A Project Report on

Parking Monitoring System

Submitted in partial fulfilment of the requirements for
the award of the degree of

Bachelor of Engineering
in
Computer Engineering

by
Shoaib Khan(17202002)
Ranjit Desai(16102003)
Bhaven Kakade(17202010)

Under the Guidance of
Prof. Amol Kalugade

Department of Computer Engineering
A. P. Shah Institute of Technology
G.B.Road, Kasarvadavli,Thane(W),Mumbai.
UNIVERSITY OF MUMBAI
Academic Year 2019-2020

Approval Sheet

This Project Report entitled “***Parking Monitoring System***” Submitted by “***Shoaib S. Khan (17202002)”, “Bhaven Kakade(17202010)”, “Ranjit Desai (16102003)”*** is approved for the partial fulfillment of the requirement for the award of the degree of **Bachelor of Engineering** in **Computer Engineering** from **University of Mumbai.**

Prof. Amol Kalugade
Guide

Prof. Sachin Malve
Head Department of Computer Engineering

Place:A.P.Shah Institute of Technology, Thane
Date: 18/06/2020

Certificate

This is to certify that the project entitled "**Parking Monitoring System**" submitted by "**Shoaib S. Khan(17202002)**", ***"Bhaven Kakade(17202010)", "Ranjit Desai (16102003)"*** for the partial fulfillment of the requirement for award of a degree Bachelor of Engineering in Computer Engineering.,to the University of Mumbai,is a bonafide work carried out during the academic year 2019-2020.

Prof. Amol Kalugade
Guide

Prof. Sachin Malave	**Dr. Uttam D.Kolekar**
Head Department of Computer Engineering	Principal

External Examiner(s)

1.

2.

Place:A.P.Shah Institute of Technology, Thane
Date: 18/06/2020

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Shoaib Khan, 17202002)

(Bhaven Kakade,17202010)

(*Ranjit Desai, 16102003)*

Date: 18/06/2020

Table of Contents

Table of Contents 4

Chapter 1 : Project Conception and Initiation 6

- Research paper search: 6
- Research Paper finalization: 9
- Parking Monitoring System 10
- Abstract 11
- Objectives 12
- Literature Review 13
- Problem Definition 15
- Technology Stack 16
- Benefits for environment 17
- Benefits for society 18
- Applications 19

Chapter 2 : Proposed Design 20

- Proposed System: 20
- Flow of Modules 22
- Activity Diagram 23
- Use Case Diagram 24
- Description Of Use Case 25
- Modules 27
  - Module 1 :Authentication 27
  - Module 2 : Login and Register 27
  - Module 3 : Firebase OCR 27

Chapter 3 : Project Implementation 29

- Project Hierarchy 29
- main.dart 29
- Module 1 : Authentication 30
  - wrapper.dart 30
  - home.dart 30
  - authenticate.dart 31
- Module 2 : Login and Register 32
  - user.dart 32
  - sign_in.dart 32
  - register.dart 35
- Module 3 : Firebase OCR 37

Chapter 4 : Testing 39

- Design of Test Cases 39
- Testing 39
  - Path Coverage Testing 39
- Testing of inputs 39
- Finding Contours 40
- Test case to crop unwanted parts of image(i.e everything except the number plate) 41

Testing Flutter App OCR 42

Chapter 5 : Results and Analysis 44

Results of Flutter OCR 44

OCR on image 44

Authentication and Users 45

Login 46

Register 47

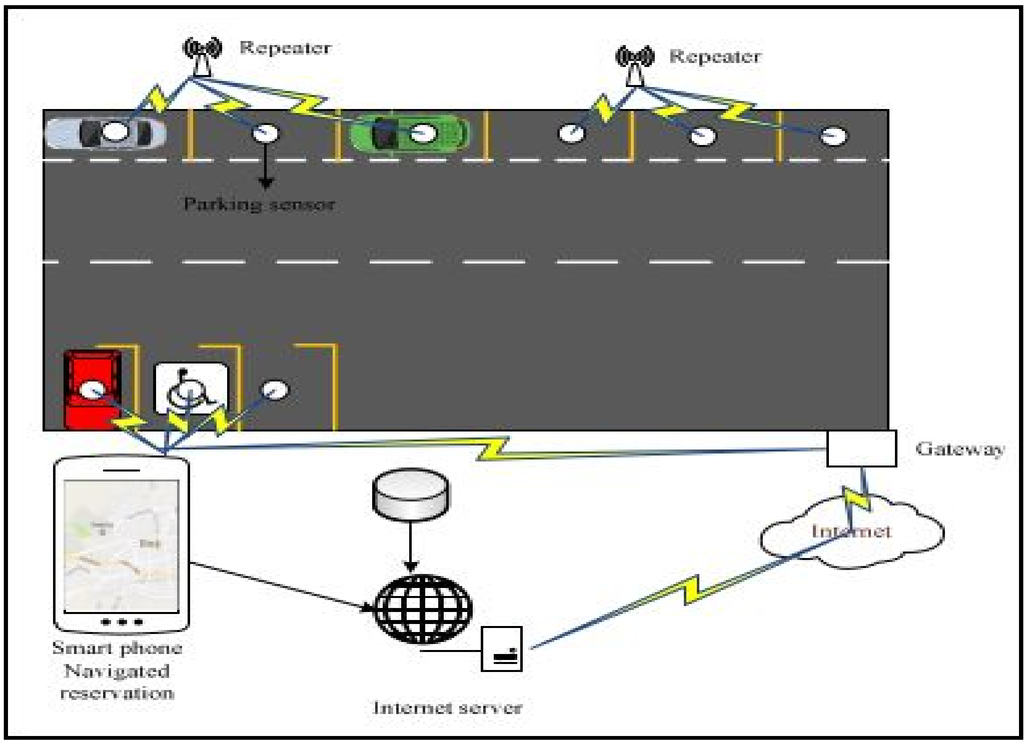
Bibliography 48

Chapter 1 : Project Conception and Initiation

Research paper search:

A) A Navigation and Reservation Based Smart Parking Platform Using Genetic Optimization for Smart Cities

1. This paper involves small devices that transmit data using Internet-of-Things.
2. Using the Genetic Algorithm, the closest current location is determined for parking.
3. This method is tested on different scenarios and accurate results were obtained.
4. The system consists of 2 modules. i)Hardware ii) Software.

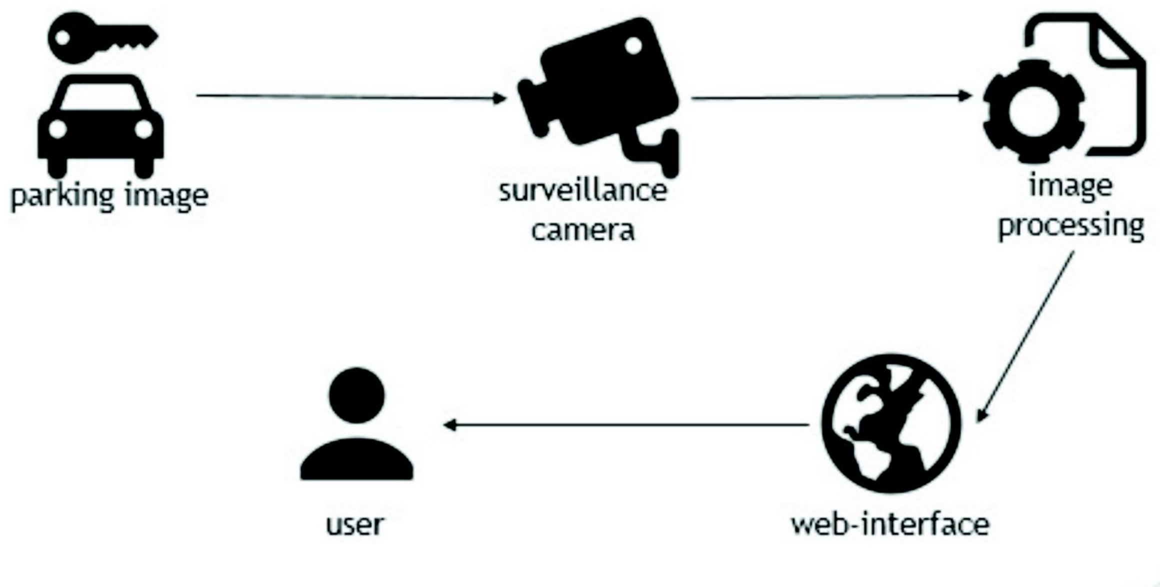


5. **Hardware :**
    a. This module acquires the condition of the parking slots and sends this data to the internet. The second module consists of the software part to determine the nearest free parking slots.
    b. Each street has a gateway device.
    c. 802.15.4 transceiver to send its parking condition to the gateway.
    d. In the gateway node, the acquired date is transferred to the internet server via GPRS module.
    e. A magnetic sensor is used to determine whether or not a parking slot is free or occupied.
6. **Software :**

a. Genetic Algorithm calculates a route from our current position to the nearest free parking slot iteratively.
b. Parking positions on the roads are saved as numbers.
c. A database is maintained which consists of Longitude and Latitude of all parking positions.
d. After the genetic algorithm is run, a route to the nearest free parking lot is obtained and this lot can be reserved for this user.

B) **Developing a Parking Monitoring System Based on the Analysis of Images from an Outdoor Surveillance Camera**

1. It allows the detection and tracking of cars in a parking lot, using collected historical data to predict availability of parking during the day based on data mining techniques.
2. This system tracks availability based on an analysis of images from an outdoor surveillance camera and a real time analysis of state of the parking lot is obtained.
3. The data obtained can be used to inform drivers about available parking spaces.
4. **Working** : The system involves two components i) System for image capture(web server) and analysis ii) a Web Interface.
    a. First component i.e the system for image capture captures video and converts it into frames of images.
    b. Using image analysis algorithms, decisions on free/occupied parking spaces are made. Following image shows the architecture of proposed system :



    c. **Web server** : Provides web GUI and API. Both are implemented as a web application, which provides direct access to the database.
    d. **Image Analyzer** : Receives images from outdoor surveillance camera and transformation of perspective is performed. New state is compared with the previous state and changes are updated into the database.

e. **User Interface :** It is a web application implemented in the python programming framework, Django.
f. Web application is based on routes.
    i. Home / parking displays all registered in the system.
    ii. Parking / id / displays the status of parking numbered id.
    iii. Parking / id / change / edit page displays the parking administration screen. The administrator can manually change the state of a parking space.

