()
vnumber = pd.GetNumber()
vnumber[0] = vnumber[0].replace(" ","")
vnumber.append('MH50PT1259')
deletedatabase()
setdatabase()
app = QtWidgets.QApplication(sys.argv)
w = window()
sys.exit(app.exec_())
```

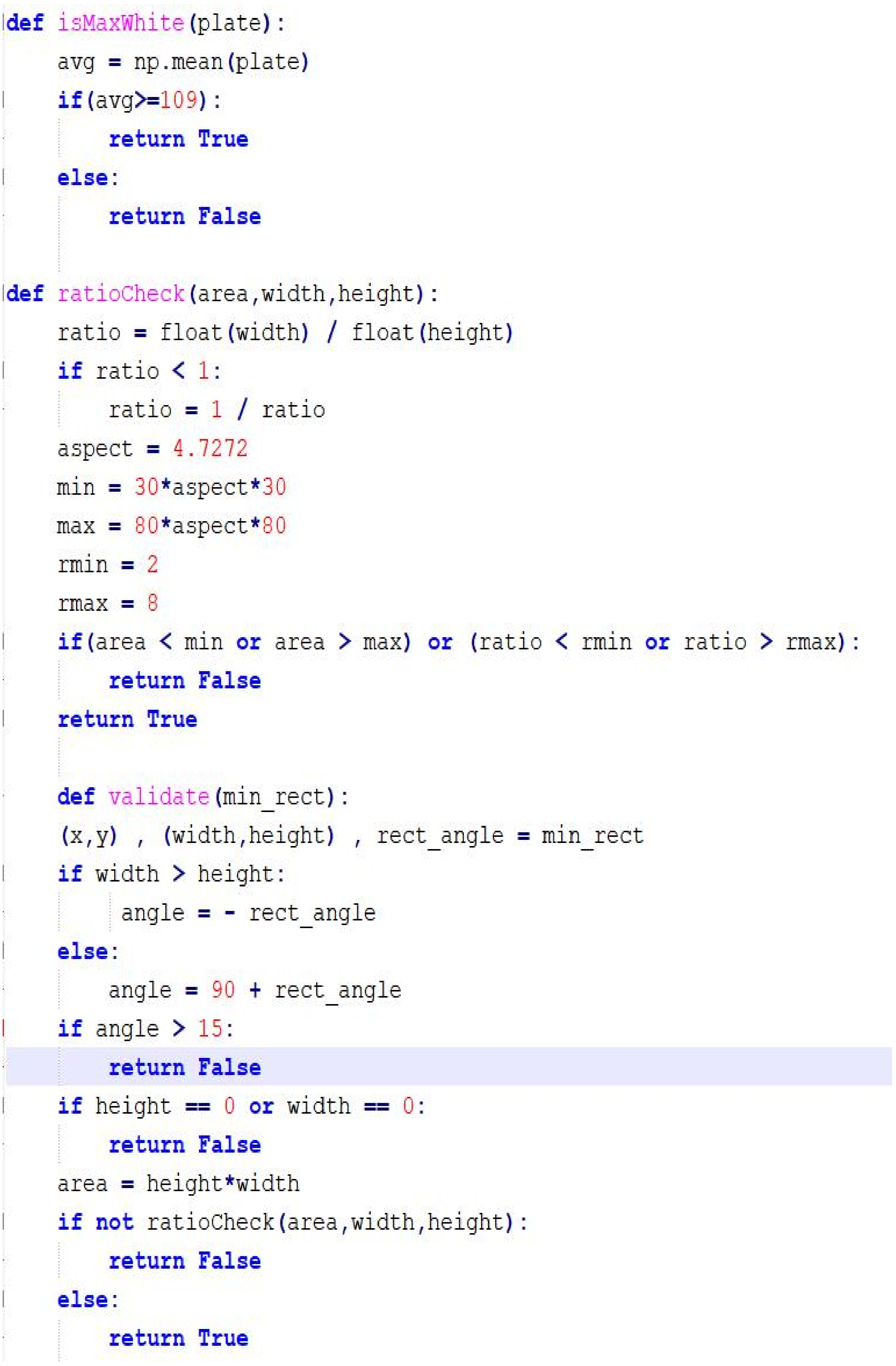
