AI ENABLED CAR PARKING USING OPEN CV ARTIFICIAL INTELLIGENCE

A PROJECT REPORT

Submitted by

DEVISRI S	**420420104009**
GEETHA M	**420420104011**
JAYANTHI N	**420420104020**
KATHIJA FYROSE FATHIMA	**420420104026**

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

Team ID: NM2023TMID21776

Naanmudhalvan

SI/ PBL-NT-GP--5672-1680797853

TABLE OF CONTENTS

TABLE OF CONTENTS

1. INTRODUCTION

1 .1PROJECT OVERVIEW

Car parking is a common problem faced by drivers in busy urban areas. For example, imagine you are driving to a shopping mall during peak hours. As you approach the mall, you notice that the parking lot is full, and several other cars are circling around looking for available spots.You join the queue of cars, hoping to find an available spot soon. However, as time passes, you realize that the parking lot is overcrowded, and it's becoming increasingly difficult to find a spot. You start to feel frustrated and anxious, knowing that you might be late for your appointment or miss out on a great shopping opportunity.

AI-enabled car parking using OpenCV is a computer vision-based project that aims to automate the parking process. The project involves developing an intelligent system that can identify empty parking spaces and it gives the count of available parking spots. The system uses a camera and OpenCV (Open Source Computer Vision) library to capture live video footage of the parking .

We propose a new method to improve the efficiency of parking lots by counting how much space is left in each parking zone and displaying that information to drivers via a smartphone app. We employ a camera to photograph the parking lot and use image processing approaches to determine if any vehicles are parked in each section. Whenever a vehicle moves into or out of a particular parking zone, the status of the whole lot

changes

1.2PURPOSE

Now-a-days increasing a rate of buying personal vehicle rapidly increase. In particular four wheeler vehicle successfully doing their business, so in that scenario think about that, if there is no place for parking or limited parking without any parking management get congestion and due to that time and energy consumption more and more. To avoid these situations in the near future, a city trying to develop as the smart city , Parking slot may be played very important role. In this article , we try to demonstrate Parking slot for particularly Four wheelers , Hough transform for shape detection . Before that disc about how different methods were used in the past. Then get simulation result for circle detection using Hough transform then trying to to get similar results using OpenCV plus Python software .

Videos were recorded using a camera that was ten feet above the parking lot. In order to ameliorate the system's ability to recognise objects, video footage was collected under various environmental and temporal situations. Frames are used to segment video. also, to reduce computational complexity, a key frame is uprooted from each segment and subjected to additional processing. Key frame subtraction is used to estimate the motion of the toy auto when it enters or exits the parking lot from the parking arena.

2.LITERATURE SURVEY

PAPER 1:

1.	**Paper title**	**" Car Parking Space Detection Using OpenCV"** Rahul Tekam, Shoheb Shaikh, Leela Bitla, Pranav Rathi and Hiamnshu Chambhare. In 2023"
	Problem Definition	• The problem of finding an appropriate parking space is a challenging one, particularly in large cities. • The increase in car ownership, parking spaces have become scarce.
	Methodology/ Algorithm	YOLOV3 Algorithm, SSD Algorithm, Deep Learning (DL)
	Advantages	• It provide upgraded security, safety and privacy • Real Time Data ,Trend Insight and Optimized Parking
	Disadvantages	• It's effective at resolving parking issues. • In provides automatic billing

PAPER 2:

2.	**Paper title**	**"A Study on Smart Parking Space Allocator and Parking Management Using Opencv"**P.K.SheelaShanthakumari1, Dr.P.Selvarani2, Dr.J.Senthil Murugan2,Dr.W.T.Chembian2, M.Mithun Kumar3,M.Karthikeyan3,M.Govindaraj 2023
	Problem Definition	• At the point when the vehicles occupy a specific parking slot, the layout diverts quickly from green to red since the average colour has changed. • The image of car taken at entrance and exit. It also shows the bill generated according to the time interval of park in
	Methodology/ Algorithm	Parking management, Parking allocator, Raspberry pi, Opencv model,Cloud server
	Advantages	• To remove the stress out of parallel parking in a tight spot without theworry of accidents and make it easier • To maneuver in trafficConvenience and independence,saves time.
	Disadvantages	• The User will not know the Shortest path available to the parking • Security issues

PAPER 3:

3.	**Paper title**	**"Design And Simulation Of A Dynamic Smart Parking System With ImageSegmentation And Deep Learning**"Mohammed Fouad Taha TAHA Master's Thesis Supervisor Asst. Prof. Dr. Abdullahi Abdu IBRAHIM Istanbul, 2022
	Problem Definition	• The parking problem is a worldwide issue. The intelligent parking spaceindications, to move the vehicle to spaces in various directions of the parking lot. • The complexity of the human visual system in order for computers to detect and analyze objects in images and videos in the same way that humans can, and this is referred to as "computer vision".
	Methodology/ Algorithm	Deep Learning (DL), Convolutional Neural Network (CNN), Support Vector Machine(SVM)
	Advantages	• The models based on decision trees is that the problem can be dividedinto several sub-problems, with clear decision rules.
	Disadvantages	• Expensive Construction & installation ,System breakdown • It requires both acutators and sensors.

PAPER 4:

4.	**Paper title**	"**Smart Parking System Using Image Processing"** Shruthi B, Raghottam Kulkarni, Sanath Kumar S, Dilip Kumar S Assistant Professor, Atria Institute of Technology, Bengaluru, Karnataka, India 2020
	Problem Definition	• If there exists any then the number plate of the car is scanned using the first camera and stored in cloud and let into the parking lot by displaying the slots available. If there are no empty slots available, then the same will be displayed. • A car enters the parking lot and the parking lot is checked for empty slots
	Methodology/ Algorithm	Image Processing, Optical Character Recognition, Multi-Storey Structures, Cloud Firestore
	Advantages	• Inherent Safety and Security • Enhanced User Experience and Integrated Payments and POS
	Disadvantages	• The parking System are usually automated ,they require regular maintenance to ensue everything is working smoothly • That's means software isn't broken &everything Works properly**.**

PAPER 5:

5.	**Paper title**	**"An Edge Based Smart Parking Solution Using Camera Networks and Deep Learning"** Harshitha Bura, Nathan Lin, Naveen Kumar, Sangram Malekar, Sushma Nagaraj, Kaikai Liu""in 2018 IEEE International Conference on Cognitive Computing.
	Problem Definition	• To evolve as an increasing number of cities struggle with traffic congestionand inadequate parking availability.
	Methodology/ Algorithm	VJ Algorithm, Deep learning, edge devices, smart cities, smart parking.
	Advantages	• It experience faster, more convenient and hassle-free. • To manage and reduce parking search traffic on the streets
	Disadvantages	• It expensive Construction & Installation. A parking management system It can cost a lot of money. • It Requires Regular Maintenance.