C) Cross Platform Smart Reservation Based Parking System

1. The number of cars that are manufactured are increasing at an immense rate of doubles, triples, quadruples and so on.
2. A proper solution is needed and advanced technology is the key to develop and implement such ideas in the physical world.
3. With this system users can find parking spaces via an Android device, book a spot and also track car timings.
4. The proposed system has 2 sections : i) User Side ii) Developer side.
5. Two types of sensors are used, UHF and Infrared.
6. **Working :**
    a. Users have to register in order to avail parking services.
    b. As soon as the user provides credentials, the app automatically searches for available parking spaces nearby.
    c. Users can book a parking slot right away or schedule a parking slot for later.
    d. If a user chooses to book a spot right away, he needs to enter arrival and departure time which further redirects to the payment page.
    e. In case of scheduling a parking spot the user first selects a desired parking spot nearer to him and the same process continues as for the booking process.
    f. Hardware for this smart parking system consists of 3 major components. Namely, **Ultra High Frequency sensor**, **the main motherboard "NodeMCU"** and the **entry/exit motor** for the system.
    g. **NodeMCU :**
        i. NodeMCU is a microcontroller consisting of inbuilt Wi-Fi module ESP8266 along with 32bit TensilicaXtensa IX106 core clocked at 80MHz as compared to Arduino's 8 bit ATMEGA clocked at 16 MHz.
    h. **HC-SR04 :**
        i. It is an ultrasonic module that provides the non-contact measurement from the range of 2 CM to 400 CM.
        ii. The trigger input signal is of 10 micro STTL pulse.

iii. These sensors are used mainly at two places. Firstly to detect whether a car is parked in a particular slot hence requires one component per slot.
iv. Secondly at the entry and exit system to ensure whether the car has fully entered or not after generating the ticket.

i. **TowerPro SG90 Servo :**
   i. These are a 3-pole motor type with the pulse width ranging from 500-2400 microseconds which are capable of rotation.
   ii. The rotation can be approximately 90 degrees in each direction.

j. **Working :**
   i. Users will be able to search nearby parking spaces through an Android app.
   ii. System has a time limit after the completion of booking. If the user is not able to arrive within the time frame allotted then the booking will be cancelled.
   iii. A centralized server will monitor and make updates in the system as per real time scenario.
   iv. A random parking slot nearest to the users will be allotted.

Research Paper finalization:

1. The paper which was finalized is **Developing a Parking Monitoring System Based on the Analysis of Images from an Outdoor Surveillance Camera.**
2. We selected this paper because it offers a convenient way of finding parking slots.
3. An outdoor camera is used to capture live parking data and analyze video frames to find available parking spaces. It uses OpenCV to achieve this.
4. Image processing using OpenCV makes it easier and more efficient to analyze data in real time.
5. The user interface is a web application implemented using Django(a python framework) and Bootstrap.
6. It consists of routes:
   a. Home / parking displays all registered in the system.
   b. Parking / id / displays the status of parking numbered id.
   c. Parking / id / change / edit page displays the parking administration screen. The administrator can manually change the state of a parking space.

Parking Monitoring System

Abstract

This implementation involves building a Parking Monitoring System for our college parking premises using opencv. It allows us to detect and track cars in the parking slot and book a parking slot for that respective car by creating a log of the car when it enters and exits. The log is generated by noting the number plate of the car using Optical Character Recognition. When the car exits the premises, again a log will be generated stating that the car has been moved from the allotted space and the parking slot is available for a different car to be parked. The parking monitoring system tracks availability of parking based on real time outdoor camera surveillance. This system is developed to determine the location and parking number of the available parking slot and to check if the car is parked at its allotted slot.

Objectives

1. To develop a parking monitoring system capable of analyzing and providing real time data about parking space to users.
2. Using OCR(Optical Character Recognition) to analyze car number plates(this will help to keep track of vehicles entering and exiting the parking lot).
3. To build a system that will help reduce the hassle of searching for available parking spaces.
4. To analyze the parking usage based on real-time data.
5. To reduce human intervention in parking lots.

Literature Review

1. **A Navigation and Reservation Based Smart Parking Platform Using Genetic Optimization for Smart Cities**
    a. With the development of technology, smart devices are becoming more common in everyday life. The development of devices that can connect to the Internet and transmit data has been a source of inspiration for smart city designs.
    b. The common problem in our cities is the difficulty of finding free parking slots. The parking problem causes traffic to congest and people who go to work are looking for a place.
    c. In this study, a navigation and reservation based parking proposal system was developed for smart cities. The proposed method involves the development of small devices that send data to the internet using the internet of things (IoT) technology.
    d. The free parking space closest to the current location is found by genetic algorithm. The proposed method is tested for different scenarios and accurate results are obtained.

2. **Cross Platform Smart Reservation Based Parking System**
    a. Governments today are taking the most of the initiatives to educate and bring talents forward as an idea to make our country better and smart.
    b. With the concept of smart cities, on one hand, the originality of bringing ideas to the physical world with Real Time working and implementation is what the citizen's and the government is aiming for.
    c. It is said that most commuters spend more time in finding spaces for parking than driving around with the odd probability of actually getting the parking spaces for themselves.
    d. The idea here is to implement Smart Parking Solution.
    e. Smart Parking devices will be introduced to various parking spots that will be connected to the cloud and provide Real-time updates from the UHF installed sensors for available parking spaces for the user.
    f. The aim of the device is to bring ease and eliminate basic problems like traffic congestion with more practical and purposeful solutions.
    g. However certain parameters are to comply like minimum display width and processing power. The Proposed system is an "Originality" because the idea of making smart city solutions have not yet been implemented in the crowded areas or areas where getting parking spaces is difficult.

h. A User-friendly app is introduced for the operations for User from finding a parking space to booking space confirmed. The circuitry used in the whole operation is easily built and cost-effective for the organizations to implement it.

3. **Developing a Parking Monitoring System Based on the Analysis of Images from an Outdoor Surveillance Camera**
    a. In this paper, a solution for monitoring of parking availability based on computer vision is described.
    b. It allows us to detect and track cars in a parking lot, while collected historical data helps us to predict availability status of parking during the day based on data mining techniques.
    c. This system tracks availability based on an analysis of images from an outdoor surveillance camera and analyzes in real time the state of the parking complex.
    d. The system is developed to determine the number and location of available parking places and to inform the drivers.
    e. We provide an algorithm for image capture and analysis to recognize a car in a parking lot and to define parking spaces as either reserved or free.

Problem Definition

In recent years, the number of cars owned by college employees is rapidly increasing. This means they need to know in advance the availability of parking spaces in college campus. The inflow and outflow of vehicles in the college are monitored. Their availability is continuously updated in the database.

This helps faculties to get real time availability of parking slots in college campus. Android applications can be used to search and book available parking spaces.

Technology Stack

1. **OpenCV :** For detection of cars and number plates.
2. **Optical Character Recognition (OCR) :** Used for detecting alphabets and numbers in the number plate and converting them into strings.
3. **Python :** Language used for implementing implementation.
4. **Firebase :** To store real-time logs for cars.
5. **Spyder :** IDE for implementing implementation.
6. **Flutter :** Flutter is an open-source UI software development kit created by Google. It is used to develop applications for Android, iOS, Windows, Mac, Linux, Google Fuchsia and the web from a single codebase.
7. **VS Code :** Visual Studio Code is a free source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

Benefits for environment

1. Due to the availability of real-time parking system, car owners spend less time searching for parking spaces thus reducing CO2 emissions.
2. Will reduce parking search traffic on the roads thus optimizing traffic flow.
3. Less wastage of fuel.

Benefits for society

1. **Optimized parking system** – Users can find best parking slot available, thus saving time, fuel and effort.
2. **Decreased costs of management** - Due to automation, human intervention is eliminated, thus, labor cost decreases significantly.
3. **Reduction in traffic** - Traffic flow increases as fewer cars are required to drive around in search of available parking slots.
4. **Increased safety** - Using real time lot data can help detect parking violations and suspicious activity.
5. **New revenue streams** - Reward programs can be introduced to encourage repeat users. Also, payment options based on parking location can be offered to attract customers.

Applications

1. Users can access occupancy data to determine the availability of parking spots and then pay for them via their mobile phone.
2. Parking authorities can analyze parking usage based on real time data.
3. Parking space can be booked in advance.
4. Predict and sense vehicle occupancy in real time.
5. Guide the users to available parking.
6. It can also be used in the parking lot of various shopping malls.
7. Parking can be made more cost effective by implementing dynamic pricing based on demand and time.
8. Parking violations or suspicious activities can be detected using lot data.