## Plate-Detection



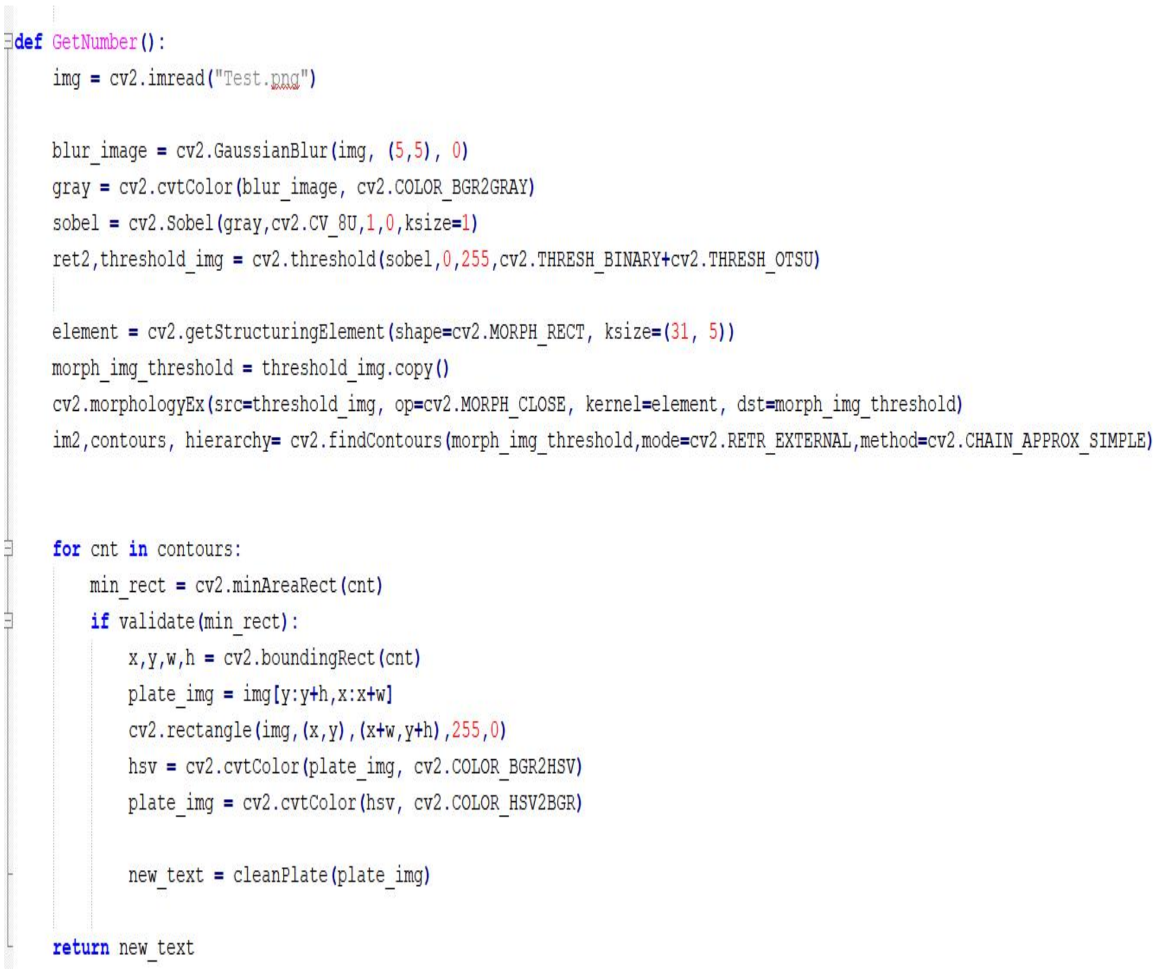
```
import numpy as np
import cv2
from PIL import Image
import pytesseract
pytesseract.pytesseract.tesseract_cmd = 'C:\\Program Files\\Tesseract-OCR\\tesseract'