3. IDEATION & PROPOSED SOLUTION

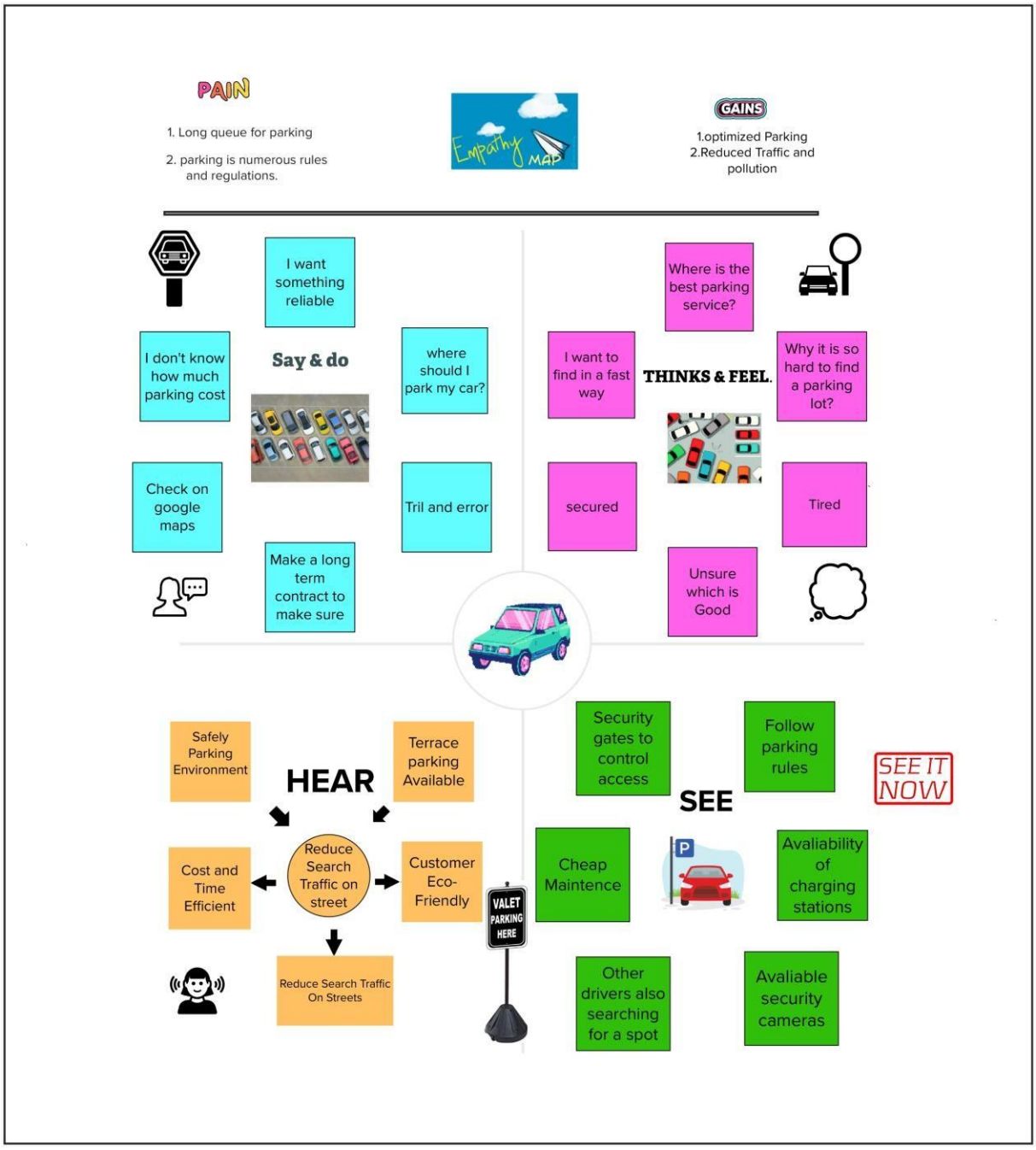
3.1 PROBLEM STATEMENT DEFINITION

Developing a computer vision-based system that can automatically detect and monitor parking spots in a parking lot using cameras. The system should be able to analyze live video feeds from the cameras and determine the availability of parking spaces.

- This system will use algorithms to detect vehicles and determine whether a parking spot is occupied or vacant.
- The primary goal of this system is to improve the efficiency of parking management, reduce congestion in parking lots, and providea better parking experience for customers.
- The system should be able to provide real-time information on the availability of parking spots, allowing drivers to find parking quickly and easily.
- The system should be able to handle different types of vehicles, including cars, trucks, and motorcycles, and should be able to detectand track multiple vehicles simultaneously.
- The goal of this project is to create an AI-enabled car parking system that is reliable, efficient, and user-friendly