Chapter 2 : Proposed Design

Proposed System:

Detecting Parking Spaces in a Frame :

A parking space is just a place where a car parks for a long time. So maybe we don’t need to detect parking spaces at all. We can just detect cars that don’t move for a long time and assume that they are in parking spaces. In other words, valid parking spaces are just places containing non-moving cars. So if we can detect cars and figure out which ones aren’t moving between frames of video, we can infer the locations of parking spaces

Number plate recognition

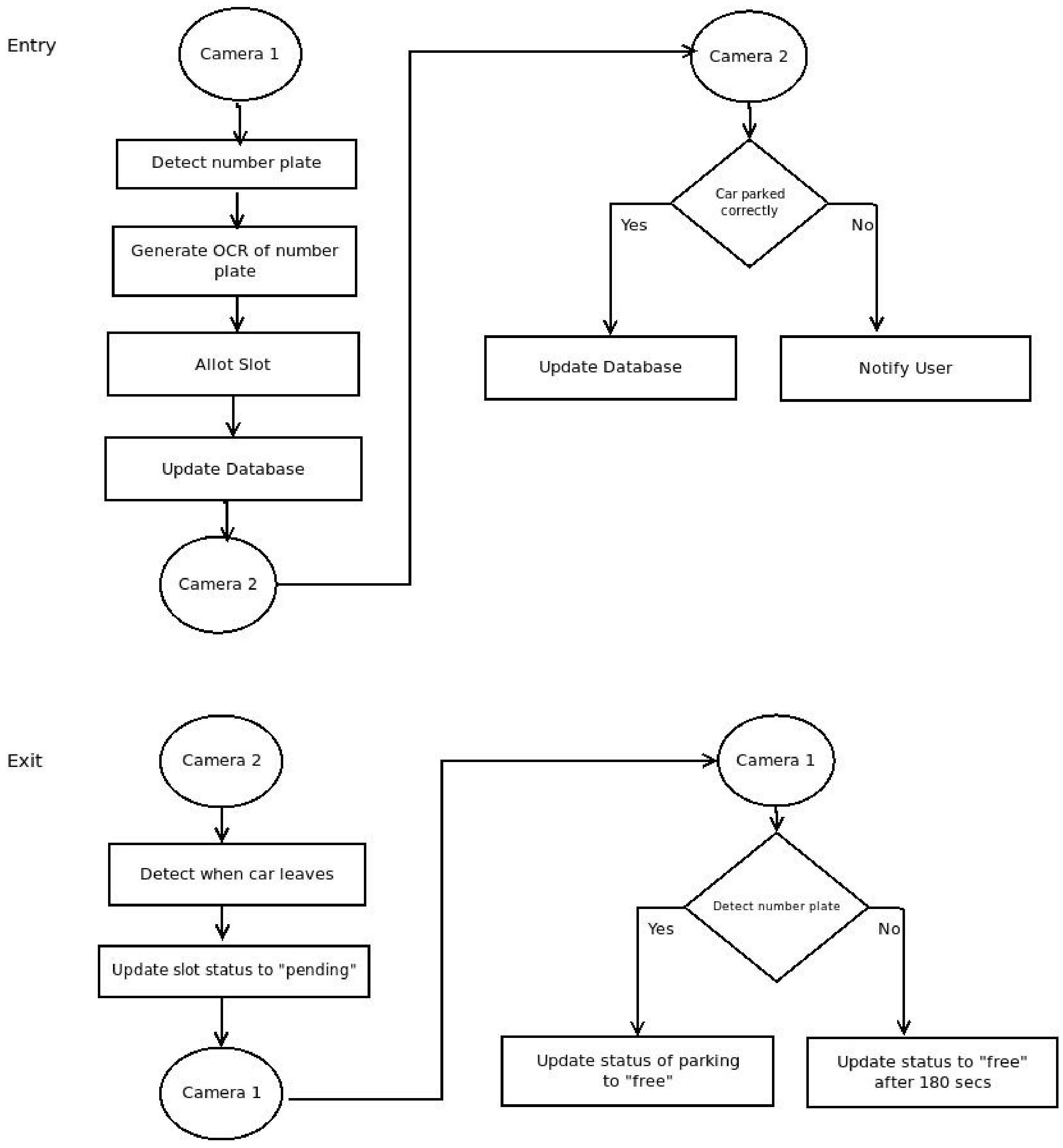
Number plate recognition is a form of automatic vehicle identification. A number plate is the unique identification of a vehicle. It is an image processing technology used to identify vehicles by their own number plates. Real time number plate recognition plays an important role in maintaining law enforcement and maintaining traffic rules. It has wide applications areas such as toll plaza, parking area, highly security areas, boarder’s areas etc. Number plate recognition is designed to identify the number plate and then recognize the vehicle number plate from a moving vehicle automatically.

Automatic number plate recognition has three major parts: vehicle number plate extraction, character segmentation and Optical Character Recognition (OCR). Number plate extraction is that stage where the vehicle number plate is detected. The detected number plate is pre-processed to remove the noise and then the result is passed to the segmentation part to segment the individually characters from the extracted number plate. The segmented characters are normalized and passed to an OCR algorithm. At last the optical character information will be converted into encoded text. The characters are recognized using Template matching. The final output must be in the
form of string of characters.

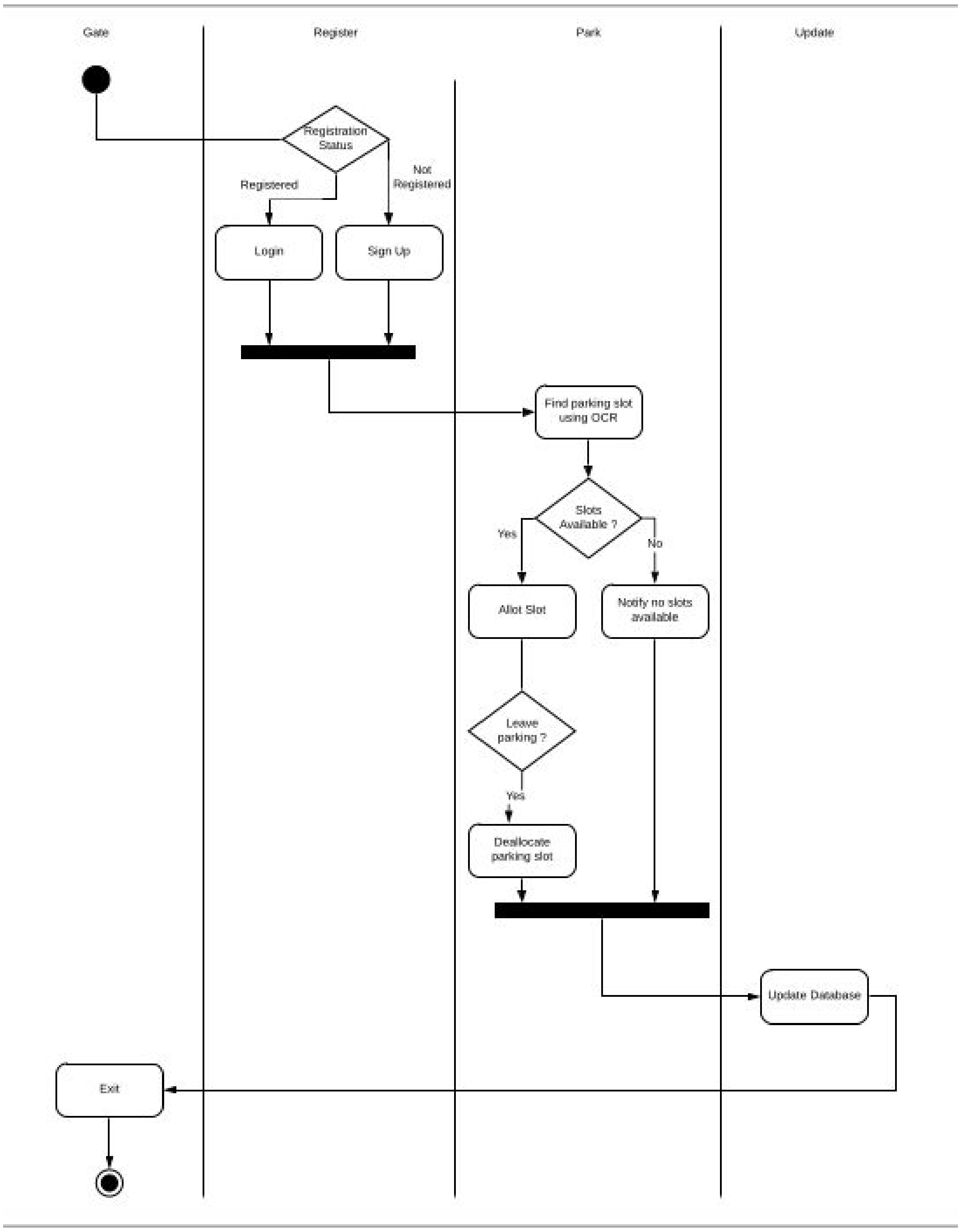
Steps :

1. The first step in the pipeline is to detect all possible parking spaces in a frame of video. Obviously we need to know which parts of the frame are parking spaces before we can detect which parking spaces are unoccupied.
2. The second step is to detect all the cars in each frame of video. This will let us track the movement of each car from frame to frame.
3. The third step is to determine which of the parking spaces are currently occupied by cars and which aren't. This requires combining the results of the first and second steps.