vnumber = []
def cleanPlate(plate_img):
    gray= cv2.cvtColor(plate_img,cv2.COLOR_BGR2GRAY)
    ret1, thresh = cv2.threshold(gray, 150, 255, cv2.THRESH_BINARY)
    cv2.imwrite("temp.png",thresh)
    image = cv2.imread('temp.png', cv2.IMREAD_GRAYSCALE)
    ret,thresh = cv2.threshold(image,220,255,cv2.THRESH_BINARY)
    ret1,thresh1 = cv2.threshold(image,220,255,cv2.THRESH_BINARY)
    gaus = cv2.adaptiveThreshold(image,255,cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY,115,1)
    ret2,otsu = cv2.threshold(image,220,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

    ret, labels1 = cv2.connectedComponents(gaus)
    # Map component labels to hue val
    label_hue = np.uint8(179*labels1/np.max(labels1))
    blank_ch = 255*np.ones_like(label_hue)
    labeled_img = cv2.merge([label_hue, blank_ch, blank_ch])

    # cvt to BGR for display
    labeled_img = cv2.cvtColor(labeled_img, cv2.COLOR_HSV2BGR)

    # set bg label to black
    labeled_img[label_hue==0] = 255
    labeled_img[label_hue!=0] = 0

    labeled_img = cv2.resize(labeled_img, (0,0), fx=2, fy=2)
    gray_img = cv2.cvtColor(labeled_img, cv2.COLOR_BGR2GRAY)
    ret,lthresh = cv2.threshold(gray_img,220,255,cv2.THRESH_BINARY)

    _,cnts,hierarchy= cv2.findContours(lthresh, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
```



```
list_rect = list()
cntts = 0
for c in cnts:
    #area = cv2.contourArea(c)
    x,y,w,h = cv2.boundingRect(c)
    rect_area = w*h
    if(rect_area > 1000 and rect_area < 9999):
        cv2.rectangle(labeled_img,(x,y),(x+w,y+h),255,1)
        list_rect.append([x,y,w,h])
        cntts = cntts +1

ht , wt , c = labeled_img.shape
count = 0;
if cntts < 15 and cntts > 6:
    for x in range(0,wt):
        for y in range(0,ht):
            for item in list_rect:
                if not (x in range(item[0],item[0]+item[2]+1) and y in range(item[1],item[1]+item[3]+1)):
                    count = count + 1
            if count==len(list_rect):
                labeled_img[y,x] = [0,0,0]
            count = 0

    for x in range(0,wt):
        for y in range(0,ht):
            if labeled_img[y,x].all() == 0:
                labeled_img[y,x] = [255,255,255]
            else:
                labeled_img[y,x] = [0,0,0]
    labeled_img = cv2.resize(labeled_img, (0,0), fx=0.5, fy=0.5)

    plate_im = Image.fromarray(labeled_img)
    text = pytesseract.image_to_string(plate_im, lang='eng')
    print ("Detected Text : ",text)
```

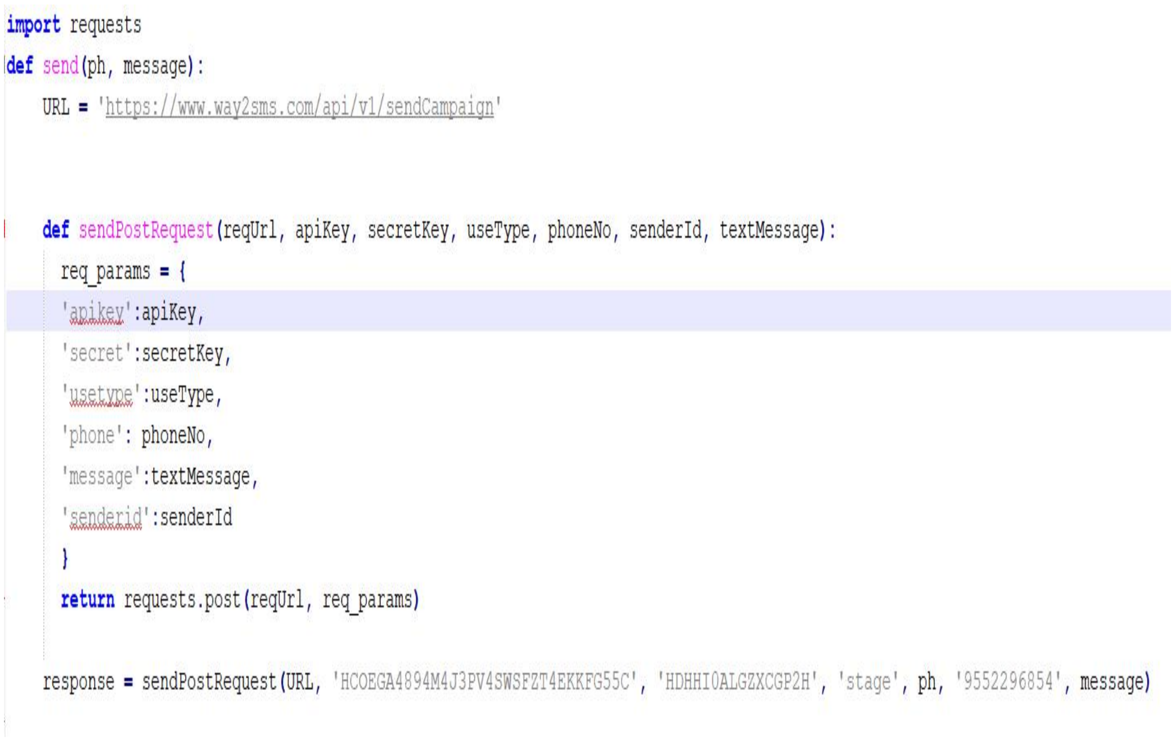