TEMPLATE FOR PROBLEM STATEMENT

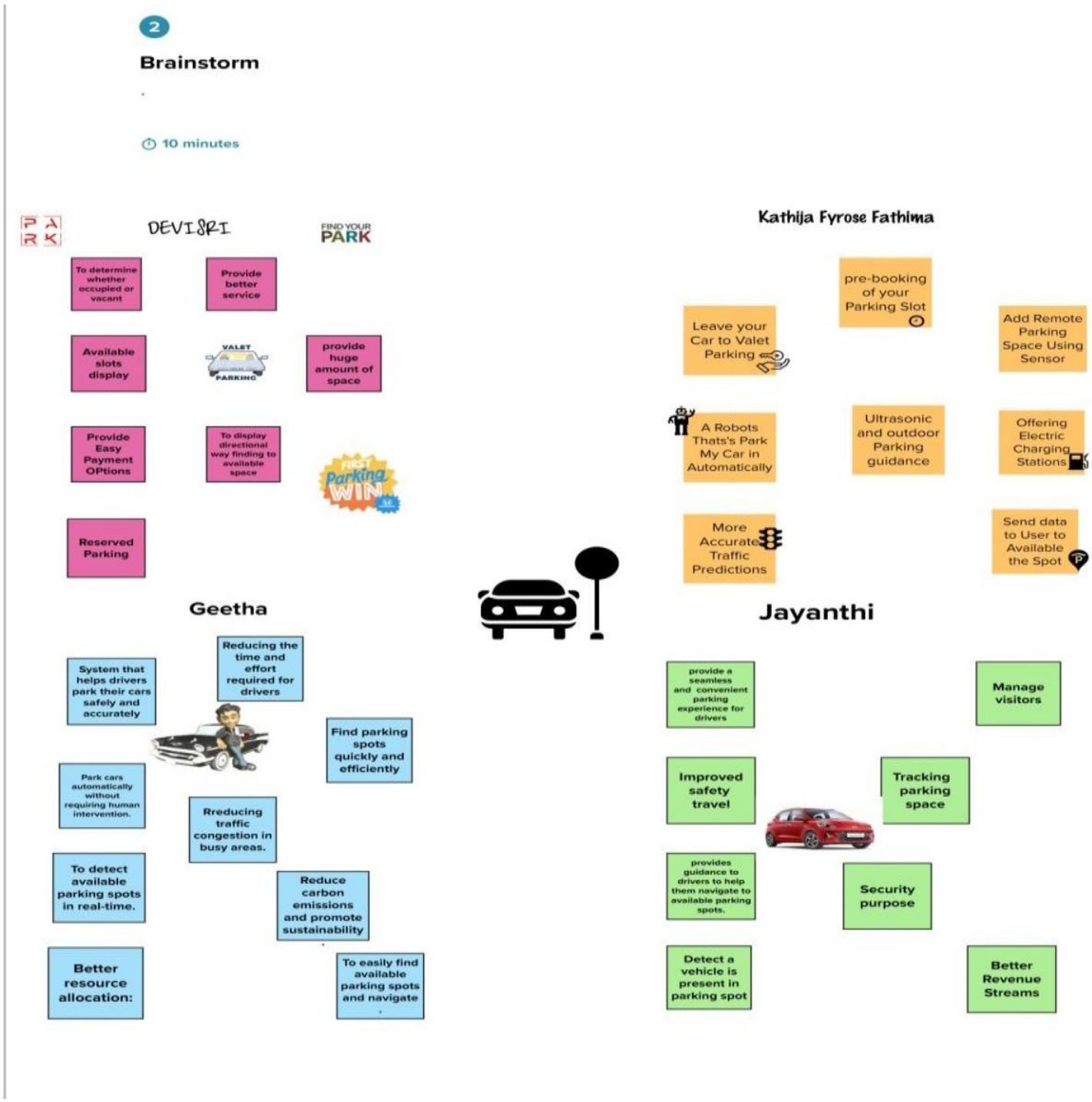
3.2 Empathy Map Canvas



3.3 IDEATION & BRAINSTROMING

Step-1: Team Gathering, Collaboration and Select the Problem Statement.

Step-2: Brainstorm Idea Listing



Step 3: Grouping Ideas

Group ideas

20 minutes

To detect available parking spots in real-time.

System helps drivers park their cars safely accurately

It offers flexible &User-friendly

providing solution in Parking Industry

Reducing traffic congestion busy area

Offering electric charging solution

To Display Way Finding available Space

Detect Parking violation as double

Step-4: Idea Prioritization

3.4 PROPOSED SOLUTION

SNO	PARAMETER	DESCRIPTION
1	Problem Statement (Problem to be solved)	• Developing a computer vision-based system that can automatically detect and monitor parking spots in a parking lot using cameras. • The system should be able to analyze live video feeds from the cameras and determine the availability of parking spaces.
2	Idea / Solution description	• The efficiency of parking lots by counting how much space is left in each parking zone and display that information to drivers via a smart phone app. • We employ a camera to photograph the parking lot and use image processing approaches to determine if any vehicles are parked in each section.
3	Novelty Uniqueness	• It will help people to find out parking space quickly. • This system will be implemented with the help of IOT(Internet of Things)

4	Social Impact / Customer Satisfaction	• Smart parking will reduce search traffic on the streets... • There are fewer traffic jams, and drivers will benefit by having less traffic on the streets
5	Business Model (Revenue Model)	• Videos were recorded using a camera that was ten feet above the parking lot. • In order to ameliorate the system's ability to recognize objects, video footage was collected under various environmental and temporal situations.
6	Scalability of the Solution	• Once the location of each parking spot is known, deep learning can be used to make a prediction on whether it is vacant or not to ensure scalability, the algorithms used for car detection and parking space detection should be optimized, and the system should be designed

4. REQUIREMENT ANALYSIS

4.1 Functional Requirements:

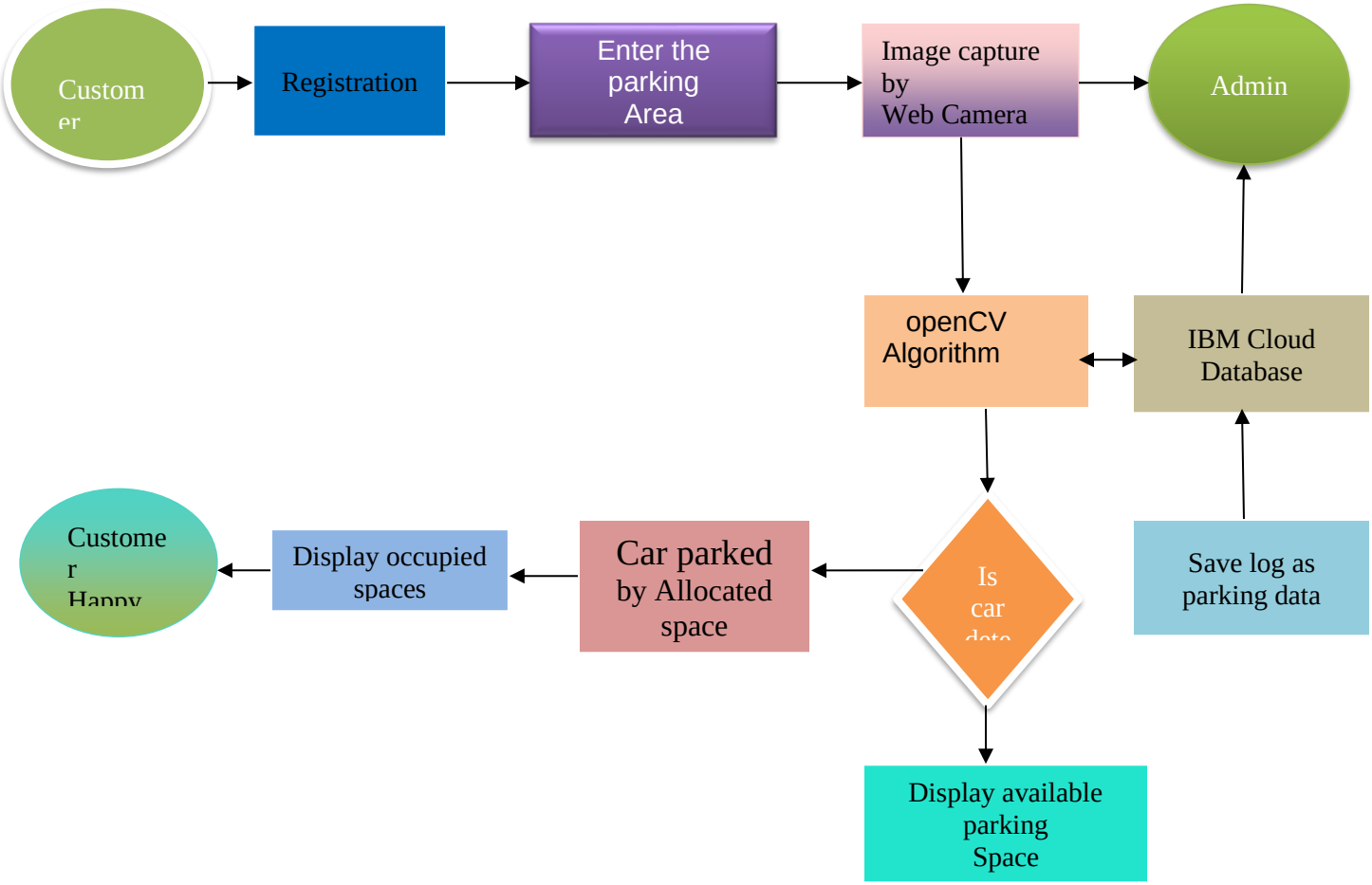
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Payment and Billing	Parking operator must be able to Issue bills to users on checkout.
FR-2	Space Management	User must be able to find a parking area from the list of areas, registered by parking admins.
FR-3	Time Management	User must be able to view the details of a selected parking area such as the name, worth per minute, number of total available lots.
FR-4	User Interface	Car parking system should be user-friendly and intuitive, allowing customers to easily locate available parking spaces.
FR-5	Security and Surveillance	View the data of all registered parking areas.
FR-6	Maintenance	Backend management system can Accept reservation of parking lots based on availability.