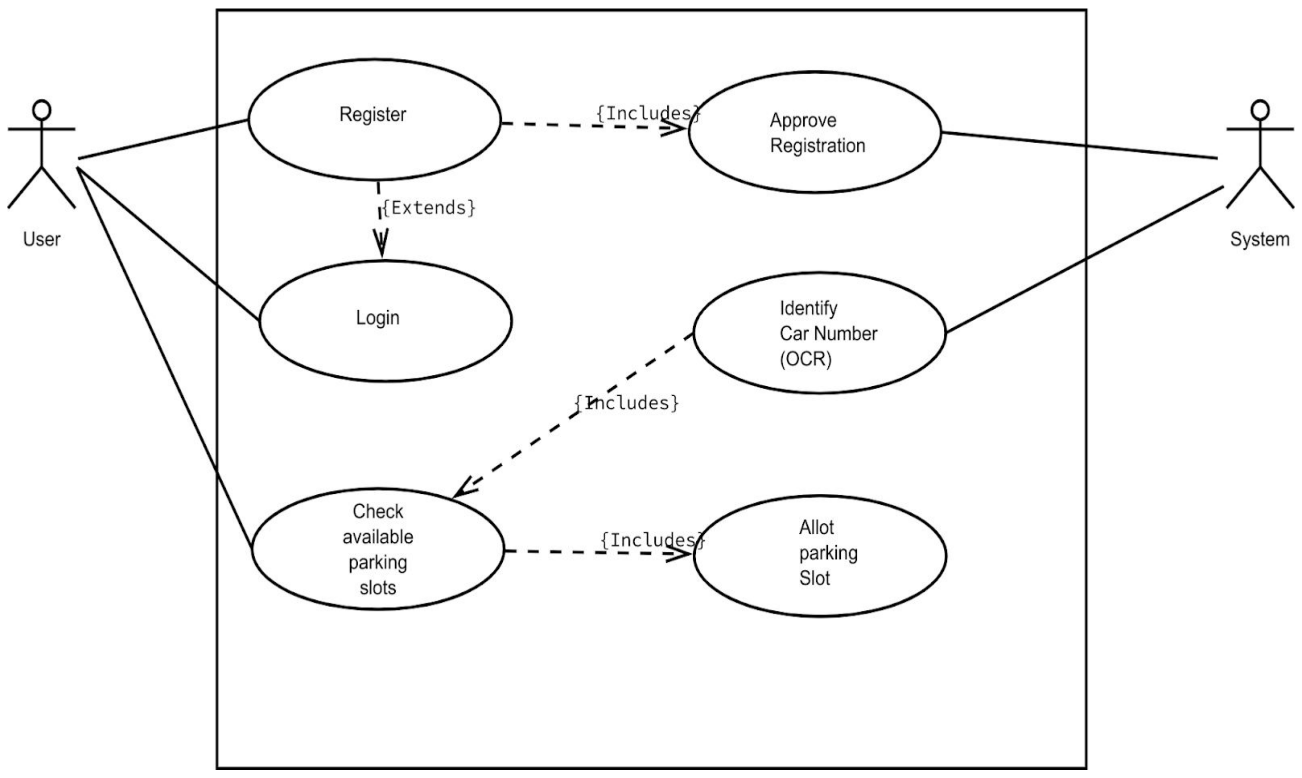
Flow of Modules



Activity Diagram



Use Case Diagram



Description Of Use Case

1. **Register:**
    a. Users have to register themselves by providing an email address and password.
    b. Users' details will be stored on the Firestore database.
    c. Users need to register themselves, in order to book parking spaces.
    d. This is required so that a parking slot can be allotted to the corresponding user with its slot number and email address.

2. **Login:**
    a. Users need to login in order to book parking slots.
    b. As soons as a user logs in, a log is generated at the backend i.e Firestore.
    c. User authentication is done by firebase itself.

3. **Check available parking slots:**
    a. The problem is that the bounding boxes of the cars in our frame partially overlap.
    b. So if we assume that each of those bounding boxes represents a parking space, it’s possible that the box can be partially occupied by a car even when the space is empty.
    c. We need a way to measure how much two objects overlap so we can check for “mostly empty” boxes.
    d. The measure we will use is called Intersection Over Union or IoU.
    e. IoU is calculated by finding the amount of pixels where two objects overlap and dividing it by the amount of pixels covered by both objects.
    f. **Formula :** Intersection over union = Intersection of boxes / Union of boxes

4. **Identify car number(OCR):**
    a. Number plate recognition is a form of automatic vehicle identification. A number plate is the unique identification of a vehicle. It is an image processing technology used to identify vehicles by their own number plates. Real time number plate recognition plays an important role in maintaining law enforcement and maintaining traffic rules.
    b. It has wide application areas such as toll plaza, parking area, highly security areas, boarder’s areas etc. Number plate recognition is designed to identify the number plate and then recognize the vehicle number plate from a moving vehicle automatically.

5. **Allot parking slot:**
    a. After user has logged in, a camera at the entrance of the parking slot scans the number plate, a database entry is made with a parking slot number corresponding to the user.
    b. While exiting the parking slot, a secondary camera scans the number plate, and a database search is performed for the corresponding number plate, if found, parking slot is deallocated and the database is updated accordingly.

Modules

Module 1 :Authentication

1. Firebase is an open source mobile SDK built by Google for developing cross platform applications using a single code base.
2. Flutter provides various plugins and APIs for functionalities like login, authentication, OCR etc.
3. Flutter provides plugins like firebase_auth plugin that uses firebase authentication API.
4. Google services plugin is required to use firebase authentication.
5. For this app level and project level gradle files need to be updated.

Google Services plugin(app.gradle):

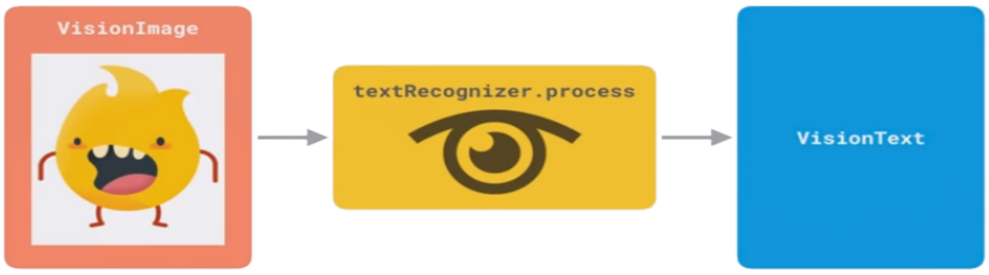
- apply plugin: 'com.google.gms.google-services'

Module 2 : Login and Register

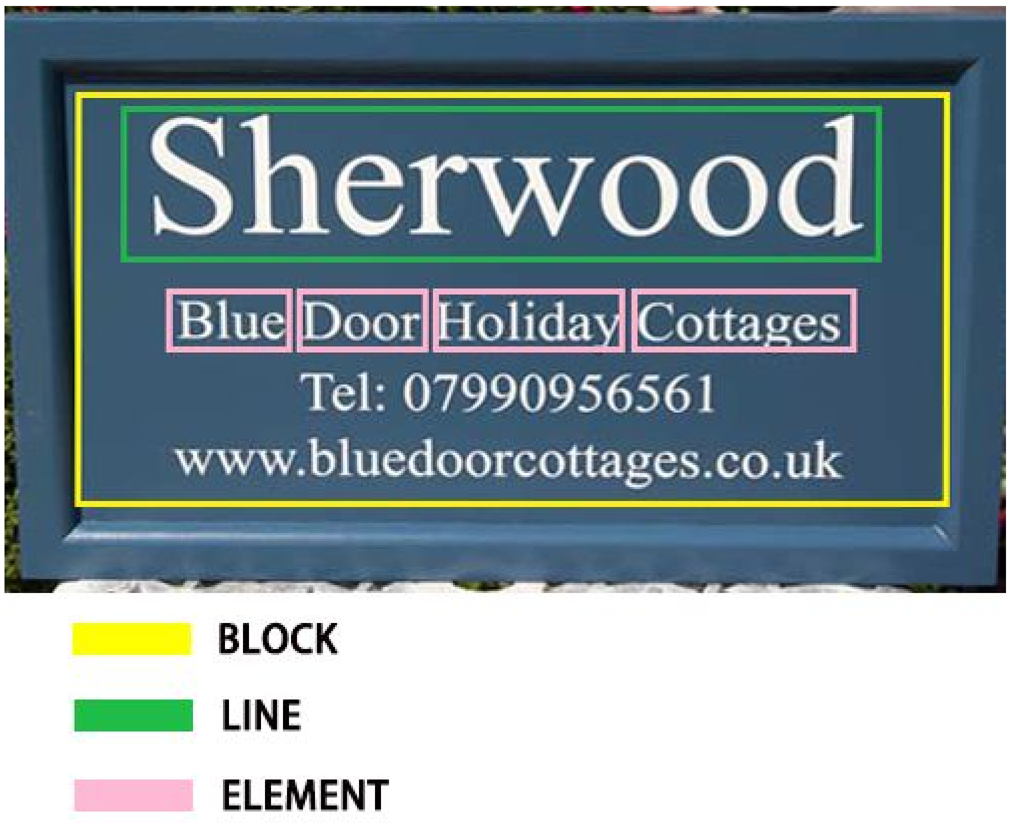
1. As mentioned earlier flutter uses firebase_auth plugin for this functionality.
2. AuthService class is used for achieving this functionality.
3. Users need to provide a registered email address and password which will be then authenticated by firebase.
4. In case of a register, users need to enter a valid Gmail ID and a password of minimum 6 characters in length.

Module 3 : Firebase OCR

1. Firebase ML kit provides both cloud based and on-device APIs
2. ML kit text recognizer works the following way :



3. Text Recogniser segments text into blocks, lines, and elements.



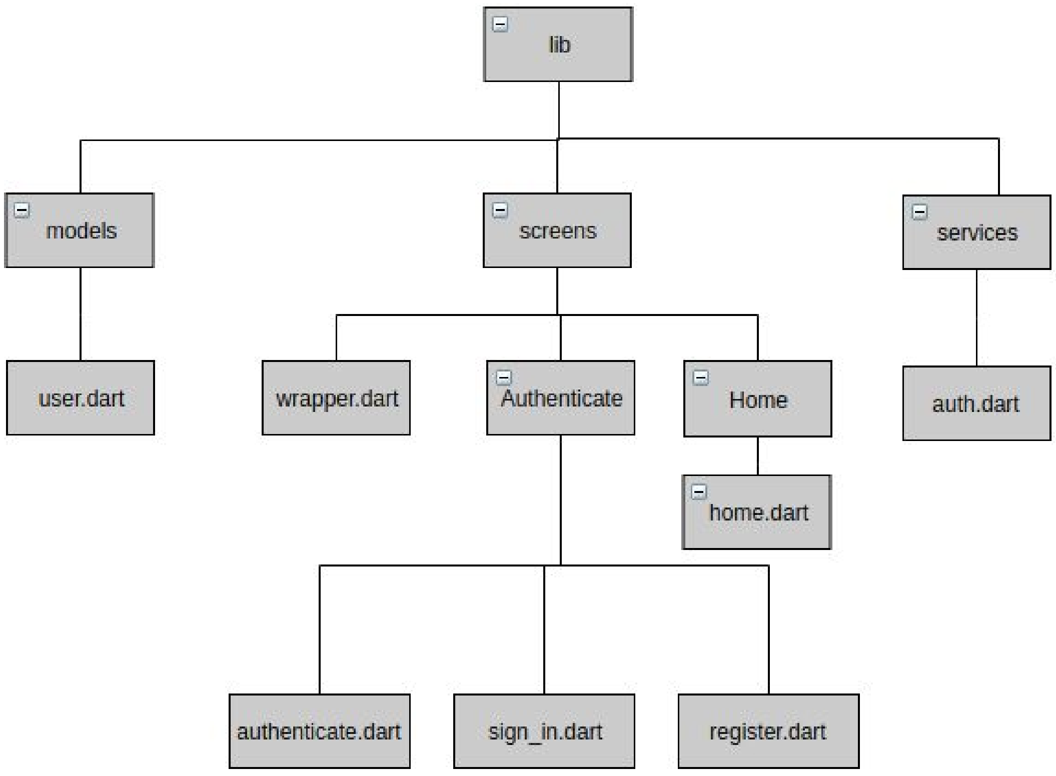
a. **Element**: Set of characters or word

