# Intelligent Parking Management System (IPMS)

**A Project Report**

***Submitted by***

**Akarsh Sharma – E032**

**Aakansh Togani – E049**

**Rohan Tuli – E052**

***Under the Guidance of***

**Prof. Ameyaa Biwalkar**

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

**B.Tech**

**Computer Science**
At


**MUKESH PATEL SCHOOL TECHNOLOGY MANAGEMENT AND ENGINEERING, VILE PARLE**

**MARCH, 2020**

# DECLARATION

We, Akarsh Sharma, Aakansh Togani and Rohan Tuli, Roll No. E032, E049, E052 B.Tech (Computer Engineering), VI semester understand that plagiarism is defined as anyone or combination of the following:

1. Un-credited verbatim copying of individual sentences, paragraphs or illustration (such as graphs, diagrams, etc.) from any source, published or unpublished, including the internet.

2. Un-credited improper paraphrasing of pages paragraphs (changing a few words phrases, or rearranging the original sentence order)

3. Credited verbatim copying of a major portion of a paper (or thesis chapter) without clear delineation of who did wrote what. ( Source: IEEE, The institute, Dec. 2004)

4. I have made sure that all the ideas, expressions, graphs, diagrams, etc., that are not a result of my work, are properly credited. Long phrases or sentences that had to be used verbatim from published literature have been clearly identified using quotation marks.

5. I affirm that no portion of my work can be considered as plagiarism and I take full responsibility if such a complaint occurs. I understand fully well that the guide of the seminar/ project report may not be in a position to check for the possibility of such incidences of plagiarism in this body of work.

Signature of the Students: _____, _____, _____, _____

Names: Akarsh Sharma, Aakansh Togani, Rohan Tuli

Roll Nos. : E032, E049, E052

Place: Mumbai

Date:

# CERTIFICATE

This is to certify that the project entitled "Intelligent Parking Management System" is the bonafide work carried out by Akarsh Sharma, Aakansh Togani and Rohan Tuli of B.Tech, MPSTME (NMIMS), Mumbai, during the IV semester of the academic year 2019-2020 in partial fulfillment of the requirements for the Course Programming Language - III.

_____

Prof. Ameyaa Biwalkar

Internal Mentor

_____ _____

Examiner 1 Examiner 2

# Acknowledgements

We would like to thank Prof. Ameyaa Biwalkar for her suggestions and guidance which led us towards the successful completion of the project. Additionally, We would like to acknowledge the sincere help extended to us by Sughosh Sudhanvan and Hitesh Soneji.

# Table of Contents

# Introduction

This era has observed a rapid amplification in vehicle ownership along with the development in mechanization of road, bridge and highway construction. Rise in population, rapid industrialization and gradual advancements in urban areas have given rise to various obstructions related to traffic control and vehicle owner identification. There is a necessity of an intelligent online based system that reprimands those offenders. Our project comprises of a web application that dynamically handles user database, where the user are civilians who can check if they have been allotted any kind of penalty for any offense and also pay the required fine through an online payment gateway. Respected government officials have been allotted unique service through which they can access the ANPR software, which would help them allot fines to violators by simply uploading a number plate image of the vehicle. An ANPR system identifies the characters and numbers present on the license plate and stores them in a string. This allows the authorities to match the string in the pre-existing database and seamlessly, add a fine to the profile of the violator.

The website has a user-friendly user interface, the web application provides user services such as to check existing records, new penalty allocation, payment gateway and can also submit queries through the ‘Contact Us’ page.

The website has a different interface for government officials which allows them to reprimand violators by just uploading the picture of the vehicle’s license plate.

ANPR system is a smart program which can overcome limitations like illumination, speed of car, non-uniform license plate by pre-processing techniques such as adaptive thresholding and Gaussian blur. Further to remove noise such as text which is not the part of the number plate required are being removed via mathematical equations.

# Softwares and API

Our project consists of two components namely, the website and the Automatic Number Plate Recognition System.

For the website we have used the following software:

- **CSS**
  CSS stands for Cascading Style Sheets. CSS describes how the HTML elements are to be displayed on screen, paper, or in other media. It can control the layout of multiple web pages all at once. So, we have made use of CSS style on our webpages which have been written using HTML5.

- **JavaScript**
  JavaScript is a very powerful and high-level client-side scripting language. It is a just-in-time compiled programming language which is used mainly for enhancing the interaction of a user with the webpage. In other words, JavaScript allows us to make our webpages livelier and more interactive. We have used JavaScript mainly for:

  1. **Form Verification**
     - JS was used in registration forms to basically verify details like the format of contact number, e-mail address, etc.

  2. **Designing Purposes**
     - JS was used to provide smooth transitioning between the HTML elements and to give alert prompt on successful execution of certain actions.

- **JQuery**
  With the help of HTML and CSS we can only build visually appealing static web pages but it is with the help of jQuery which is a JavaScript library that we can add dynamic effects to the static web page elements and make them lively. jQuery helps us to perform several JavaScript tasks much faster and more simply.

  We have made use of jQuery in our navigation bar which basically keeps the navigation bar fixed on the webpage as we scroll down.

- **HTML5**
  HTML5 is the latest version of Hypertext Markup Language, the code that describes web pages. It's actually three kinds of code: HTML, which provides the structure; Cascading Style Sheets (CSS), which take care of presentation; and JavaScript, which makes things happen.

  We have used HTML5 to write our base source code on which, all other web designing API's are applied to

- **Bootstrap:**
  Bootstrap is an open-source CSS framework which helps us to design websites faster and in an easier manner. It mainly includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels, etc. Bootstrap also gives you support for JavaScript plugins and it makes use of JavaScript to implement some of its features. Some of the bootstrap elements that we have used are:

  1. **Container**
     - We have made use of the container class to justify the contents of our page so that it becomes appealing to the end user. It has also been used to set the content's margins dealing with the responsive behaviour of the layout and to create boxed content.

  2. **Well**
     - We have made use of the well class to simply add a rounded border around an element with a gray background color and some padding

  3. **Img-circle**
     - We made use of the img-circle class to shape the image to a circle just for beautification purpose.

  4. **Glyphicons**
     - We have used some of the bootstrap glyphicon classes to add icons in our project wherever they were required.

For the Automatic Number Plate Module we have used the following softwares:

- **Python:**
  To implement this algorithm, we have used Python for coding this algorithm. Python is a versatile and robust language which contains the powerful in built library of opencv and numpy. These two libraries in conjunction allow us to create the desired software. Opencv contains several Computer Vision and Machine Learning Libraries which alleviates tedious coding. The code for our project is written in PyCharm IDLE.

  - **NumPy:**
    NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
    We have used the NumPy library to store the digital images in numpy array which store image intensity values for every pixel.

- **OpenCv:**
  OpenCV is used for all sorts of image and video analysis, like facial recognition and detection, license plate reading, photo editing, advanced robotic vision, optical character recognition, and a whole lot more.

  In OpenCV we have used the following functions:
  - findNearest(): For each input vector (a row of the matrix samples), the method finds the k nearest neighbors. In case of regression, the predicted result is a mean value of the particular vector's neighbor responses. In case of classification, the class is determined by voting.

  - drawContours(): Calculates all of the moments up to the third order of a polygon or rasterized shape.

  - imwrite(): Reads an image from a buffer in memory. The function reads an image from the specified buffer in the memory. If the buffer is too short or contains invalid data, the empty matrix/image is returned.