4.2 Non-functional Requirements:

S No.	Non-Functional Requirement	Description
NFR-2	**Usability**	The system should be user friendly and intuitive, allowing users to easily access and use the system.
NFR-2	**Security**	The system should ensure the security and privacy of the data collected, processed and stored.
NFR-3	**Reliability**	The car parking system should be reliable, with minimal downtime or errors. The system should be designed to handle high volumes of traffic without any performance issues.
NFR-4	**Performance**	The system should perform well under various conditions, including different lighting conditions, weather conditions, and traffic volumes.
NFR-5	**Availability**	The car parking system should be scalable, with the ability to handle an increasing number of users and vehicles.
NFR-6	**Scalability**	The system should be scalable to handle multiple parking lots of varying sizes and complexities.

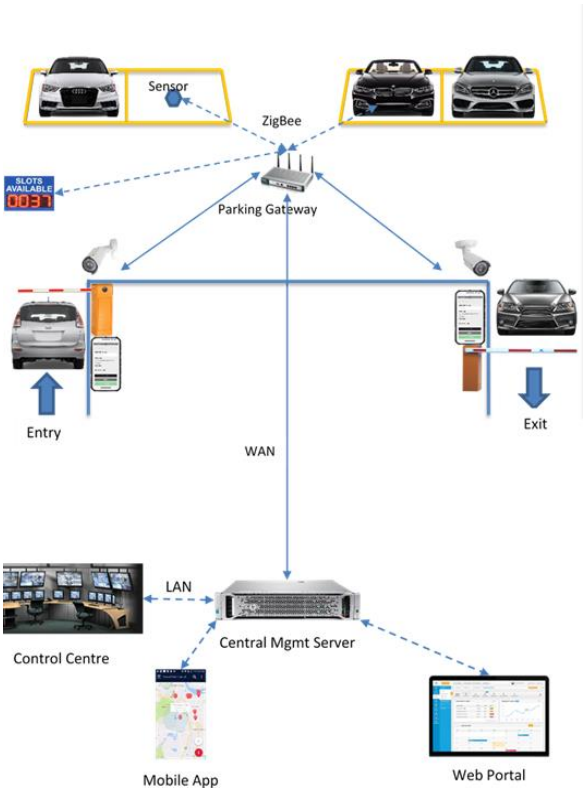
5. PROJECT DESIGN

5.1 **Data Flow Diagram**



5.2 SOLUTION & TECHNICAL ARCHITECTURE

Technical Architecture



Guidelines:

1. Follow the Direction of Traffic Flow.
2. Park within Marked Spaces.
3. Queries is Sent to Watson Assistant.
4. Watson Assistant accepts the Query.
5. Watson using finds the relevant response from cloud using Watson .
6. The Queries are stored in Database
7. Secure your vehicle

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g. Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Application Logic-1	Logic for a process in the application	Python
3.	Application Logic-2	Logic for a process in the application	IBM Watson STT service
4.	Application Logic-3	Logic for a process in the application	IBM Watson Assistant
5.	Database	Data Type, Configurations etc.	MySQL
6.	Cloud Database	Database Service on Cloud	IBM DB2, IBM Cloud etc.
7.	File Storage	File storage requirements	IBM Block Storage or Local Filesystem
8.	External API-1	Purpose of External API used in the application	IBM Weather API, etc.
9.	External API-2	Purpose of External API used in the application	Aadhar API, etc.
10.	OpenCV Algorithm	Purpose of Machine Learning Model	Object detection, video processing etc.
11.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

S.no	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	Python Flask
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	IBM Watson Assistant, IBM Cloud DB
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	Client side: Flask (python) Web Server: IBM Watson Assistant Cloud Server : IBM Cloud
4.	Availability	Detect the cars in the image using object detection algorithms such as YOLO (You Only Look Once)	IBM Cloud, Flask(python), CNN
5.	Performance	Responds to Several number of Queries at the same time.	IBM Load Balancer, IBM cloud

5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
Customer	Registration	USN-1	As a user, I can register for the application by entering email, password.	I can access my registration page	High	Jayanthi
	Identifying and tracking vehicles	USN-2	As a user, I can identify where the allocated space or empty space	I can receive the number of spaces allocated in parking Area.	Medium	Jayanthi
	Security Monitoring	USN-3	As a user, I can prevent unauthorized access and ensure the safety of my customers.	I can to monitor the parking lot in real-time to ensure the safety of their vehicles	High	Geetha
	Vehicle Recognition	USN-4	As a user, I can able to recognize the license plates of the vehicles.	I can leaving the parking area using OpenCValgorithms	High	Kathija Fyrose Fathima
Admin	payment	USN-5	As a user, I can be able to calculate the parking fee based on the duration of parking	I can maintained security purpose	High	Devisri

6. CODING & SOLUTION

6.1 Feature

Python Flask

Python Flask is used to develop AI Enabled Car Parking open CV using python. Flask is mainly used to render and Integrate the Car Parking using open cv in the browser by Providing API. By running the Python application, the suitable server domain link is obtained and run in the browser.

HTML

The HTML and CSS , JS images we stored in static folder. For this project we have to used Bootstrap which we stored HTML files.

Build PYTHON FLASK Code:

APP.PY

```
from flask import Flask, render_template
import cv2
import pickle
import cvzone
#import ibm_db
import numpy as np

app = Flask(__name__)

conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=54a2f15b-5c0f-46df-8954-
7e38e612c2bd.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=32733;SECURIT
Y=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=fkp98941;PWD=HzrFKgExo
Id39jOx")
print("connected")
@app.route('/')
def project():
    return render_template('index.html')
```

```
@app.route('/hero')
def home():
    return render_template('index.html')


@app.route('/model')
def login():
    return render_template('model.html')


@app.route('/predict_live')
def liv_pred():
    # Video feed
    cap = cv2.VideoCapture('carParkingInput.mp4')
    with open('parkingSlotPosition', 'rb') as f:
        posList = pickle.load(f)
    width, height = 107, 48
    def checkParkingSpace(imgPro):
        spaceCounter = 0
        for pos in posList:
            x, y = pos
            imgCrop = imgPro[y:y + height, x:x + width]
            # cv2.imshow(str(x * y), imgCrop)
            count = cv2.countNonZero(imgCrop)
            if count < 900:
                color = (0, 255, 0)
                thickness = 5
                spaceCounter += 1
            else:
                color = (0, 0, 255)
                thickness = 2
            cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height), color,
thickness)
            """cvzone.putTextRect(img, str(count), (x, y + height - 3),
scale=1,
                              thickness=2, offset=0, colorR=color)"""
        cvzone.putTextRect(img, f'Free: {spaceCounter}/{len(posList)}',(100,
50), scale=3,thickness=5, offset=20, colorR=(200, 0, 0))
    while True:
        cap = cv2.VideoCapture('carParkingInput.mp4')
        img = cv2.imread('carParkImg.png')
        if cap.get(cv2.CAP_PROP_POS_FRAMES) ==
cap.get(cv2.CAP_PROP_FRAME_COUNT):
            cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
        success, img = cap.read()
        imgGray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        imgBlur = cv2.GaussianBlur(imgGray, (3, 3), 1)
        imgThreshold = cv2.adaptiveThreshold(imgBlur, 255,
cv2.ADAPTIVE_THRESH_GAUSSIAN_C,cv2.THRESH_BINARY_INV, 25, 16)
```

```
        imgMedian = cv2.medianBlur(imgThreshold, 5)
        kernel = np.ones((3, 3), np.uint8)
        imgDilate = cv2.dilate(imgMedian, kernel, iterations=1)
        checkParkingSpace(imgDilate)
        cv2.imshow("Image", img)
        # cv2.imshow("ImageBlur", imgBlur)
        # cv2.imshow("ImageThres", imgMedian)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break


if __name__ == '__main__':
    app.run(debug=True)
```