b. **Line**: Sequence of words in a single line

c. **Block**: Set of lines/a paragraph

Chapter 3 : Project Implementation

Project Hierarchy



main.dart

```
import 'package:flutter/material.dart';
import 'package:parking/screens/wrapper.dart';
import 'package:provider/provider.dart';
import 'package:parking/services/auth.dart';
import 'package:parking/models/user.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return StreamProvider<User>.value(
      value: AuthService().user,
      child: MaterialApp(
        home: Wrapper(),
      ),
    );
  }
}
```

Module 1 : Authentication

wrapper.dart

```
import 'package:flutter/material.dart';
import 'package:parking/screens/authenticate/authenticate.dart';
import 'package:parking/screens/home/home.dart';
import 'package:parking/models/user.dart';
import 'package:provider/provider.dart';

class Wrapper extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    final user = Provider.of<User>(context);
    print(user);

    // return either the Home or Authenticate widget
    if (user == null) {
      return Authenticate();
    } else {
      return Home();
    }
  }
}
```

home.dart

```
import 'package:flutter/material.dart';
import 'package:parking/services/auth.dart';
```

```
class Home extends StatelessWidget {
  final AuthService _auth = AuthService();

  @override
  Widget build(BuildContext context) {
    return Container(
      child: Scaffold(
        backgroundColor: Colors.white,
        appBar: AppBar(
          title: Text('Home Page'),
          backgroundColor: Colors.blue[400],
          elevation: 0.0,
          actions: <Widget>[
            FlatButton.icon(
              icon: Icon(Icons.person),
              label: Text('logout'),
              onPressed: () async {
                await _auth.signOut();
              },
            ),
          ],
        ),
      ),
    );
  }
}
```

authenticate.dart

```
import 'package:flutter/material.dart';
import 'package:parking/screens/authenticate/sign_in.dart';
import 'package:parking/screens/authenticate/register.dart';

class Authenticate extends StatefulWidget {
  @override
  _AuthenticateState createState() => _AuthenticateState();
}
```

```
class _AuthenticateState extends State<Authenticate> {
  bool showSignIn = true;
  void toggleView() {
    //print(showSignIn.toString());
    setState(() => showSignIn = !showSignIn);
  }

  @override
  Widget build(BuildContext context) {
    if (showSignIn) {
      return SignIn(toggleView: toggleView);
    } else {
      return Register(toggleView: toggleView);
    }
  }
}
```

Module 2 : Login and Register

user.dart

```
class User {
  final String uid;

  User({this.uid});
}
```

sign_in.dart

```
import 'package:parking/services/auth.dart';
import 'package:flutter/material.dart';

class SignIn extends StatefulWidget {
  final Function toggleView;
  SignIn({this.toggleView});
  @override
  _SignInState createState() => _SignInState();
}

class _SignInState extends State<SignIn> {
  final AuthService _auth = AuthService();
  final _formKey = GlobalKey<FormState>();
  String error = '';

  // text field state
  String email = '';
  String password = '';

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      appBar: AppBar(
        backgroundColor: Colors.blue[400],
        elevation: 0.0,
        title: Text('Sign in'),
        actions: <Widget>[
          FlatButton.icon(
            icon: Icon(Icons.person),
            label: Text('Register'),
            onPressed: () => widget.toggleView(),
          ),
        ],
      ),
      body: Container(
        padding: EdgeInsets.symmetric(vertical: 20.0, horizontal: 50.0),
        child: Form(
          key: _formKey,
          child: Column(
            children: <Widget>[
              SizedBox(height: 20.0),
          TextFormField(
            validator: (val) => val.isEmpty ? 'Enter an email' : null,
            onChanged: (val) {
              setState(() => email = val);
            },
          ),
          SizedBox(height: 20.0),
          TextFormField(
            obscureText: true,
            validator: (val) =>
                val.length < 6 ? 'Enter a password 6+ chars long' : null,
            onChanged: (val) {
              setState(() => password = val);
            },
          ),
          SizedBox(height: 20.0),
          RaisedButton(
              color: Colors.blue[400],
              child: Text(
                'Sign In',
                style: TextStyle(color: Colors.white),
              ),
              onPressed: () async {
                if (_formKey.currentState.validate()) {
                  dynamic result = await
_auth.signInWithEmailAndPassword(
                      email, password);
                  if (result == null) {
                    setState(() {
                      error = 'Could not sign in with those credentials';
                    });
                  }
                }
              }),
          SizedBox(height: 12.0),
          Text(
              error,
              style: TextStyle(color: Colors.red, fontSize: 14.0),
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

register.dart

```
import 'package:flutter/material.dart';
import 'package:parking/services/auth.dart';

class Register extends StatefulWidget {
 final Function toggleView;
 Register({this.toggleView});

 @override
 _RegisterState createState() => _RegisterState();
}

class _RegisterState extends State<Register> {
 final AuthService _auth = AuthService();
 final _formKey = GlobalKey<FormState>();
 String error = '';

 // text field state
 String email = '';
 String password = '';

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   backgroundColor: Colors.white,
   appBar: AppBar(
          backgroundColor: Colors.blue[400],
          elevation: 0.0,
          title: Text('Sign up '),
          actions: <Widget>[
            FlatButton.icon(
              icon: Icon(Icons.person),
              label: Text('Sign In'),
              onPressed: () => widget.toggleView(),
            ),
          ],
        ),
        body: Container(
          padding: EdgeInsets.symmetric(vertical: 20.0, horizontal: 50.0),
          child: Form(
            key: _formKey,
            child: Column(
              children: <Widget>[
                SizedBox(height: 20.0),
                TextFormField(
                  validator: (val) => val.isEmpty ? 'Enter an email' : null,
                  onChanged: (val) {
                    setState(() => email = val);
                  },
                ),
                SizedBox(height: 20.0),
                TextFormField(
                  obscureText: true,
                  validator: (val) =>
                      val.length < 6 ? 'Enter a password 6+ chars long' : null,
                  onChanged: (val) {
                    setState(() => password = val);
                  },
                ),
                SizedBox(height: 20.0),
                RaisedButton(
                    color: Colors.blue[400],
                child: Text(
                  'Register',
                  style: TextStyle(color: Colors.white),
                ),
                onPressed: () async {
                  if (_formKey.currentState.validate()) {
                    dynamic result = await
_auth.registerWithEmailAndPassword(
                        email, password);
                    if (result == null) {
                      setState(() {
                        error = 'Please supply a valid email';
                      });
                    }
                  }
                }),
            SizedBox(height: 12.0),
            Text(
              error,
              style: TextStyle(color: Colors.red, fontSize: 14.0),
            )
          ],
        ),
      ),
    ),
  );
  }
}
```

Module 3 : Firebase OCR

```
class _MyHomePageState extends State<MyHomePage> {
  File pickedImage;
  List<List<String>> textList=[];
  List<String> lineList=[];
  bool isImageLoaded = false;
  String text = "";
  Future pickImage() async {
    var tempStore = await ImagePicker.pickImage(source:
ImageSource.gallery);
    setState(() {
      pickedImage = tempStore;
      isImageLoaded = true;
    });
  }

  Future readText() async {

    textList = [];
    FirebaseVisionImage ourImage =
FirebaseVisionImage.fromFile(pickedImage);print('2');
    TextRecognizer recognizeText =
FirebaseVision.instance.textRecognizer();
    VisionText readText = await recognizeText.processImage(ourImage);
    for (TextBlock block in readText.blocks) {
      for (TextLine line in block.lines) {
        for (TextElement word in line.elements) {
          lineList.add(word.text);
        }
        textList.add(lineList);
        lineList = [];
      }
    }