  - imshow(): The window automatically fits to the image size. First argument is a window name which is a string. second argument is our image. You can create as many windows as you wish, but with different window names.

  - waitkey(): cv2.waitKey() is a keyboard binding function. Its argument is the time in milliseconds. The function waits for specified milliseconds for any keyboard event. If you press any key in that time, the program continues. If 0 is passed, it waits indefinitely for a key stroke. It can also be set to detect specific key strokes like, if key a is pressed etc which we will discuss below.

  - cvtColor(): The function converts an input image from one color space to another. In case of a transformation to-from RGB color space, the order of the channels should be specified explicitly (RGB or BGR). Note that the default color format in OpenCV is often referred to as RGB but it is actually BGR (the bytes are reversed). So the first byte in a standard (24-bit) color image will be an 8-bit Blue component, the second byte will be Green, and the third byte will be Red. The fourth, fifth, and sixth bytes would then be the second pixel (Blue, then Green, then Red), and so on.

  - GaussianBlur(): We have to specify the width and height of the kernel which should be positive and odd. We also should specify the standard deviation in the X and Y directions, sigmaX and sigmaY respectively. If only sigmaX is specified, sigmaY is taken as equal to sigmaX. If both are given as zeros, they are calculated from the kernel size. Gaussian filtering is highly effective in removing Gaussian noise from the image.

- adaptiveThreshold(): the algorithm determines the threshold for a pixel based on a small region around it. So we get different thresholds for different regions of the same image which gives better results for images with varying illumination.

- **OS:**
  Python - OS Module. It is possible to automatically perform many operating system tasks. The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc.

- **Django:**
  Django has been used in both our modules. It creates an interface between the two modules thus, it is the backbone of our software.
  Django is an open-source framework for backend web applications based on Python - one of the top web development languages. Its main goals are simplicity, flexibility, reliability, and scalability.
  Django has its own naming system for all functions and components (e.g., HTTP responses are called "views"). It also has an admin panel, which is deemed easier to work with than in Lavarel or Yii, and other technical features, including:
  - Simple syntax;
  - Its own web server;
  - MVC (Model-View-Controller) core architecture;
  - "Batteries included" (comes with all the essentials needed to solve solving common cases);
  - An ORM (Object Relational Mapper);
  - HTTP libraries;
  - Middleware support;
  - A Python unit test framework

  Our project is an integration of ANPR software (Language Python) and a web application, thus Django being the most apposite framework.

  We have imported many function from django such as auth,send_mail and messages

  - Auth: this functions helps in validation of correct credentials by matching passed data in the primitive Users database.

  - Send_mail : This method is used to send an email to a specified email. The Sender email id has been predefined in the settings.py file, thus helping us send an email confirmation in trn also checking if a correct email id has been entered.

  - Messages : This function is used to prompt the user that he has filled some fields incorrectly.

Apart from imported function we have used essentials .py file integrated with our project:

- Forms.py : This .py file is used to make a form in the template to accept information along with images to be stored in the database.

- Models.py: This .py files is used to define our database's fields along with their type and certain attributes like default value, blank=True etc

# Methods Implemented

## Web Application Methods:

- **def home():**

This functions renders the homepage of the website.

- **def register():**

This functions stores all needed details entered by the user to the database created in the models.py file.

Secondarily checks if all the blanks are filled. If the password field does not match with the confirmation password field it return an alert via messages.

Also automatically sends a confirmation email from anpripms@gmail.com to the entered email.

If all registrations fields are inputted correctly the user is redirected to the userhomepage, else gets a message to fill the fields appropriately and is redirected back to the registration page.

- **def login():**

This function authenticates the login credentials entered by the user from a pre-existing database. If credentials are valid the user is redirected to the userhomepage and if the credentials do not match with any of the pre-existing data a alert message is given to the user implying “Invalid Credentials”

This functions also checks if the credentials used to log in are of any SuperUser, if they match the super user is redirected to the adminhomepage where he can use the power to allocate fine to the registered civilians.

- **def contact():**

This functions is called when a user clicks on the “Contact Us” link. Here the user is allowed to submit queries or suggestions to the Publisher of the Web Page. Email Ids have been specified in Team section of the website.

- **def alpr():**

This method can be only invoked by a SuperUser this method is responsible for integrating the ANPR software with the website. When a SuperUser fills an application to allocate a fine to the registered user, along with an attachment of the image captured by the SuperUser, the ANPR program fetches the image from the online database, runs itself and returns a String Type variable which is then matched with a registered user. Thus automating the entire process of reprimanding a traffic rule violator.

- **def current_fine():**

This method checks if the current user has been allocated any fine and return appropriate fine allocated, it further gives the option to the user to redeem the fine by paying through an online payment gateway.

- **def fine_upload() & def fine_upload1() :**

Since the image passed to the ANPR program returns a string value which has to be stored in the same instance of the user database, thus these functions are used to do the following steps:

  - Finds the image path by using inbuilt functions like .path which is then passed to ANPR function
  - Retrieves the string value returned by the ANPR program
  - Saves the returned string value in the respective user database by using inbuilt instance._Fieldname in the respected user database.

- **def adminhome() & def userhome()**

These methods render their respective html pages.

## Automatic Number Plate Recognition (ANPR) Methods:

To begin with, all the five major functions are being called in the **main()** function

- **loadKNNDataAndTrainKNN():**

Function which is basically responsible for loading the classification.txt and flattened_images.txt file, followed by the training of KNN classifier algorithm. If the files are corrupted or the KNN training fails for some reason the loadKNNDataAndTrainKNN () returns a False value and accordingly the program prints the error message.

- **preprocess():**

In the preprocess**()** function firstly we are converting the original input image into a grayscale image using cv2.cvtColor function. Then we are using the cv2.GaussianBlur function on the grayscale image in which the image is convolved with a Gaussian filter. The Gaussian filter helps to remove any high frequency component from the image i.e. it smoothens the image. We have found out through certain experimentations that Gaussian filter is the most suitable low pass filter for our image processing requirement.
Once the grayscale image has been filtered we apply adaptive thresholding through cv2.adaptiveThreshold function on the filtered image for edge detection and to basically simplify the representation of our original input image into something that is more meaningful

and easier to analyze for us. Thus, we are simply implementing segmentation here. After completing the filtering and thresholding operations we are returning the grayscale and thresholded image output back to the main**()** function where the preprocess**()** function was called.

- **PossibleCharsInScene():**

The main objective of the PossibleCharsInScene**()** function is to identify all the possible characters present in the thresholded image and store them in a list. Firstly, we are creating an empty list 'listOfPossibleChars' which would store the possible characters and we are creating a copy of the thresholded image because the operations used in this function are destructive meaning it manipulates the image that we pass in them. Then we are making use of the cv2.findContours() function to find the contours in the image. A contour basically refers to the outline of an object. So, in the cv2.findContours() function we are passing three parameters. The first parameter is the image whose contours have to be found. The second parameter is cv2.RETR_TREE which tells OpenCV to compute the hierarchy (relationship) between contours. The third parameter is cv2.CV_CHAIN_APPROX_SIMPLE which tells OpenCV to compress the contours in order to save space. The cv2.findContours() function gives us a list of contours that it has found which we are storing in 'contours'. After obtaining the contours we are traversing through the entire contours list and passing each element of the contours list in the check**()** function of the PossibleChar class. The check**()** function is ensuring that if the contours list element satisfies certain conditions that have been laid out by us, it would be appended in the 'listOfPossibleChars' list. We are also appending the contours of all the possibleChar which satisfy the check**()** function into contours which is later used to draw the contours of all the possible characters in the scene with the help of cv2.drawContours function. Finally, we are returning the updated 'listOfPossibleChars' list to the main() function where the PossibleCharsInScene**()** function was called.