Model.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta content="width=device-width, initial-scale=1.0" name="viewport">

  <title>SHPVIBE</title>
  <meta content="" name="description">
  <meta content="" name="keywords">

  <!-- Google Fonts -->
  <link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,
600i,700,700i|Raleway:300,300i,400,400i,500,500i,600,600i,700,700i|Poppins:300
,300i,400,400i,500,500i,600,600i,700,700i" rel="stylesheet">

  <!-- Vendor CSS Files -->
  <link href="static/assets/vendor/aos/aos.css" rel="stylesheet">
  <link href="static/assets/vendor/bootstrap/css/bootstrap.min.css"
rel="stylesheet">
  <link href="static/assets/vendor/bootstrap-icons/bootstrap-icons.css"
rel="stylesheet">
  <link href="static/assets/vendor/boxicons/css/boxicons.min.css"
rel="stylesheet">
  <link href="static/assets/vendor/glightbox/css/glightbox.min.css"
rel="stylesheet">
```

```
  <link href="static/assets/vendor/swiper/swiper-bundle.min.css"
rel="stylesheet">

  <!-- Template Main CSS File -->
  <link href="static/assets/css/style.css" rel="stylesheet">
<style>
  img {
  border-radius: 2%;
}
body {
background-image: url('static/assets/img/park3.jpg');
background-repeat: no-repeat;
background-attachment: fixed;
background-size: cover;
}
</style>
</head>
<body>
<header id="header" class="d-flex align-items-center">
    <div class="container d-flex align-items-center justify-content-between">

      <h1 class="logo"><a href="index.html">AI Enable car parking using
OpenCV</a></h1>
      <nav id="navbar" class="navbar">
        <ul>
          <li><a class="nav-link scrollto" href="/hero">Home</a></li>
          <li><a class="nav-link scrollto active" href="/model">Model</a></li>

        </ul>
        <i class="bi bi-list mobile-nav-toggle"></i>
      </nav><!-- .navbar -->

    </div>
  </header><!-- End Header -->
<div class="container">

    <section class="section register min-vh-100 d-flex flex-column align-
items-center justify-content-center py-4">
        <div class="container">
          <div class="row justify-content-center">
            <div class="col-lg-4 col-md-6 d-flex flex-column align-items-
center justify-content-center">
              <div class="card mb-3" style="bs-card-bg:#212529">
                <div class="card-body">
                  <div class="pt-4 pb-2">
                    <h5 class="card-title text-center pb-0 fs-4">👇Check the
parking slot👇</h5>
                    <form action="/predict_live">
```

```
                    <center>
                      <button type="submit" class="btn btn-danger">Click
here</button>
                    </center>
                    </form>
                  </div>
                  </center>
                </div>
              </div>
            </div>
          </div>
        </div>