    String csv = const ListToCsvConverter().convert(textList);
    print(csv);
    setState(() {
      text=text+csv;
    });
  }
```

Chapter 4 : Testing

Design of Test Cases

1. Our project involves authentication, Optical Character Recognition(OCR), database CRUD operations.
2. These functionality involves rigorous testing to ensure each lie of code performs as intended by the user and developer.
3. Various testing methods such as Path Coverage, Unit testing etc can be used.
4. Our test cases are designed by taking every small module of code into consideration.
5. For example, there is a test case for taking image or video input.
6. Another test case is to see if OCR on image or video is applied properly.

Testing

Path Coverage Testing

1. Path coverage testing is a specific kind of testing in which each individual line of code is assessed.
2. The way that path coverage testing works is that the tester must look at each individual line of code and analyze every possible scenario related to that line of code.
3. So in our car detect code we test various parameters like input and OCR.

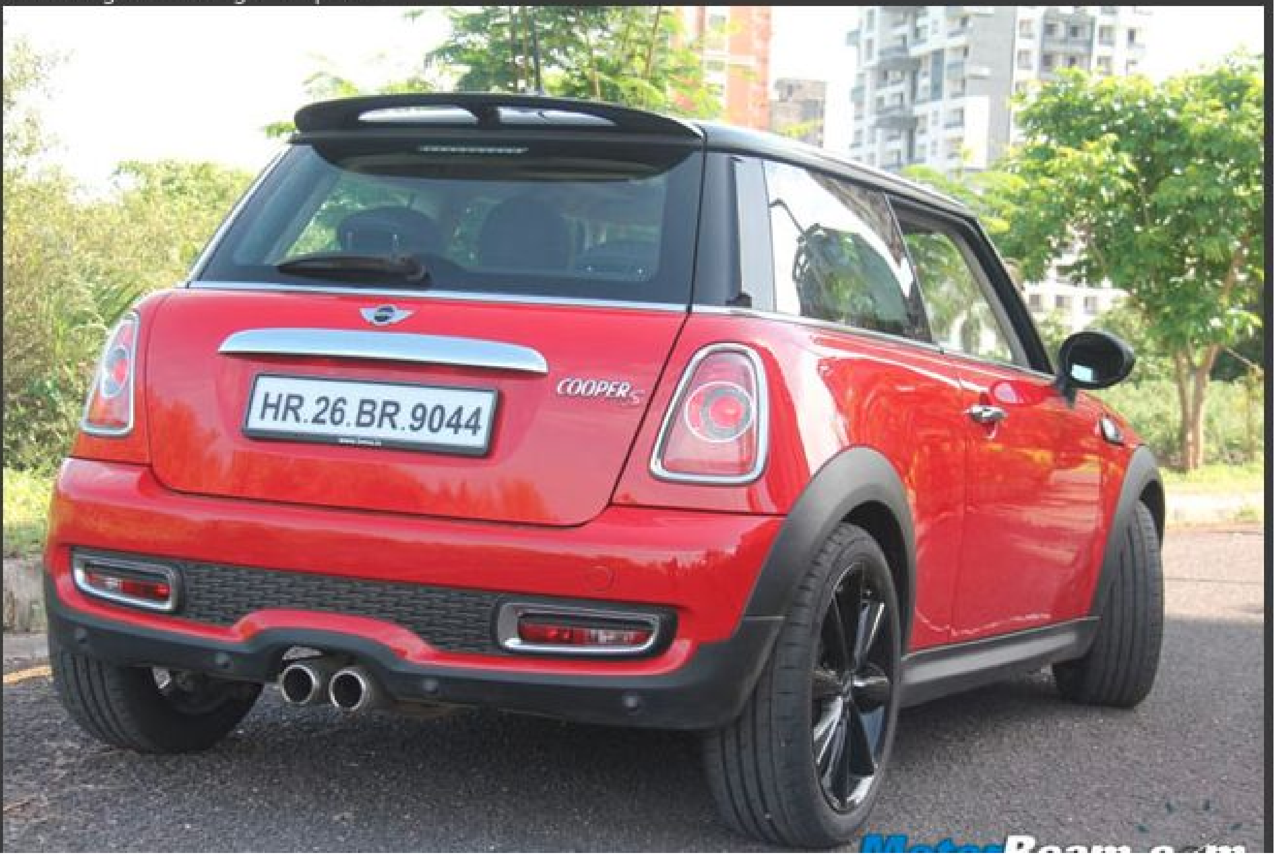
Testing of inputs

1. Here the goal of this test is to verify image and video input.
2. The following code tests if the image input is properly working and an output is generated accordingly.

```python
img = cv2.imread('/content/drive/My Drive/plate_recog/4.jpg')
cv2_imshow(img)
```

3. Output :

```
Testing of image input :
```



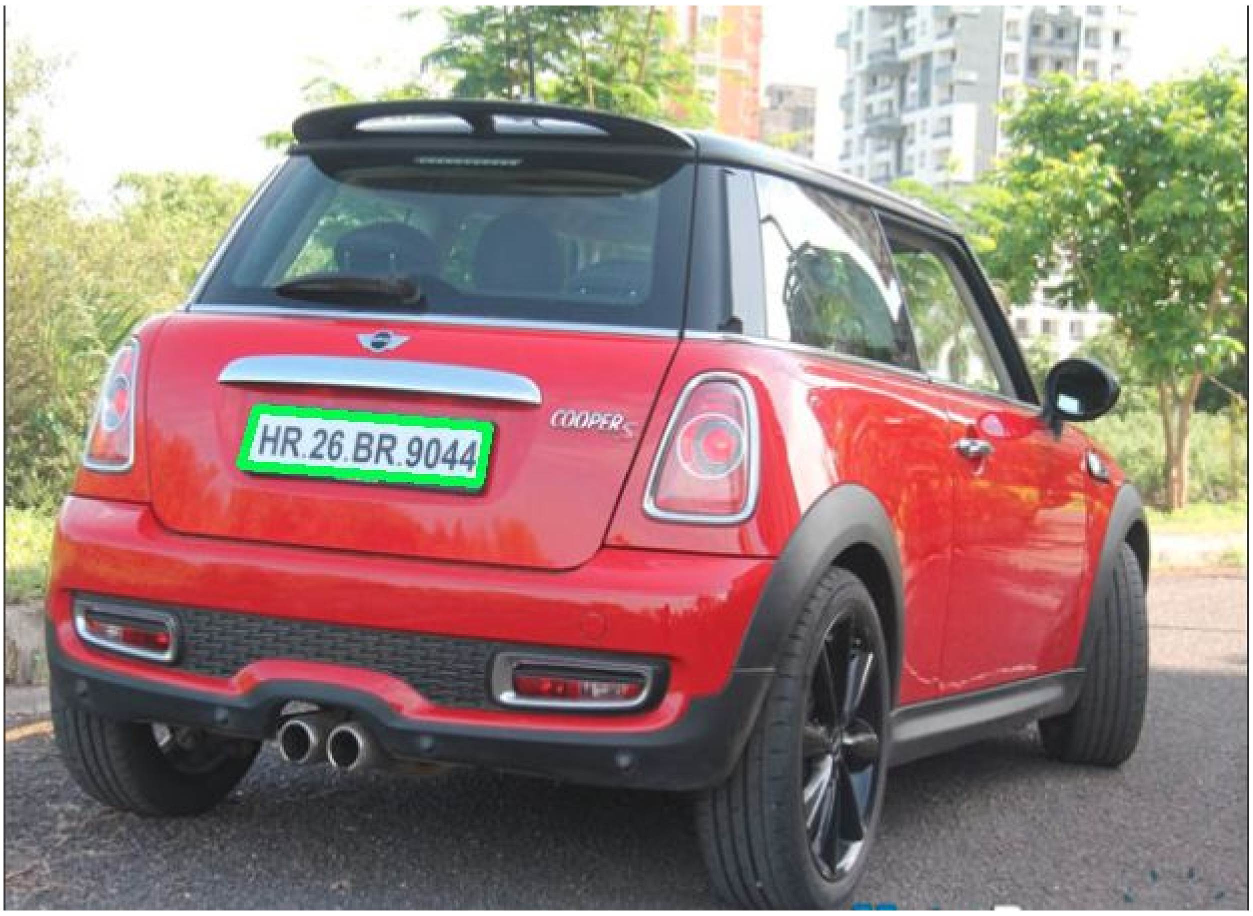
Finding Contours

1. Contours are an outline representing or bounding the shape or form of something. In this case, a number plate.