- **ObtainListOfMaatchingChars():**

In the ObtainListOfMatchingChars**()** function, firstly we are creating an empty 'listOfListsOfMatchingChars' list which would store the list of matching characters present in the image. Now there can be different form of text which could be present in our image but we only want the license plate number text, thus we are trying to match the characters with each other and then cluster all the matching characters in the 'listOfListsOfMatchingChars' list. We have laid out conditions which would supervise the matching of characters with each other such that the license plate number text gets clustered into one distinct cluster. Some of the user defined functions which are implementing the above mentioned conditions include distanceBetweenChars**()** function and angleBetweenChars**()** function. So in ObtainListOfMatchingChars**()** function we have two for loops in which each possibleChar of the 'listOfPossibleCharsInScene' list is matched with each character in the 'listOfPossibleCharsInScene' list and if a match is found then that character element is appended to the 'listOfMatchingChars' list which is an empty list that is created every time we enter the first for loop and it stores the matching characters for the entire first for loop's iteration. As each first for loop's iteration gets completed we are appending the updated 'listOfMatchingChars' list in the 'listOfListsOfMatchingChars' list. Finally, we are returning

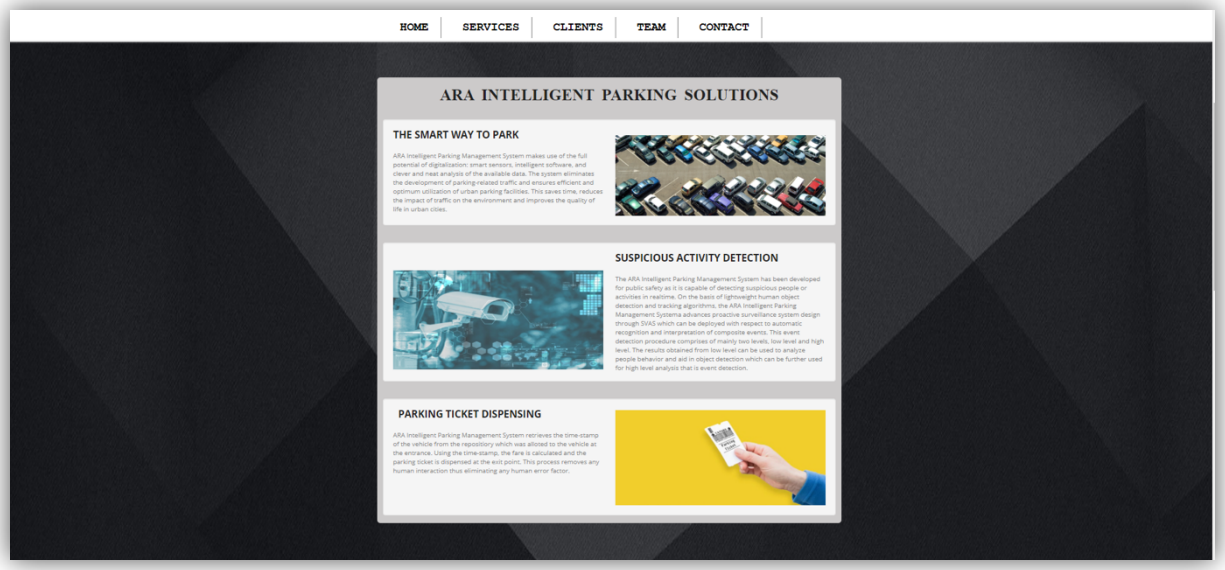
the updated 'listOfListsOfMatchingChars' list to the main**()** function where ObtainListOfMatchingChars**()** function was called

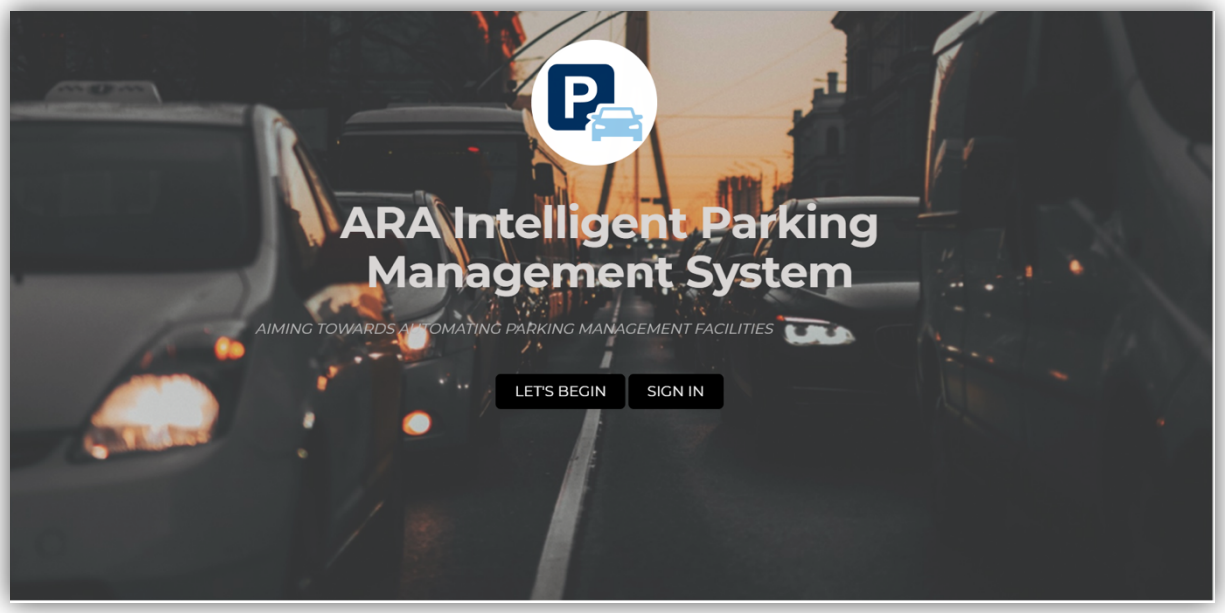
- ## recognizeCharsInPlate():