      </section>

</div>
</body>
</html>
```

Index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta content="width=device-width, initial-scale=1.0" name="viewport">

  <title>SHPVIBE</title>
  <meta content="" name="description">
  <meta content="" name="keywords">
  <!-- Google Fonts -->
  <link
href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,
600i,700,700i|Raleway:300,300i,400,400i,500,500i,600,600i,700,700i|Poppins:300
,300i,400,400i,500,500i,600,600i,700,700i" rel="stylesheet">

  <!-- Vendor CSS Files -->
  <link href="static/assets/vendor/aos/aos.css" rel="stylesheet">
  <link href="static/assets/vendor/bootstrap/css/bootstrap.min.css"
rel="stylesheet">
  <link href="static/assets/vendor/bootstrap-icons/bootstrap-icons.css"
rel="stylesheet">
```

```
  <link href="static/assets/vendor/boxicons/css/boxicons.min.css"
rel="stylesheet">
  <link href="static/assets/vendor/glightbox/css/glightbox.min.css"
rel="stylesheet">
  <link href="static/assets/vendor/swiper/swiper-bundle.min.css"
rel="stylesheet">

  <!-- Template Main CSS File -->
  <link href="static/assets/css/style.css" rel="stylesheet">
<style>
  img {
  border-radius: 2%;
}
</style>
</head>
<body>
  <!-- ======= Header ======= -->
  <header id="header" class="d-flex align-items-center">
    <div class="container d-flex align-items-center justify-content-between">

      <h1 class="logo"><a href="index.html">SHPVIBE</a></h1>
      <nav id="navbar" class="navbar">
        <ul>
          <li><a class="nav-link scrollto active" href="#hero">Home</a></li>
          <li><a class="nav-link scrollto" href="#about">About Us</a></li>
          <li><a class="nav-link scrollto" href="/model">Model</a></li>
          <li><a class="nav-link scrollto" href="#contact">Contact</a></li>
        </ul>
        <i class="bi bi-list mobile-nav-toggle"></i>
      </nav><!-- .navbar -->

    </div>
  </header><!-- End Header -->

  <!-- ======= Hero Section ======= -->
  <section id="hero" class="d-flex align-items-center">
    <div class="container position-relative" data-aos="fade-up" data-aos-
delay="500">
      <h1>Revolutionizing Parking</h1>
      <h2>AI Enable car parking using OpenCV</h2>
      <a href="#about" class="btn-get-started scrollto">Get Started</a>
    </div>
  </section><!-- End Hero -->

  <main id="main">

    <!-- ======= About Section ======= -->
    <section id="about" class="about">
```

```
      <div class="section-title">
          <span>About</span>
          <h2>About</h2>
      </div>
      <div class="container">

        <div class="row">
          <div class="col-lg-6 order-1 order-lg-2" data-aos="fade-left">
            <img src="static/assets/img/park1.jpg" class="img-fluid" alt="">
          </div>
          <div class="col-lg-6 pt-4 pt-lg-0 order-2 order-lg-1 content" data-
aos="fade-right">
            <h3>AI Enable car parking using OpenCV</h3>
            <p class="fst-italic">
              The AI-Enabled Car Parking System Utilizing OpenCV Technology is
a cutting-edge project that aims to revolutionize the way parking lots
operate. This system uses OpenCV, a popular computer vision library, to enable
vehicles to park autonomously.
            </p>
            <ul>
              <li><i class="bi bi-check-circle"></i> The OpenCV library
analyzes the footage and identifies the available parking spaces in the
lot.</li>
              <li><i class="bi bi-check-circle"></i> The system is designed to
be highly accurate, and it can detect small and large vehicles, even in low-
light conditions.</li>
              </ul>
          </div>
        </div>

      </div>
    </section><!-- End About Section -->


    <!-- ======= Contact Section ======= -->
    <section id="contact" class="contact">
      <div class="container">

        <div class="section-title">
          <span>Contact</span>
          <h2>Contact</h2>
           </div>

        <div class="row" data-aos="fade-up">
          <div class="col-lg-6">
            <div class="info-box mb-4">
              <i class="bx bx-map"></i>
              <h3>TheSmartBridge</h3>
              <p></p>
```

```
            </div>
          </div>

          <div class="col-lg-3 col-md-6">
            <div class="info-box  mb-4">
              <i class="bx bx-envelope"></i>
              <h3>Email Us</h3>
              <p>contact@example.com</p>
            </div>
          </div>

          <div class="col-lg-3 col-md-6">
            <div class="info-box  mb-4">
              <i class="bx bx-phone-call"></i>
              <h3>Call Us</h3>
              <p>+1 5XXX 554XX 5X</p>
            </div>
          </div>

        </div>

        <div class="row" data-aos="fade-up">

          <div class="col-lg-6 ">
            <iframe class="mb-4 mb-lg-0"
src="https://www.google.com/maps/embed?pb=!1m14!1m8!1m3!1d12097.433213460943!2d-
74.0062269!3d40.7101282!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x0%3A0xb89d1fe6bc499443!2sDowntown+Conference+Center!5e0!3m2!1smk!2sbg!4v1539943755621"
frameborder="0" style="border:0; width: 100%; height: 384px;"
allowfullscreen></iframe>
          </div>

          <div class="col-lg-6">
            <form action="static/forms/contact.php" method="post" role="form"
class="php-email-form">
              <div class="row">
                <div class="col-md-6 form-group">
                  <input type="text" name="name" class="form-control"
id="name" placeholder="Your Name" required>
                </div>
                <div class="col-md-6 form-group mt-3 mt-md-0">
                  <input type="email" class="form-control" name="email"
id="email" placeholder="Your Email" required>
                </div>
              </div>
              <div class="form-group mt-3">
```

```
                <input type="text" class="form-control" name="subject"
id="subject" placeholder="Subject" required>
              </div>
              <div class="form-group mt-3">
                <textarea class="form-control" name="message" rows="5"
placeholder="Message" required></textarea>
              </div>
              <div class="my-3">
                <div class="loading">Loading</div>
                <div class="error-message"></div>
                <div class="sent-message">Your message has been sent. Thank
you!</div>
              </div>
              <div class="text-center"><button type="submit">Send
Message</button></div>
            </form>
          </div>

        </div>

      </div>
    </section><!-- End Contact Section -->

  </main><!-- End #main -->

  <!-- ======= Footer ======= -->
  <footer id="footer">
    <div class="footer-top">
      <div class="container">
        <div class="row">

          <div class="col-lg-4 col-md-6">
            <div class="footer-info">
              <h3>SHPVIBE</h3>
              <p>
                AXXX Adam Street <br>
                NY 5XXXX2, USA<br><br>
                <strong>Phone:</strong> +1 5XXX 55XXX 55<br>
                <strong>Email:</strong> info@example.com<br>
              </p>
              <div class="social-links mt-3">
                <a href="#" class="twitter"><i class="bx bxl-twitter"></i></a>
                <a href="#" class="facebook"><i class="bx bxl-
facebook"></i></a>
                <a href="#" class="instagram"><i class="bx bxl-
instagram"></i></a>
                <a href="#" class="google-plus"><i class="bx bxl-
skype"></i></a>
```

```
              <a href="#" class="linkedin"><i class="bx bxl-
linkedin"></i></a>
            </div>
          </div>
        </div>

        <div class="col-lg-2 col-md-6 footer-links">
          <h4>Useful Links</h4>
          <ul>
            <li><i class="bx bx-chevron-right"></i> <a
href="#hero">Home</a></li>
            <li><i class="bx bx-chevron-right"></i> <a href="#about">About
us</a></li>
<!--              <li><i class="bx bx-chevron-right"></i> <a
href="#services">Demo</a></li>-->
            <li><i class="bx bx-chevron-right"></i> <a
href="#contact">Contact</a></li>
          </ul>
        </div>

        <div class="col-lg-4 col-md-6 footer-newsletter">
          <h4>Our Newsletter</h4>
              <form action="" method="post">
            <input type="email" name="email"><input type="submit"
value="Subscribe">
          </form>

        </div>

      </div>
    </div>
  </div>
</footer><!-- End Footer -->

<a href="#" class="back-to-top d-flex align-items-center justify-content-
center"><i class="bi bi-arrow-up-short"></i></a>
<div id="preloader"></div>

<!-- Vendor JS Files -->
<script src="static/assets/vendor/aos/aos.js"></script>
<script
src="static/assets/vendor/bootstrap/js/bootstrap.bundle.min.js"></script>
<script src="static/assets/vendor/glightbox/js/glightbox.min.js"></script>
<script src="static/assets/vendor/isotope-
layout/isotope.pkgd.min.js"></script>
<script src="static/assets/vendor/swiper/swiper-bundle.min.js"></script>
<script src="static/assets/vendor/php-email-form/validate.js"></script>
```

```
  <!-- Template Main JS File -->
  <script src="static/assets/js/main.js"></script>

</body>

</html>
```