```python
cnts = cv2.findContours(edged.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
cnts = imutils.grab_contours(cnts)
cnts = sorted(cnts, key = cv2.contourArea, reverse = True)[:10]
screenCnt = None
```

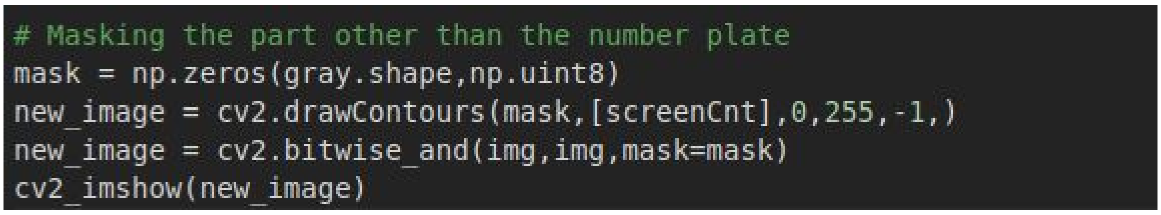
```python
if detected == 1:
  cv2.drawContours(img, [screenCnt], -1, (0, 255, 0), 3)
```

2. Output : The contour helps in finding number plate. If found, the border of the number plate is highlighted in a green bounding box.



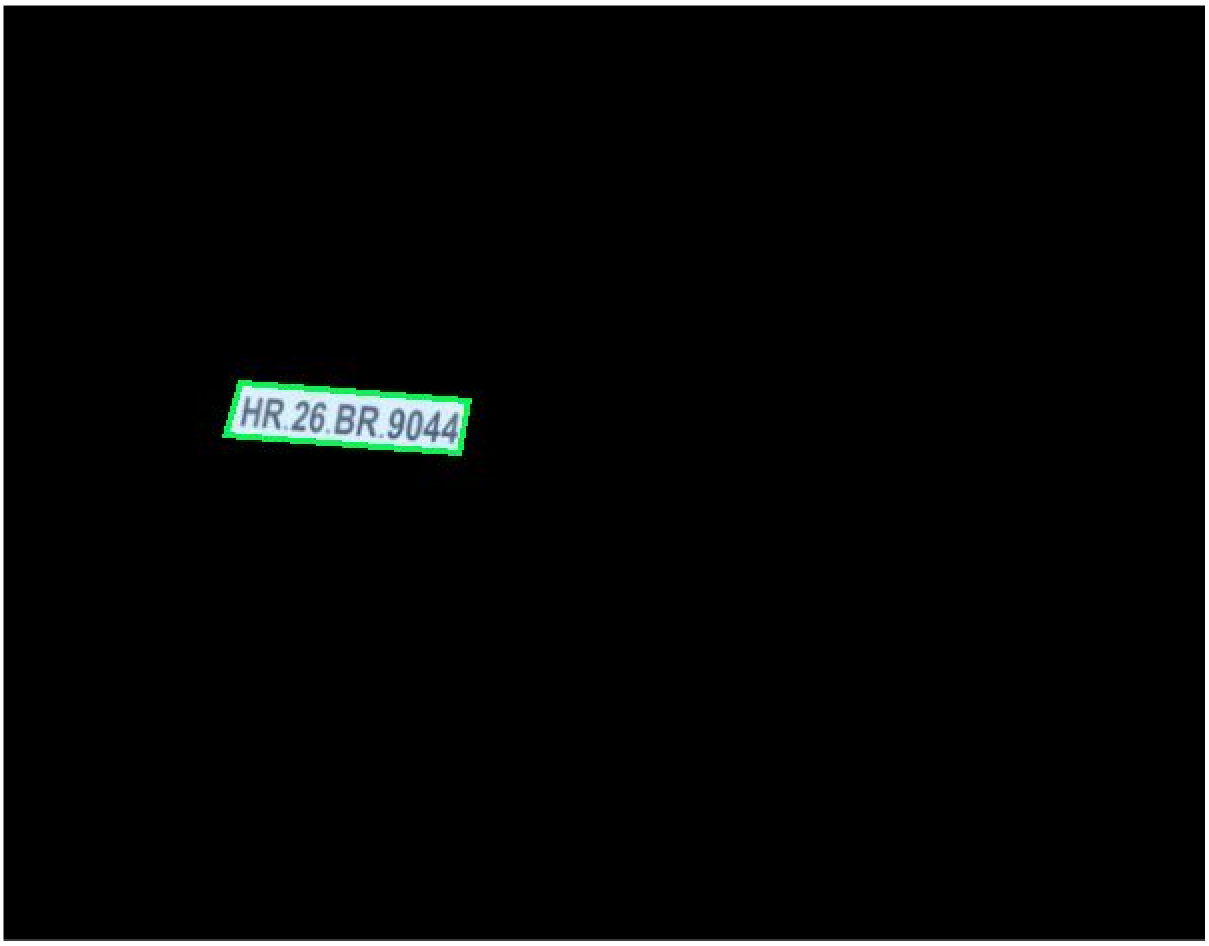
Test case to crop unwanted parts of image(i.e everything except the number plate)

1. The following code is used to crop the remaining part of image except the number plate.



```
# Masking the part other than the number plate
mask = np.zeros(gray.shape,np.uint8)
new_image = cv2.drawContours(mask,[screenCnt],0,255,-1,)
new_image = cv2.bitwise_and(img,img,mask=mask)
cv2_imshow(new_image)
```

2. Output : We obtain a marked image which shows only the number plate.



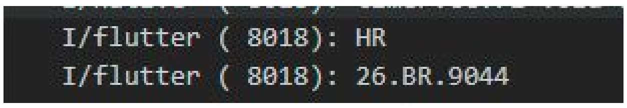
Testing Flutter App OCR

```
Future readText() async {

  textList = [];
  FirebaseVisionImage ourImage = FirebaseVisionImage.fromFile(pickedImage);print('2');
  TextRecognizer recognizeText = FirebaseVision.instance.textRecognizer();
  VisionText readText = await recognizeText.processImage(ourImage);
  for (TextBlock block in readText.blocks) {
    for (TextLine line in block.lines) {
      for (TextElement word in line.elements) {
        lineList.add(word.text);
      }
      textList.add(lineList);
      lineList = [];
    }
  }
```

```
    String csv = const ListToCsvConverter().convert(textList);
    print(csv);
    setState(() {
      text=text+csv;
    });
  }
```

Output :

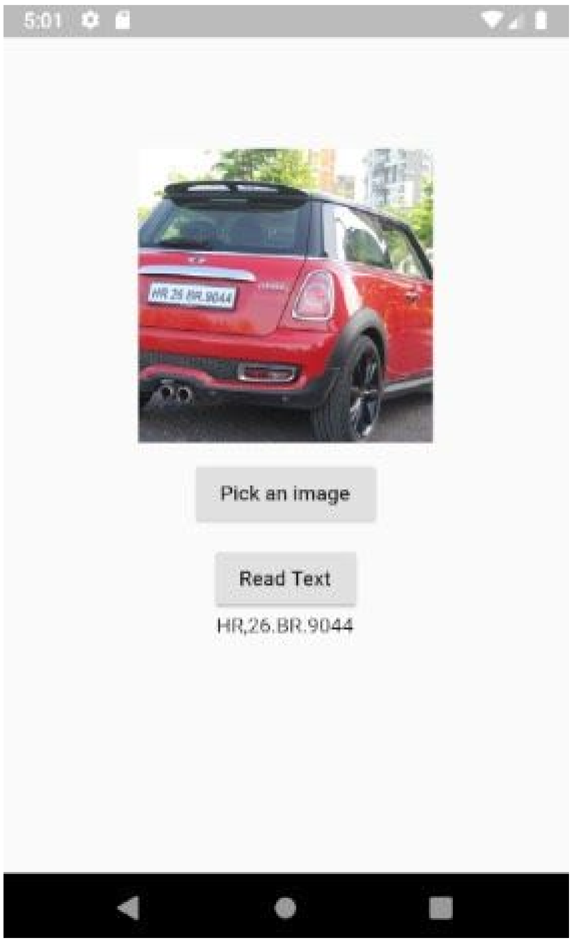


```
I/flutter ( 8018): HR
I/flutter ( 8018): 26.BR.9044
```

Chapter 5 : Results and Analysis

1. Results of Flutter OCR

a. An image input is given to an app which performs Firebase OCR and the number plate of the car is obtained.
b. The image can be given as input using a primary camera or can be supplied from local storage.
c. In this case, the image was supplied form local storage.
d. Below you can see



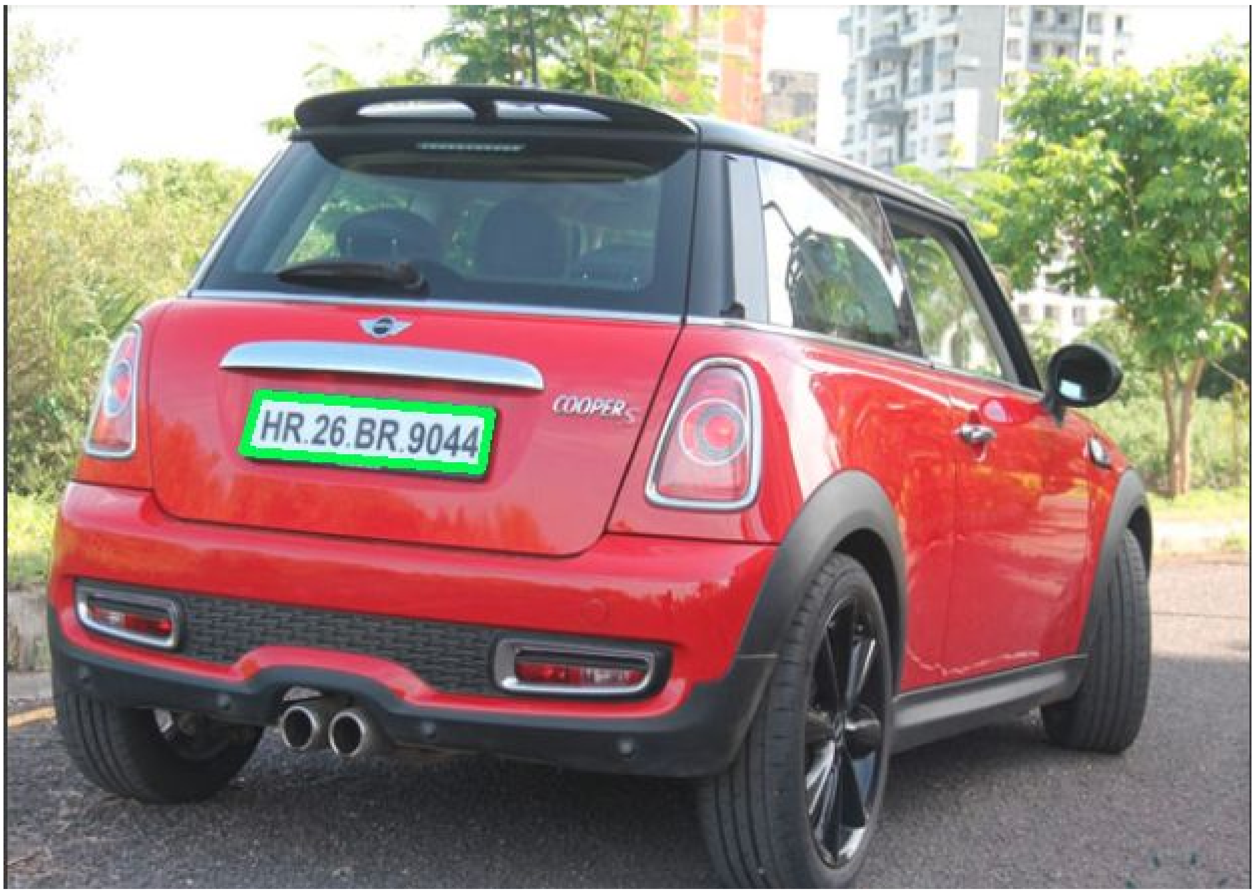
HR,26.BR.9044

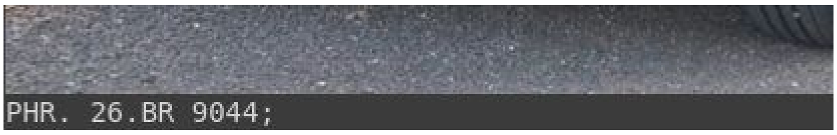
2. OCR on image

a. The following test is done to ensure OCR is correctly applied on the image and the expected output is generated.