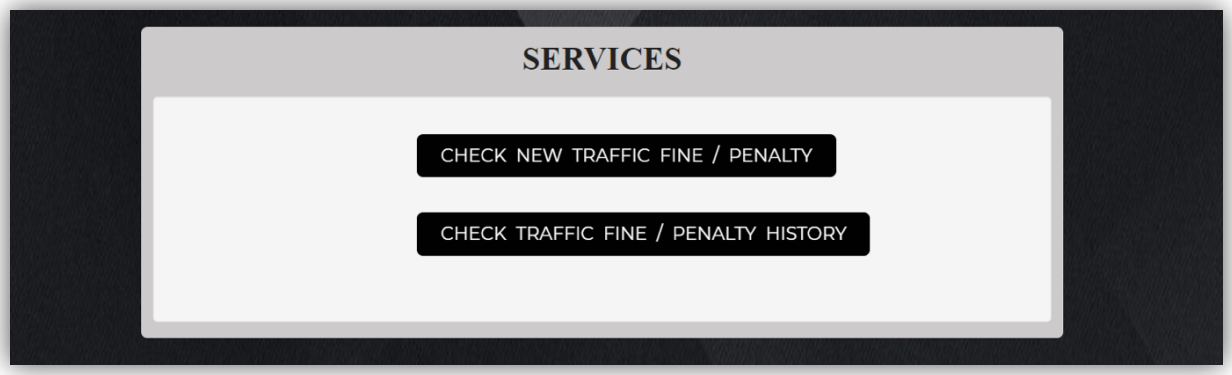
In the **recognizeCharsInPlate()** function we are initially creating an empty string 'strChars' which would later store the recognized license plate characters. Then we are traversing the 'listOfMatchingChars' list and we are trying to highlight each list element as the Region of Interest (ROI) with the help of cv2.rectangle function followed by its extraction in 'imgROI'. The ROI is then resized with the help of cv2.resize function and later reshaped through the .reshape function. We are resizing and reshaping the obtained ROI in order to ensure its accurate recognition by the KNN classifier. After resizing and reshaping the ROI we are applying np.float32 conversion on the 'npaROIResized' where np.float32 is the single precision float: sign bit, 8 bits exponent, 23 bits mantissa. Once the np.float32 conversion is complete, we are passing the converted 'npaROIResized' to the kNearest.findNearest() function with the value of k=5 where k is the number of used nearest neighbours. The kNearest.findNearest() function finds the nearest neighbours of the 'npaROIResized' data wrt the trained data and helps to classify the characters accordingly. The output of the kNearest.findNearest() function is being stored in the 'npaResults' list which contains the ascii values of all the recognized characters, so we are using the chr() function to convert the ascii values into characters and concatenating these characters in the 'strChars' string. Once the entire 'listOfMatchingChars' list is traversed we are returning the updated 'strChars' string to the **main()** function where **recognizeCharsInPlate()** function was called.