SelectROI.py

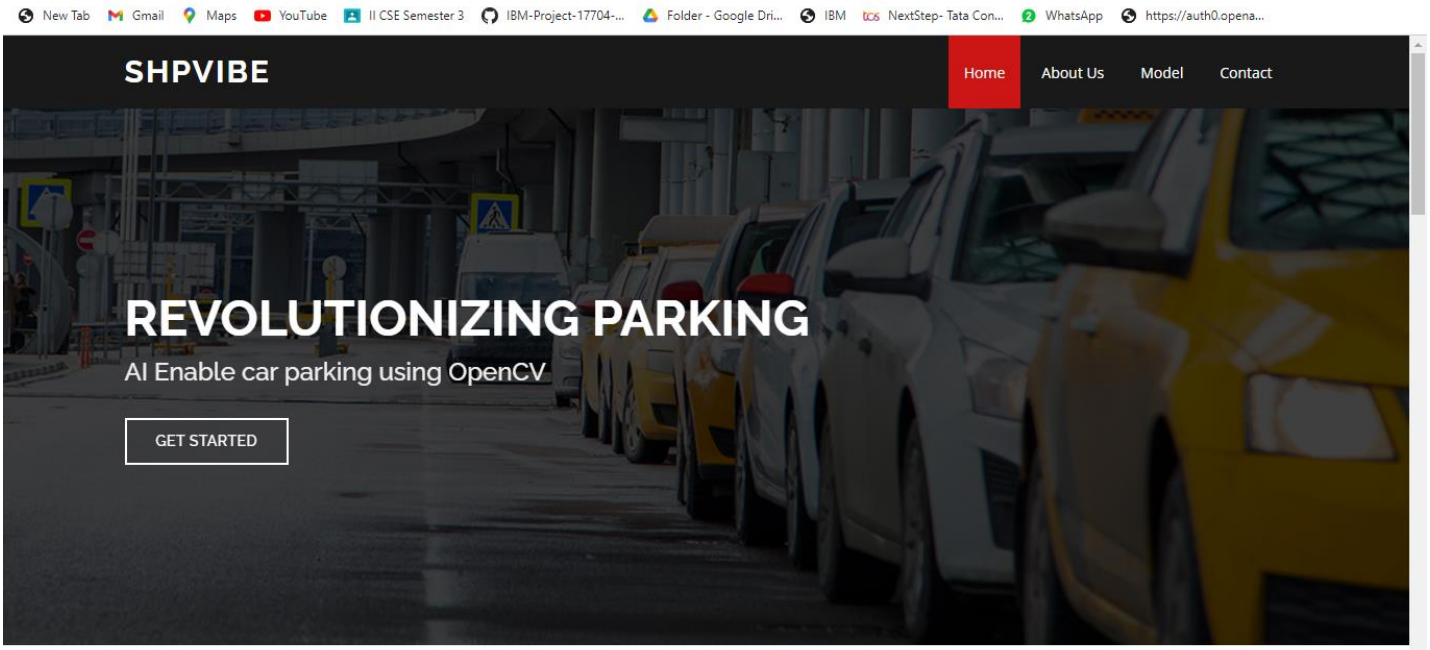
```
import cv2
import cvzone
import pickle

width, height = 107, 48

try:
    with open('ParkingSlotPosition', 'rb') as f:
        posList = pickle.load(f)
except:
    posList = []

def mouseClick(events, x, y, flags, params):
    if events == cv2.EVENT_LBUTTONDOWN:
        posList.append((x, y))
    elif events == cv2.EVENT_RBUTTONDOWN:
        for i, pos in enumerate(posList):
            x1, y1 = pos
            if x1 < x < x1 + width and y1 < y < y1 + height:
                posList.pop(i)
    with open('parkingslotposition', 'wb') as f:
        pickle.dump(posList, f)

while True:
    img = cv2.imread(r"C:\Users\PC\Desktop\PDP\flask\uploads\carParkImg.png")
    for pos in posList:
        cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height), (255, 255,
255), 2)

    cv2.imshow("Image", img)
    cv2.setMouseCallback("Image", mouseClick)
    cv2.waitKey(1)
```

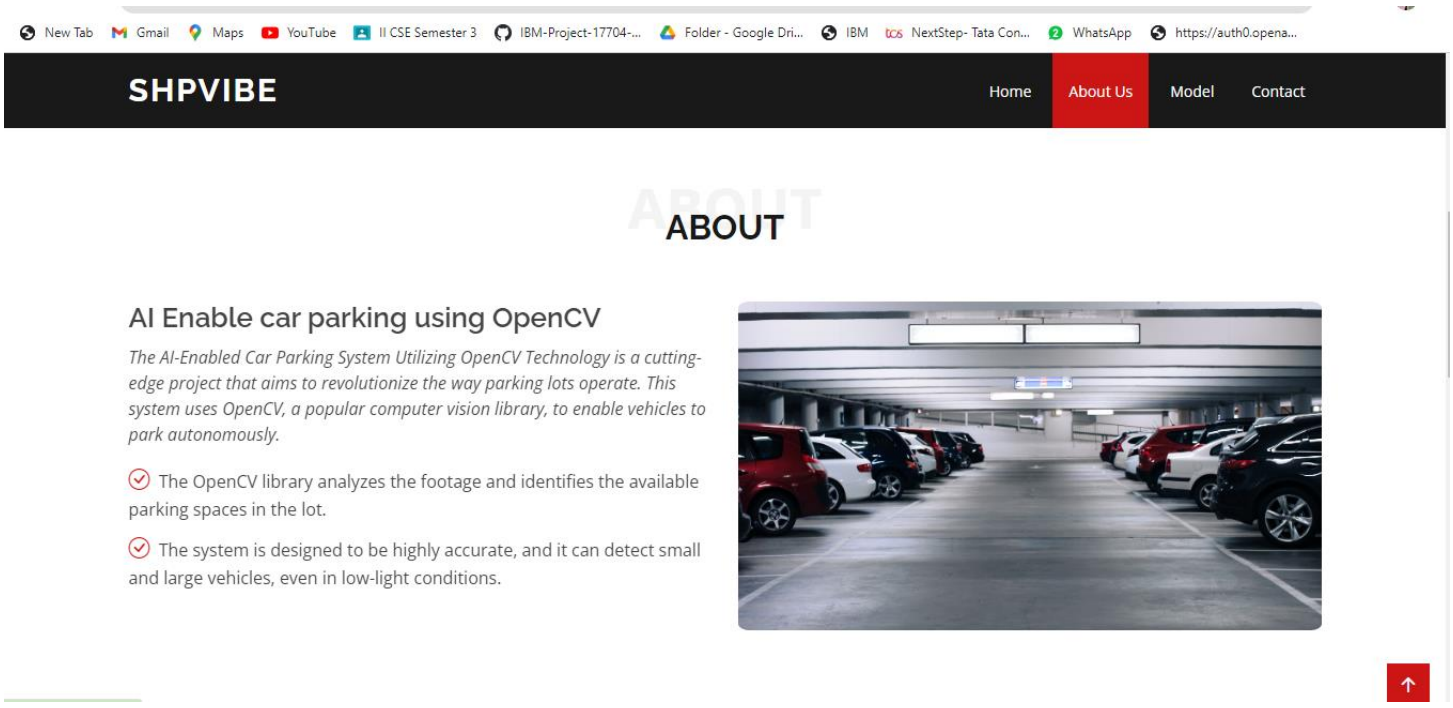
Car_park.py

```
"""# **Import packages**"""

import cv2
import pickle
import cvzone
import numpy as np

"""# **Reading Input And Loading ROI File**"""

#vedio feed
cap =
cv2.VideoCapture(r"C:\Users\PC\Desktop\PDP\flask\uploads\carParkingInput.mp4")
#Loading the ROI from parkingSlotPosition file
with open(r"C:\Users\PC\Desktop\PDP\flask\parkingSlotPosition", 'rb') as f:
   posList = pickle.load(f)
#Define width and height
width, height = 197, 48

"""# **Checking For Parking Space**"""

def checkParkingSpace(ImgPro):
    spaceCounter = 0
    for pos in posList:
        x, y = pos
        #Crop the image based on ROI
        imgCrop = ImgPro[y:y + height, x:x + width]

        # Counting the pixel values from cropped image
        count = cv2.countNonZero(imgCrop)
        if count < 900:
            color = (0, 255, 0)
            thickness = 5
            spacecounter += 1
        else:
            color = (0, 0, 255)
            thickness = 2
        #Draw the rectangle based on the condition defined above
        cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height), color,
thickness)
    #Display the available  parking slot count / total parking slot count
    cvzone.putTextRect(img, f'Free: {spaceCounter}/{len(posList)}', (100, 50),
scale=3, thickness=5, offset=20, colorR=(0,200,0))

"""# **Looping The Video**"""
```