```
def isMaxWhite(plate):
    avg = np.mean(plate)
    if(avg>=109):
        return True
    else:
        return False

def ratioCheck(area,width,height):
    ratio = float(width) / float(height)
    if ratio < 1:
        ratio = 1 / ratio
    aspect = 4.7272
    min = 30*aspect*30
    max = 80*aspect*80
    rmin = 2
    rmax = 8
    if(area < min or area > max) or (ratio < rmin or ratio > rmax):
        return False
    return True

    def validate(min_rect):
    (x,y) , (width,height) , rect_angle = min_rect
    if width > height:
         angle = - rect_angle
    else:
        angle = 90 + rect_angle
    if angle > 15:
        return False
    if height == 0 or width == 0:
        return False
    area = height*width
    if not ratioCheck(area,width,height):
        return False
    else:
        return True
```



```python
def GetNumber():
    img = cv2.imread("Test.png")

    blur_image = cv2.GaussianBlur(img, (5,5), 0)
    gray = cv2.cvtColor(blur_image, cv2.COLOR_BGR2GRAY)
    sobel = cv2.Sobel(gray,cv2.CV_8U,1,0,ksize=1)
    ret2,threshold_img = cv2.threshold(sobel,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)

    element = cv2.getStructuringElement(shape=cv2.MORPH_RECT, ksize=(31, 5))
    morph_img_threshold = threshold_img.copy()
    cv2.morphologyEx(src=threshold_img, op=cv2.MORPH_CLOSE, kernel=element, dst=morph_img_threshold)
    im2,contours, hierarchy= cv2.findContours(morph_img_threshold,mode=cv2.RETR_EXTERNAL,method=cv2.CHAIN_APPROX_SIMPLE)


    for cnt in contours:
        min_rect = cv2.minAreaRect(cnt)
        if validate(min_rect):
            x,y,w,h = cv2.boundingRect(cnt)
            plate_img = img[y:y+h,x:x+w]
            cv2.rectangle(img,(x,y),(x+w,y+h),255,0)
            hsv = cv2.cvtColor(plate_img, cv2.COLOR_BGR2HSV)
            plate_img = cv2.cvtColor(hsv, cv2.COLOR_HSV2BGR)

            new_text = cleanPlate(plate_img)

    return new_text
```

## Send SMS



```python
import requests
def send(ph, message):
    URL = 'https://www.way2sms.com/api/v1/sendCampaign'



    def sendPostRequest(reqUrl, apiKey, secretKey, useType, phoneNo, senderId, textMessage):
      req_params = {
      'apikey':apiKey,
      'secret':secretKey,
      'usetype':useType,
      'phone': phoneNo,
      'message':textMessage,
      'senderid':senderId
      }
      return requests.post(reqUrl, req_params)