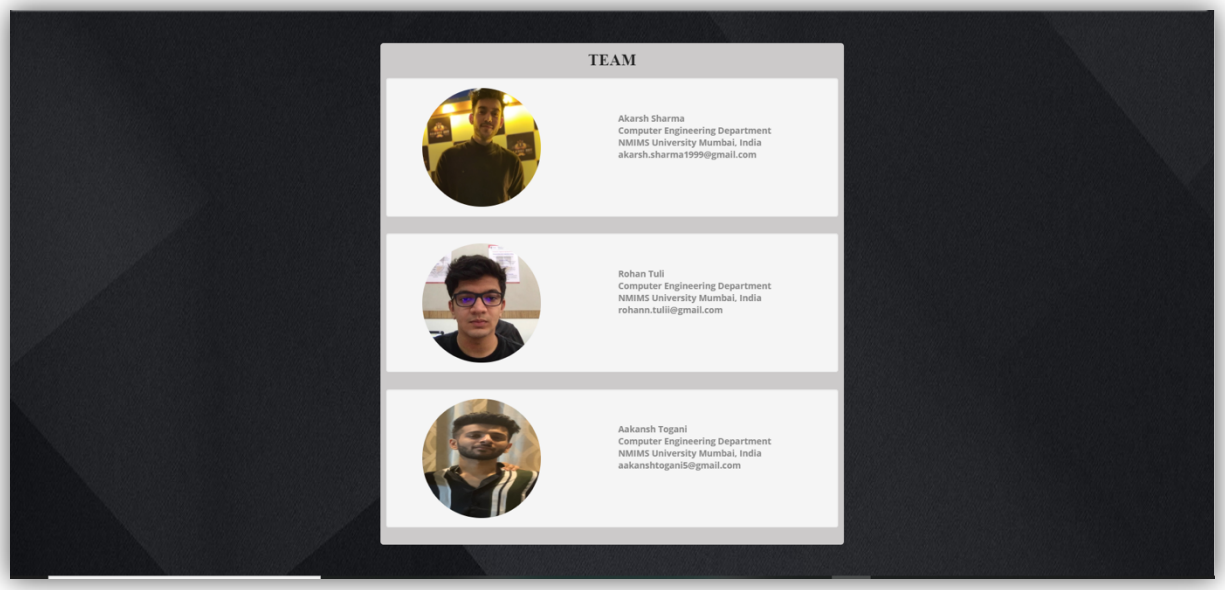
# Screenshots

## 1. Home Page

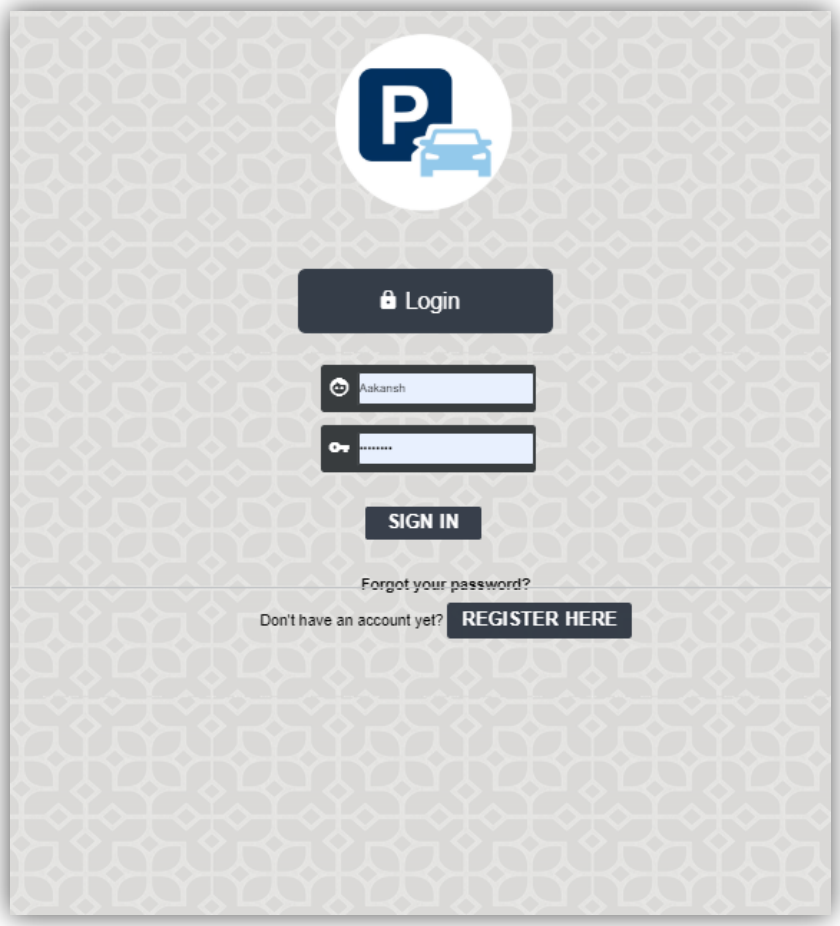


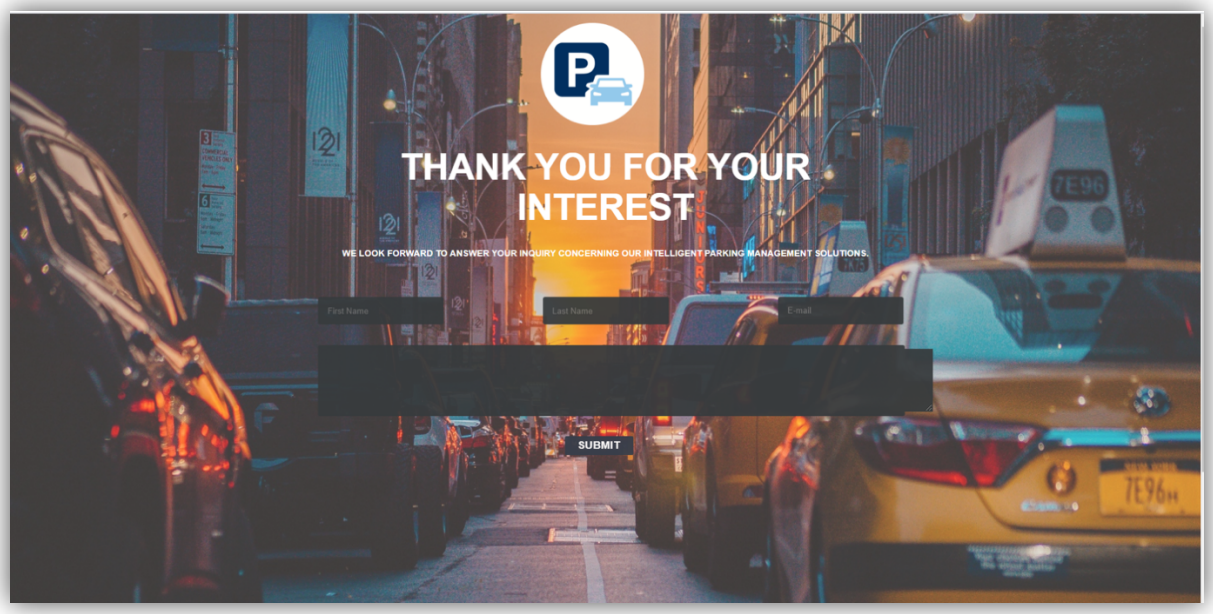

SERVICES
CHECK NEW TRAFFIC FINE / PENALTY
CHECK TRAFFIC FINE / PENALTY HISTORY


TEAM
Akarsh Sharma
Computer Engineering Department
NMIMS University Mumbai, India
akarsh.sharma1999@gmail.com
Rohan Tuli
Computer Engineering Department
NMIMS University Mumbai, India
rohann.tuli@gmail.com
Aakansh Togani
Computer Engineering Department
NMIMS University Mumbai, India
aakanshtogani5@gmail.com