```
text = pytesseract.image_to_string(Cropped)
#print("Detected Number is:".format(text))
print(text)
```

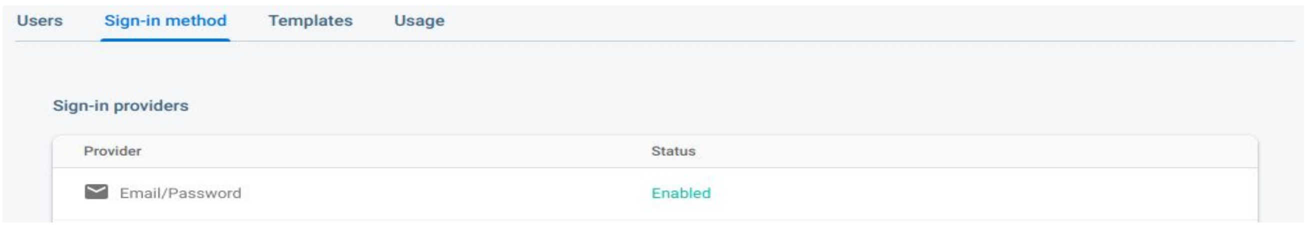
b. Output :





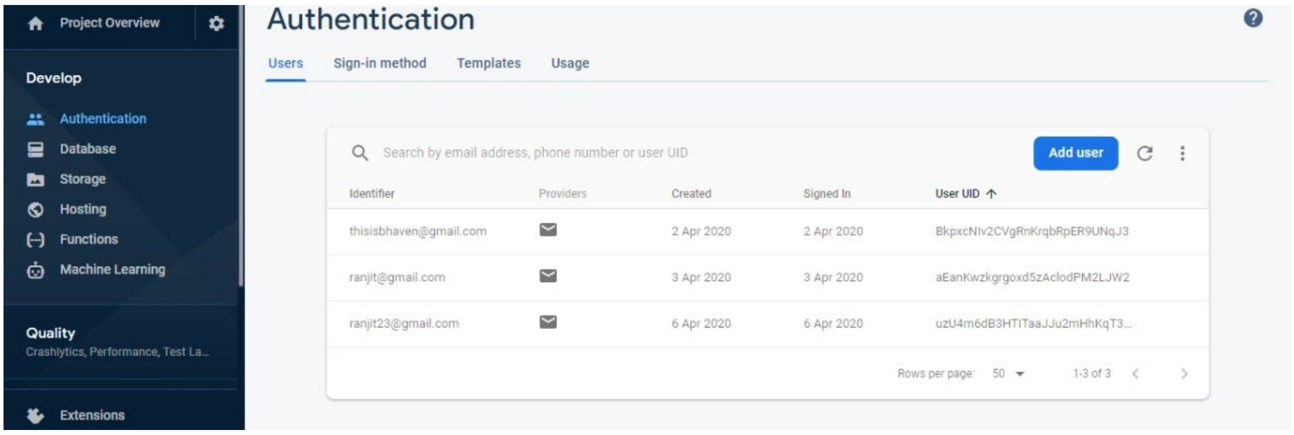
```
PHR. 26.BR 9044;
```

3. Authentication and Users



Sign-in providers

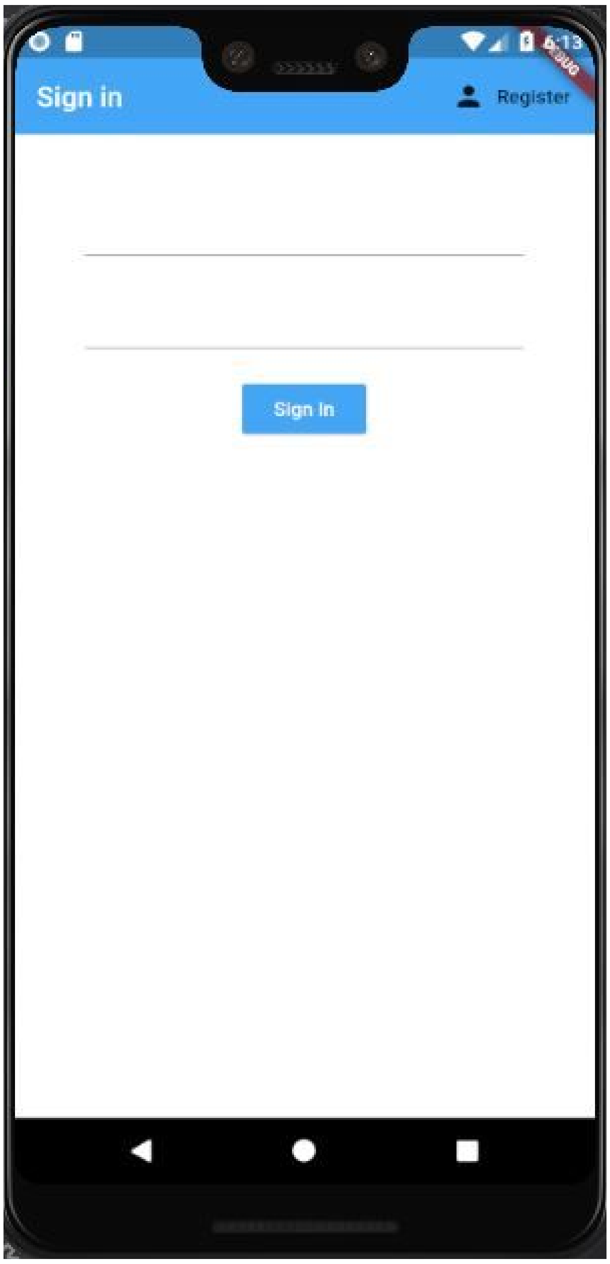
Provider	Status
Email/Password	Enabled



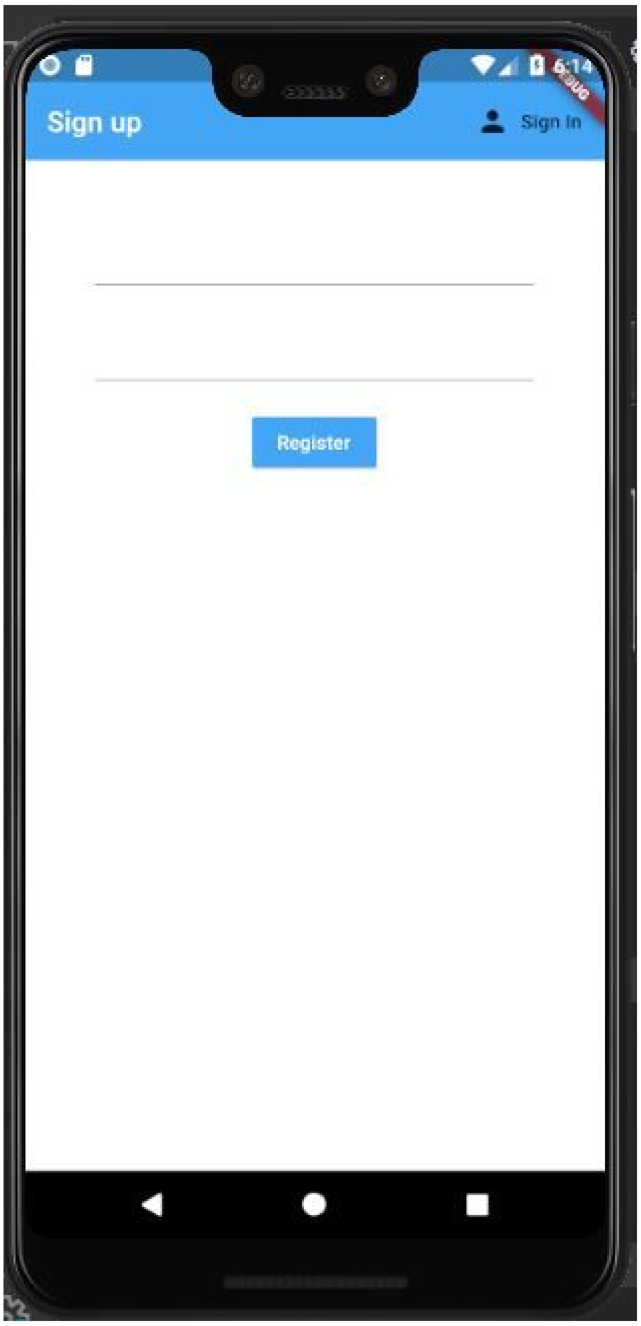
Identifier	Providers	Created	Signed In	User UID ↑
thisisbhaven@gmail.com	✉	2 Apr 2020	2 Apr 2020	BkpxcNIv2CVgRnKrqbRpER9UNqJ3
ranjit@gmail.com	✉	3 Apr 2020	3 Apr 2020	aEanKwzkgrgoxd5zAclodPM2LJW2
ranjit23@gmail.com	✉	6 Apr 2020	6 Apr 2020	uzU4m6dB3HTITaaJJu2mHhKqT3...

Rows per page: 50 1-3 of 3

4. Login



5. Register



Bibliography

1. Ilhan Aydin, Mehmet Karakose, Ebru Karakose,”***A Navigation and Reservation Based Smart Parking Platform Using Genetic Optimization for Smart Cities***”,Pg no 6.

2. I.V. Sukhinskiy, E.A.Nepovinnykh and G.I Radchenko,”***Developing a Parking Monitoring System Based on the Analysis of Images from an Outdoor Surveillance Camera”Developing a Parking Monitoring System Based on the Analysis of Images from an Outdoor Surveillance Camera****,* Pg no 7.

3. Azhar Somani, Shubham Periwal, Kesha Patel, Pranit Gaikwad, “***Cross Platform Smart Reservation Based Parking System***”, Pg no 8.