    response = sendPostRequest(URL, 'HCOEGA4894M4J3PV4SWSFZT4EKKFG55C', 'HDHHI0ALGZXCGP2H', 'stage', ph, '9552296854', message)
```

# Chapter 5
# TESTING

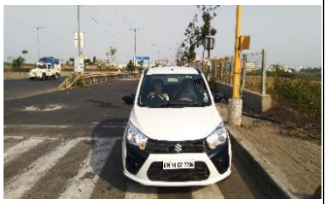
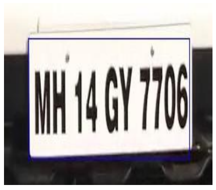
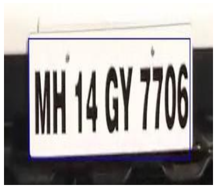
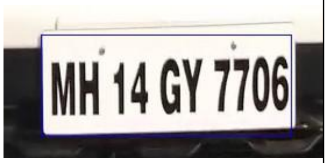
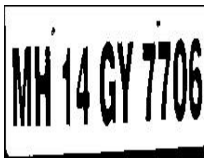
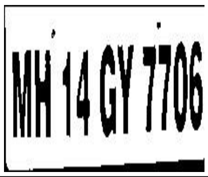
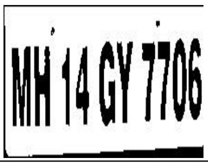
Software testing is the process of evaluation a software item to detect differences between given input and expected output. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process. The various types of testing that we have tested the system over are as follows:

1- Unit Testing 2- Integration Testing 3- Acceptance Testing

This chapter covers the testing approach used and the test cases.

## 5.1 Unit Testing

Unit Testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output. The following table shows the various test cases of smell recognition module.

| Sr.No. | Testing Conducted | Input | Expected Output | Actual Output |
|---|---|---|---|---|
| 1. | Pre-processing |  |  |  |
| 2. | Pre-processing |  |  |  |
| 3. | Feature Extraction |  | MH 14 GY 7706 | MH 14 GY 7706 |

| | | | | |
|---|---|---|---|---|
| 4. | Verification | MH 14 GY 7706 | Owner of the vehicle identified and E-challan Generated. | Owner of the vehicle identified and E-challan Generated. |

**Table 5.1 Unit Testing**

## 5.2 Integration Testing

Integration Testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Integration Testing is the second level of testing performed after Unit Testing and before System Testing. Integration testing, in context of the system, is the testing of various modules with the dependencies taken into consideration. That means, integration testing deals with the impact of the one module over other module. The following depicts the various scenarios in which the system works:

| Sr.No. | Testing Con-Ducted | Input | Expected Output | Actual Output |
|---|---|---|---|---|
| 1. | Pre-Processing | Captured Image | Noise free number plate image | Noise free number plate image |
| 2. | Feature Extraction and Verification | Noise free number plate image | Found Owner of the Vehicle and E-challan generated successfully. | Found Owner of the Vehicle and E-challan generated successfully. |

**Table 5.2 Integration Testing**

## 5.3 Acceptance Testing

Acceptance Testing is a level of software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery. Acceptance Testing is the fourth and last level of software testing performed after System Testing and before making the system available for actual use.

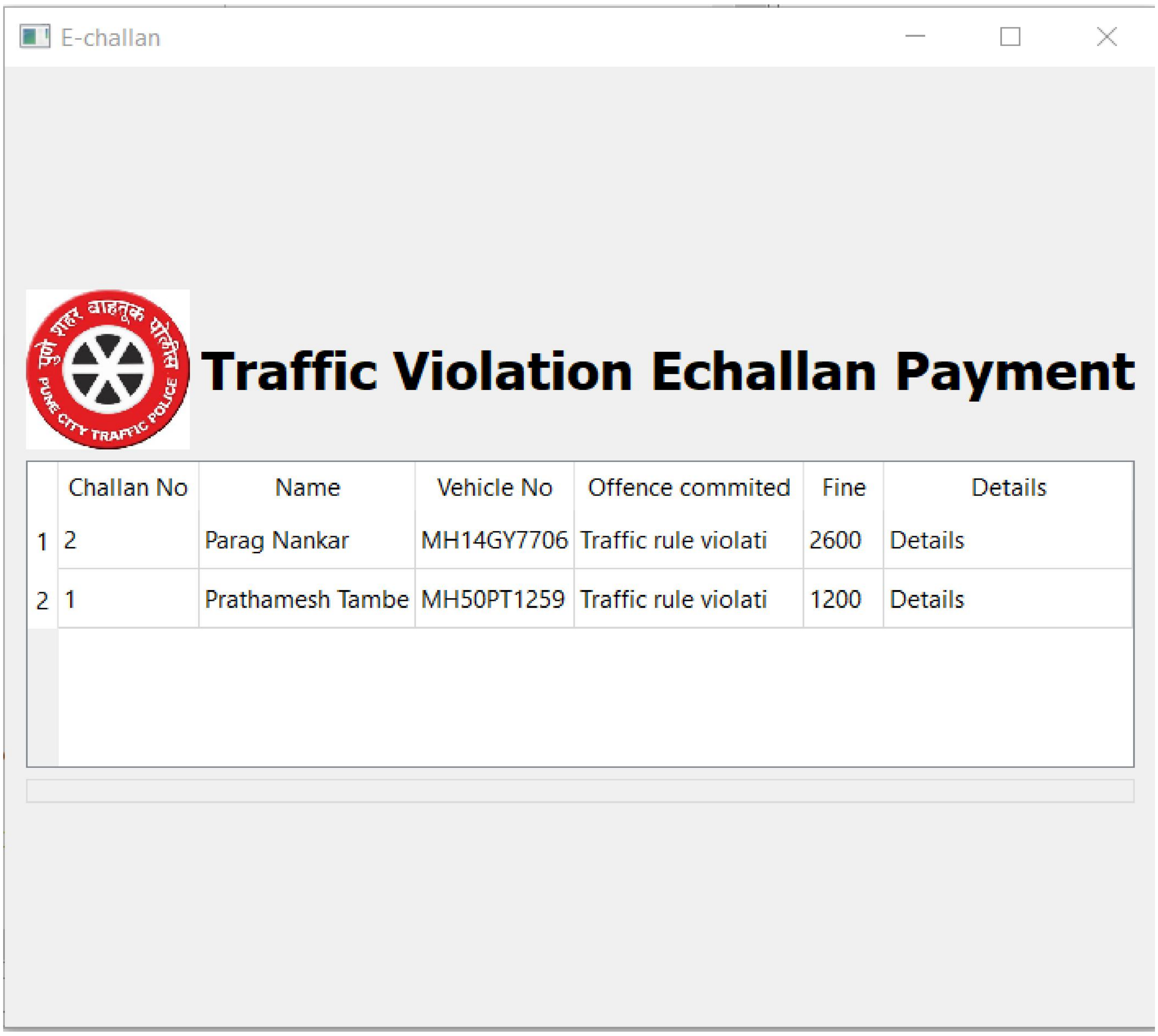
| Test ID | Test Category | Test Description |
|---|---|---|
| 1. | License plate Detection and License plate number Recognition. | License plate is detected and license plate number is recognized for further processing. |
| 2. | E-challan Generation | E-challan generated and notified to user using message. |

**Table 5.3 Acceptance Testing**

# Chapter 6
# RESULTS AND DISCUSSION

## 6.1 Main GUI Snapshots

Initial Window

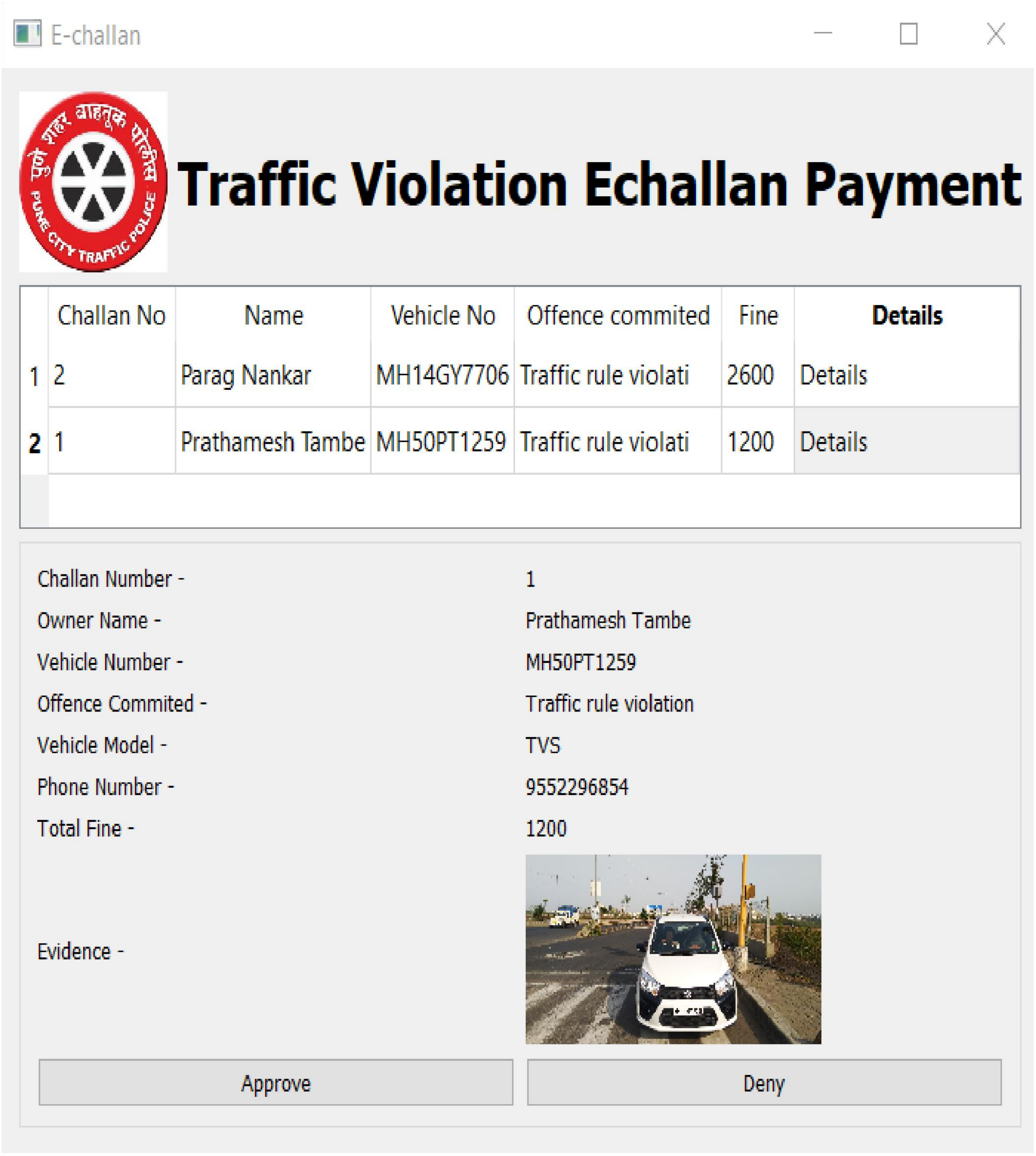


Traffic Violation Echallan Payment

| | Challan No | Name | Vehicle No | Offence commited | Fine | Details |
|---|---|---|---|---|---|---|
| 1 | 2 | Parag Nankar | MH14GY7706 | Traffic rule violati | 2600 | Details |
| 2 | 1 | Prathamesh Tambe | MH50PT1259 | Traffic rule violati | 1200 | Details |

**Figure 6.1 Initial window**

After clicking on details of User 1:



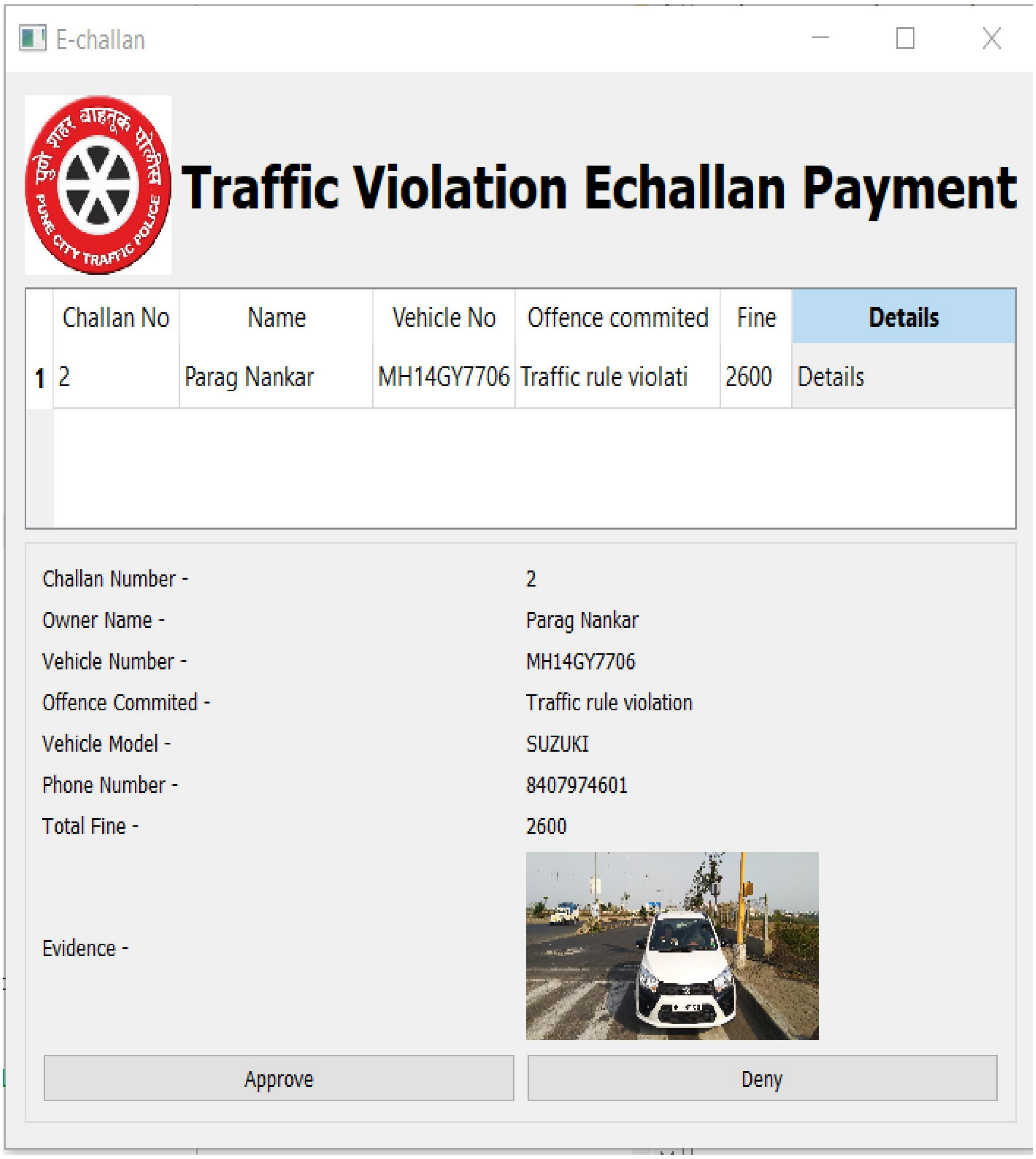
Traffic Violation Echallan Payment

| | Challan No | Name | Vehicle No | Offence commited | Fine | **Details** |
|---|---|---|---|---|---|---|