## 2. Login Page

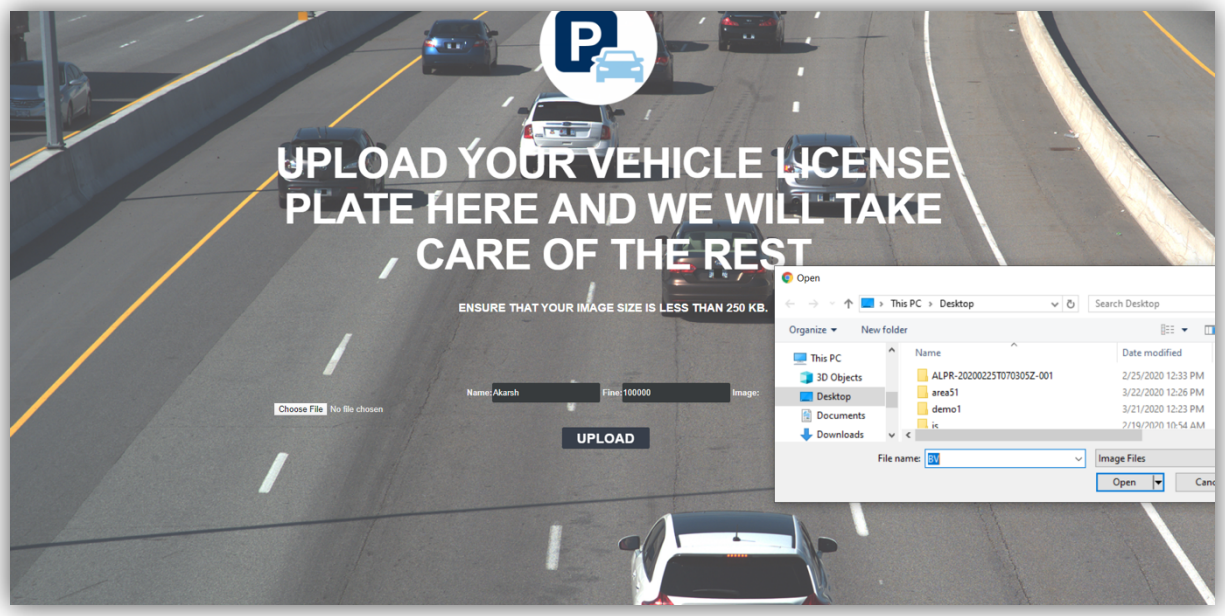



## 3. Contact us

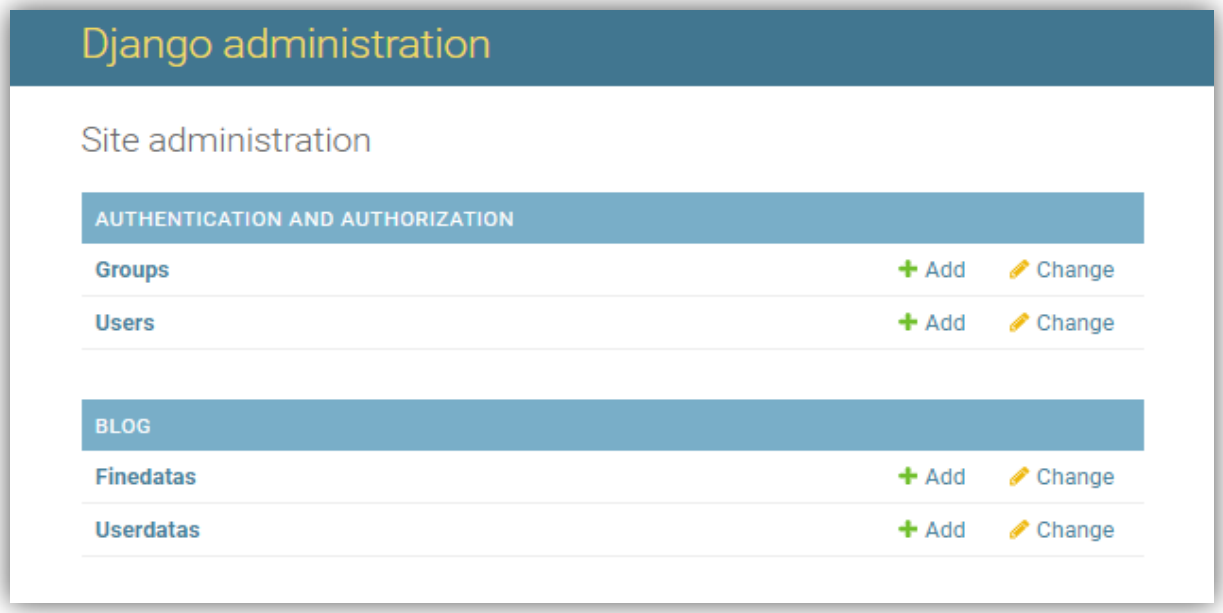


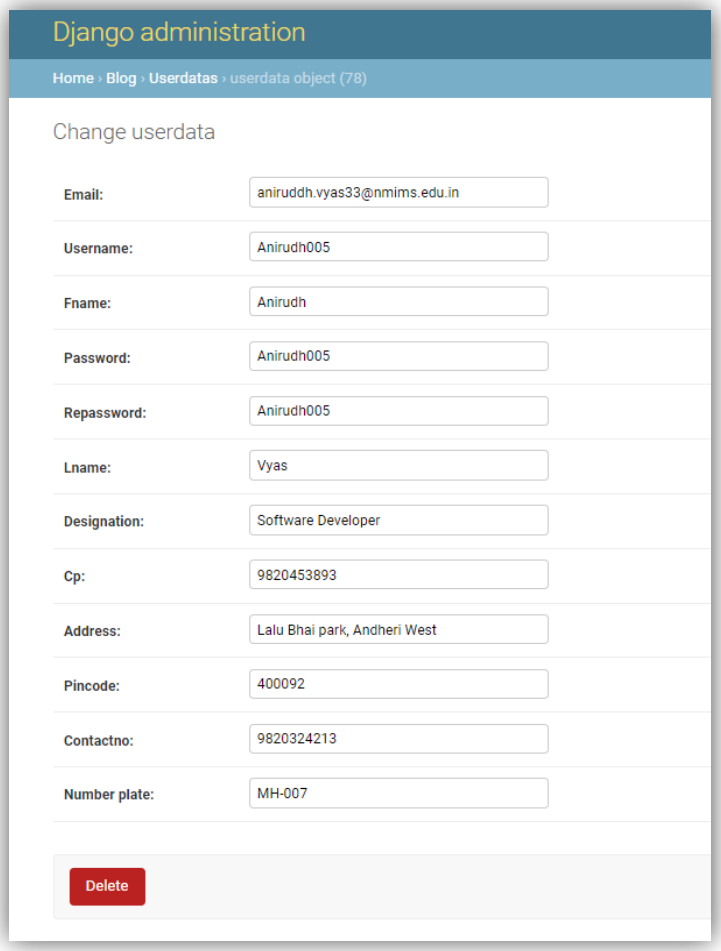
# I. Admin (SuperUser):

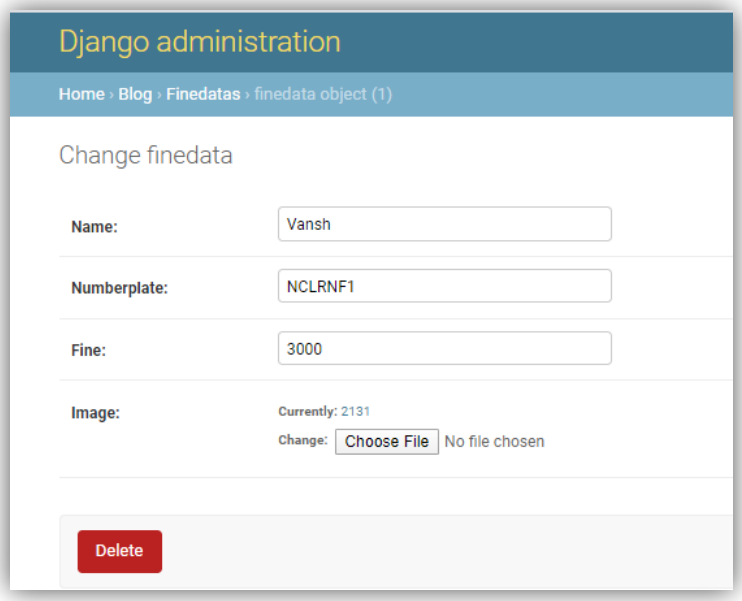
## 4. Upload fine




## 5. Django Database

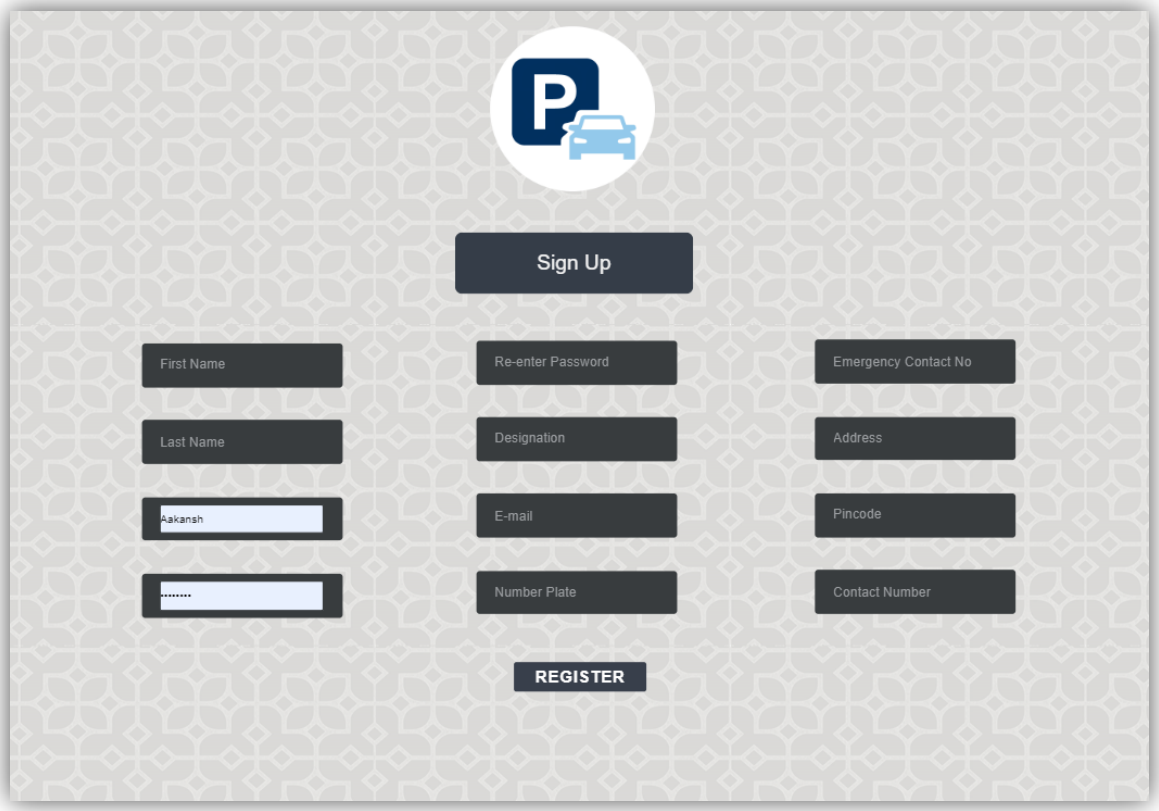



Django administration
Home › Blog › Userdatas › userdata object (78)
Change userdata
Email: aniruddh.vyas33@nmims.edu.in
Username: Anirudh005
Fname: Anirudh
Password: Anirudh005
Repassword: Anirudh005
Lname: Vyas
Designation: Software Developer
Cp: 9820453893
Address: Lalu Bhai park, Andheri West
Pincode: 400092
Contactno: 9820324213
Number plate: MH-007
Delete