| 1 | 2 | Parag Nankar | MH14GY7706 | Traffic rule violati | 2600 | Details |
| **2** | 1 | Prathamesh Tambe | MH50PT1259 | Traffic rule violati | 1200 | Details |

| | |
|---|---|
| Challan Number - | 1 |
| Owner Name - | Prathamesh Tambe |
| Vehicle Number - | MH50PT1259 |
| Offence Commited - | Traffic rule violation |
| Vehicle Model - | TVS |
| Phone Number - | 9552296854 |
| Total Fine - | 1200 |
| Evidence - | |

**Figure 6.2 Output 1**

After clicking on details of User 2:



E-challan

Traffic Violation Echallan Payment

| | Challan No | Name | Vehicle No | Offence commited | Fine | Details |
|---|---|---|---|---|---|---|
| 1 | 2 | Parag Nankar | MH14GY7706 | Traffic rule violati | 2600 | Details |

| | |
|---|---|
| Challan Number - | 2 |
| Owner Name - | Parag Nankar |
| Vehicle Number - | MH14GY7706 |
| Offence Commited - | Traffic rule violation |
| Vehicle Model - | SUZUKI |
| Phone Number - | 8407974601 |
| Total Fine - | 2600 |
| Evidence - | |

**Figure 6.3 Output 2**

## 6.2 Discussions

The Traffic E-challan Generation is totally python based application. It collects data from camera and detect the violated vehicles’ license number. On this application, first user will get list of violated incidents. Details of incident is displayed on this application with all necessary information and image. After user approves, this application sends message to vehicle owner when vehicle breaks the rule.

# Chapter 7

# Conclusion and Future Work

## 7.1 Conclusion of final project

The system designed works on the input video taken from any traffic signal, it checks for rule violation, detects the number plates and produces e-challan automatically. The video taken is broken down into frames and frames are processed for detecting rule violation. The frame is initially cropped to detect the violated vehicles and the cropped image is further processed to detect the license plate. Tesseract is used to find license plate number which will be used for identifying the owner and e-challan generation by the authority.

## 7.2 Future Work

In the future, the proposed methodology can be integrated with other yet to be developed methods for better rule detection and e-challan generation using more complex and accurate image processing algorithms. The performance of the system can be increased by using better camera.

# References

[1] TANG Yufeng, LUO Bin,TANG Jin, WU Fubu. New heuristic algorithm of license plate location based on statistical feature.[J] Computer Engineering and Application, 2015,51(6): 154-158.

[2] Abolghasemi V, Ahmadyfard A. An edge-based color-aided method for license plate detection[J]. Image Vision Computing, 2009, 27(8):1134-1142.

[3] Comelli P,Ferragina P,Granieri M N. Optical Recognition of Motor Vehicle License Plates[J], Vehicular Technology, IEEE Transactions on, 1995,44: 790-799.

[4] C. J. Ahmad and M. Shridhar, "Recognition of handwritten numerals with multiple feature and multistage classifier, "Pattern Recognit.,vol.2,no. 28, Feb. 1995, pp. 153–160

[5] K. K. Kim, K. I. Kim, J. B. Kim, and H. J. Kim, "Learning-based approach for license plate recognition," in Proc. IEEE Signal Process.Soc Workshop Neur. Netw. Signal Process., 2000,pp.614–623.

[6] A. Rahman, W. Badawy, and A. Radmanesh, "A real time vehicle'slicense plate recognition system," inProc. IEEE Conf. Advanced VideoSignal Based Surveillance, pp. 2003,163–166.

[7] Deneen K M V. An Algorithm for License Plate Recognition Applied to Intelligent Transportation System[J]. IEEE Transactions on Intelligent Transportation Systems, 2011, 12(3):830-845.

**Websites:**

[1] **https://www.python.org**

[2] **https://www.github.com**

[3] **https://www.tutorialspoint.com**

[4] **https://www.opencv.org**

[5] **https://www.sourceforge.org**