Django administration
Home › Blog › Finedatas › finedata object (1)
Change finedata
Name: Vansh
Numberplate: NCLRNF1
Fine: 3000
Image: Currently: 2131
Change: Choose File No file chosen
Delete

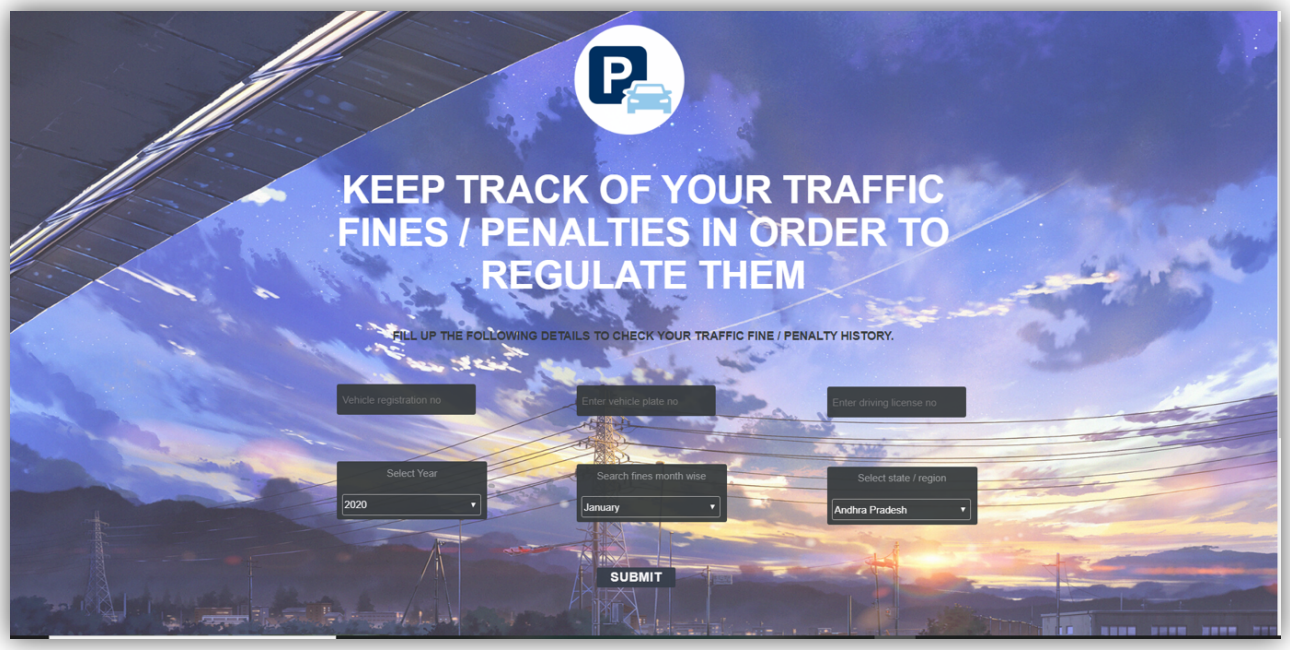
# II. User:

## 6. Register page

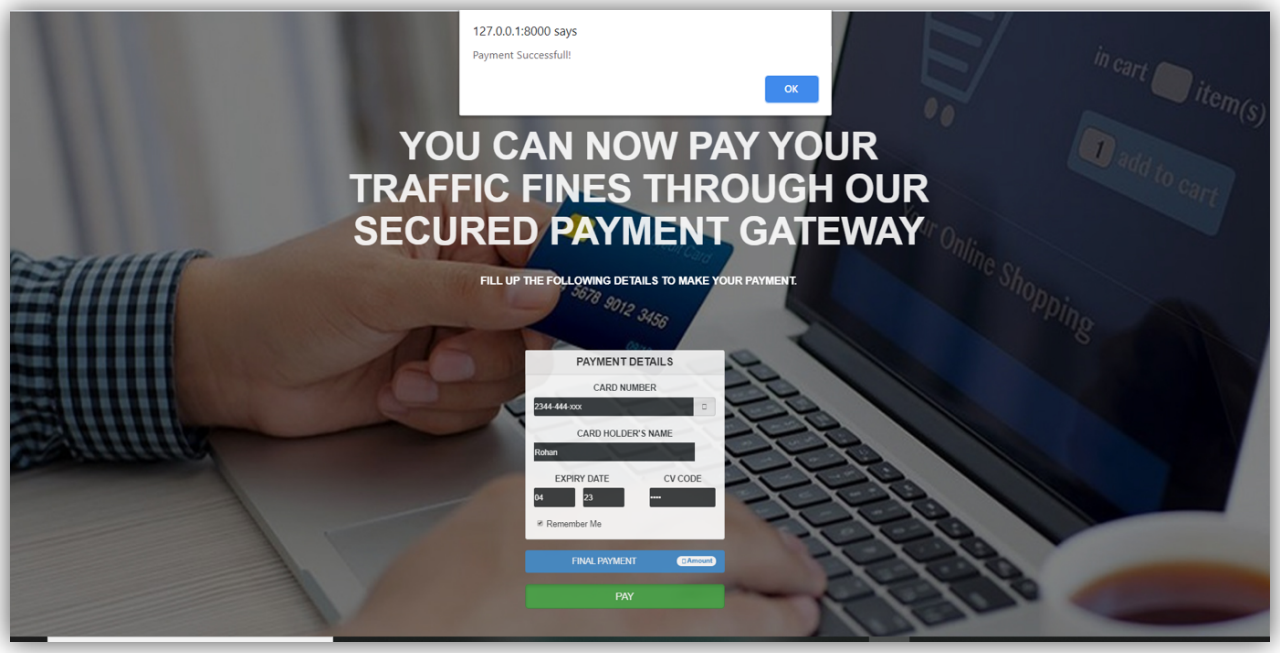



## 7. Current fines

## 8. Previous fines




## 9. Payment page

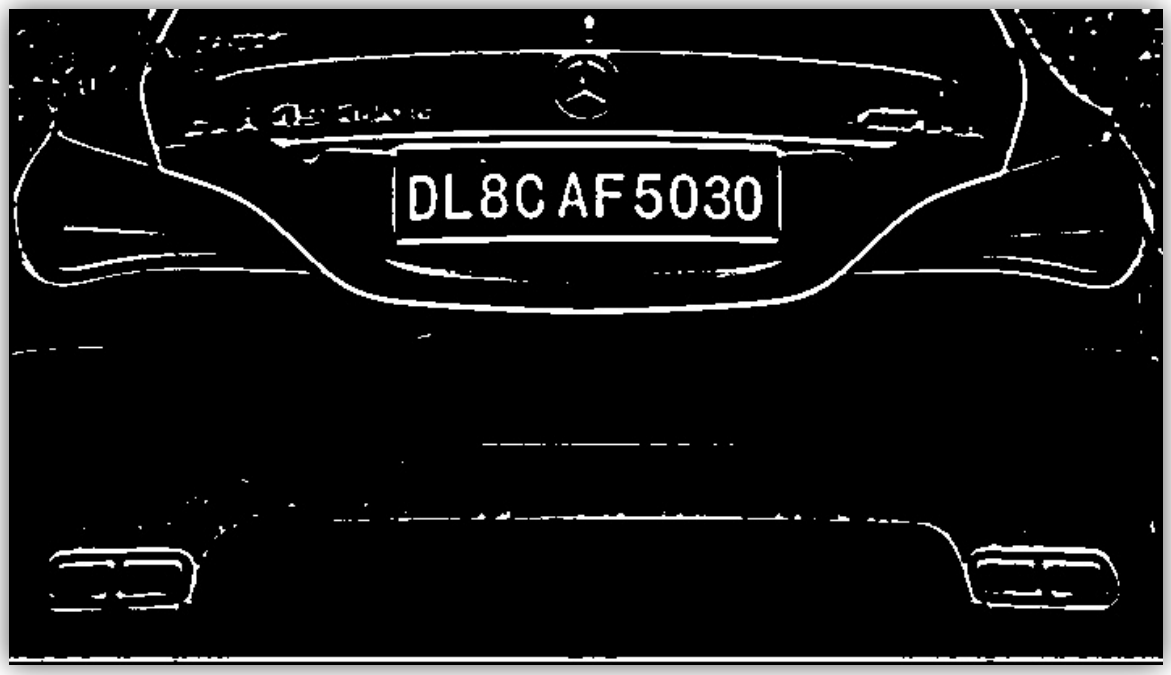


# Automatic Number Plate Recognition

## 10.Original Image


## 11.Grayscale Image

## 12.Binary Image

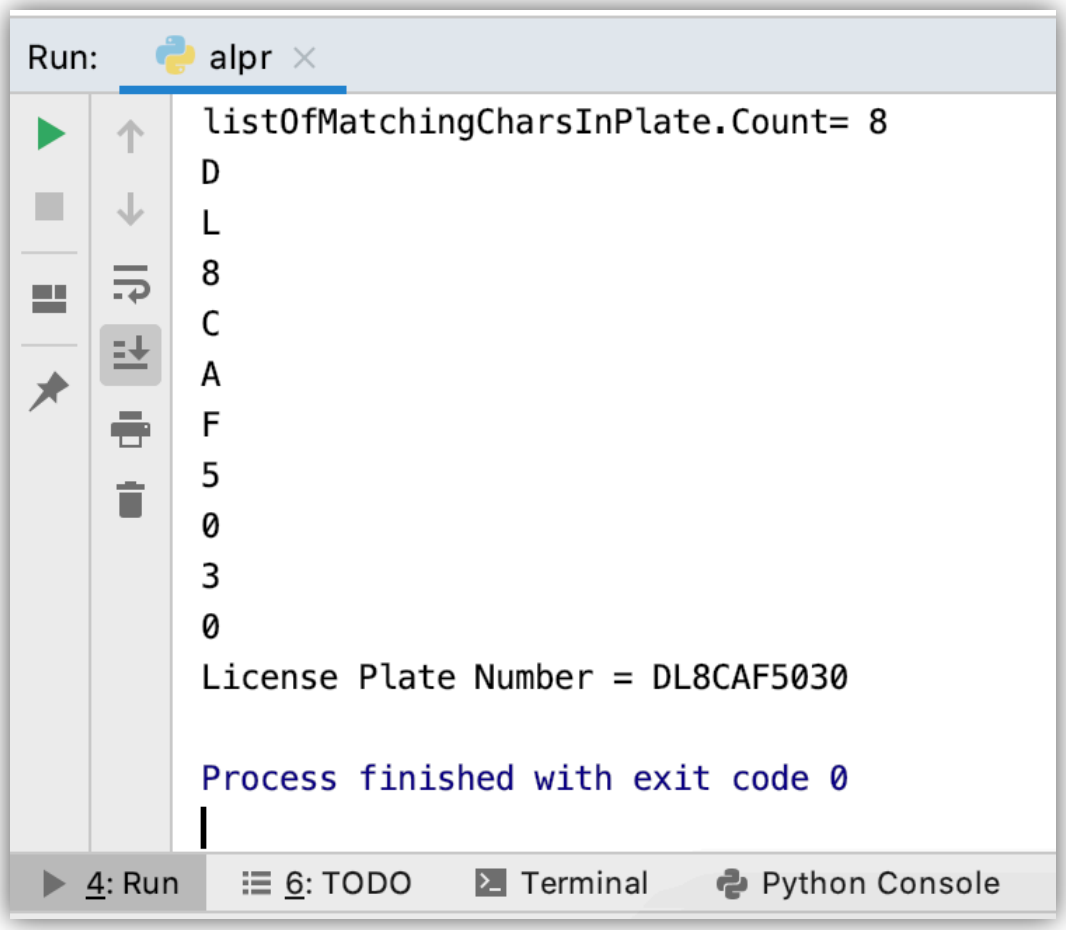



## 13.Contour Construction

## 14.List of matching characters


## 15.ROI extraction and classification

## 16.Final output



# Conclusion & Future Scope

Since traffic control is a very tedious job, our projects aims at making the entire system with minimal human-interactions. The web application takes care of allotting fines to violators flawlessly. Officials will take minimum amount of time to allot fines without manually surfing through the database, making their work hassle-free while the civilians will be assured with photographs and time as in when and where the fine was allocated to them. With also an option to redeem the fine, through an online payment gateway.
The result analysis of our ALPR program depicts that the value of the pre-processing parameters play a significant role in determining how accurately our ALPR system can give us the desired output.
Factors such as illumination, vehicle shadow, noise, non-uniform characters present on the number plate and font of the characters also affects the performance of the ALPR system.
Another matter of concern is how well you are training your ML algorithm and what kind of a classifier is being deployed

Further Implications to enhance security can be to introducing Facial recognition at the parking lot entry point to identify and validate the driver. An application can also be developed which would guide the user to the closest parking lot and also direct the user to the parking spot inside the parking lot. Also, an extended feature of the application can be introduced where in the user can pre-book the parking spot. To utilize the parking area efficiently, the system must recognise a vehicle parked in an incorrect orientation thus, alarming the driver at real time.

LPDR can tremendously reduce human interaction and increase efficiency. It can deployed in several fields such as Law Enforcement, Speeding cameras, Traffic control, Electronic toll collection and Intelligent parking management system. Enhancing pre-processing techniques which can further simplify degraded pictures, thus usable by the LPDR software. The LPDR software can be upgraded by adding more feature detection techniques which can identify the state from which the automobile is also, License plate recognition allows convenient search and retrieval of stolen vehicles apart from identifying any unauthorized within the parking facility premises.

# Societal Applications

In this project, we have highlighted the significance of an Intelligent Parking Management System as a crucial step towards smart transportation in urban environments. This system is highly capable of providing real time information of parking spot status.

The implementation of video analytics features allows continuous monitoring of all the vehicles present at a parking facility thus, any suspicious activity can be identified immediately alarming the authorities to take suitable action at real time.

License plate recognition allows convenient search and retrieval of stolen vehicles apart from identifying any unauthorized within the parking facility premises. The usage of such intelligent parking management systems in places like malls and shopping complexes can enhance the entire experience for shoppers. Data obtained from video surveillance can provide insights into the peak traffic times thus helping the shopping stores and the parking attendants to be prepared accordingly during peak hours in advance.

The principal objective of our project is to implement an intelligent parking management system based on video analytics which is safe, resourceful, efficacious and capable of being effective in finding a vacant parking spot facility with minimal human interaction.