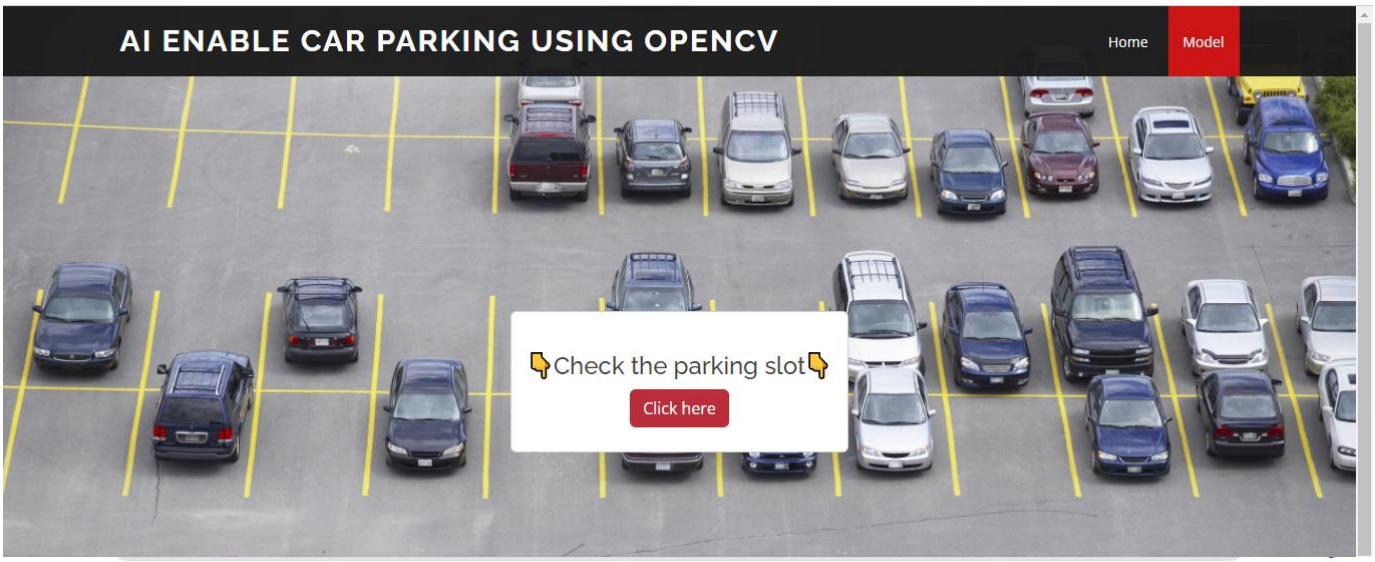
```
if cap.get(cv2.CAP_PROP_POS_FRAMES) == cap.get(cv2.CAP_PROP_FRAME_COUNT):
   cap.set(cv2.CAP_PROP_POS_FRAMES, 0)

"""# **Frame Processing And Empty Parking Slot Counters**"""

while True:
    #Looping the video
    if cap.get(cv2.CAP_PROP_POS_FRAMES) == cap.get(cv2.CAP_PROP_FRAME_COUNT):
        cap.set(cv2.CAP_PROP_POS_FRAMES, 0)

    #Reading frame by frame from video
    Success, img = cap.read()
    #Converting to gray scale image
    img = cv2.imread(r"C:/Users/PC/Desktop/PDP/flask/uploads/carParkImg.png")
    cv2.imshow('original',img)
    cv2.waitKey(0)
    imgGray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    imgBlur = cv2.GaussianBlur (imgGray, (3, 3), 1) # Applying blur to image
    # Applying threshold to the image
    imgThreshold = cv2.adaptiveThreshold(imgBlur, 255,
cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV, 25, 16)
    imgMedian= cv2.medianBlur (imgThreshold, 5)
    kernel = np.ones((3, 3), np.uint8)
    imgDilate = cv2.dilate(imgMedian, kernel, iterations=1)
    #Passing dilate image to the function
    checkParkingSpace(imgDilate)
    cv2.imshow("Image", img)
    cv2.waitKey(10)
```

Output

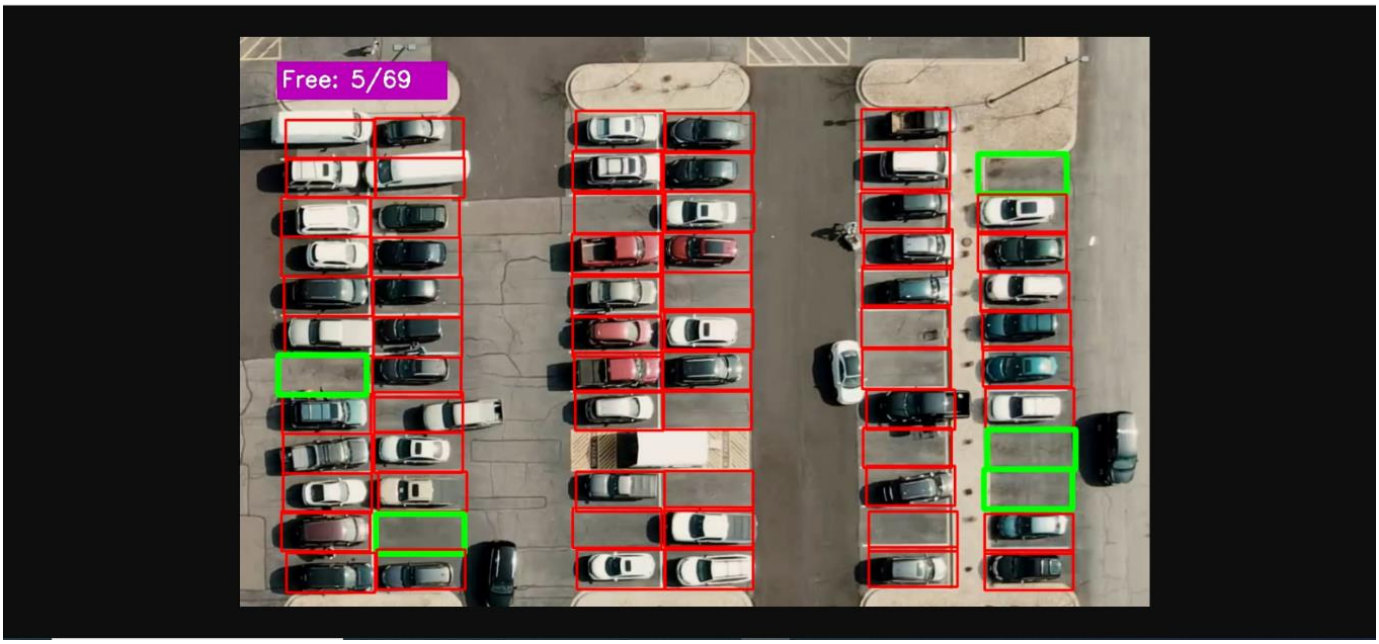

New Tab
Gmail
Maps
YouTube
II CSE Semester 3
IBM-Project-17704-...
Folder - Google Dri...
IBM
NextStep- Tata Con...
WhatsApp
https://auth0.opena...
SHPVIBE
Home
About Us
Model
Contact
REVOLUTIONIZING PARKING
AI Enable car parking using OpenCV
GET STARTED


New Tab
Gmail
Maps
YouTube
II CSE Semester 3
IBM-Project-17704-...
Folder - Google Dri...
IBM
NextStep- Tata Con...
WhatsApp
https://auth0.opena...
SHPVIBE
Home
About Us
Model
Contact
ABOUT
AI Enable car parking using OpenCV
The AI-Enabled Car Parking System Utilizing OpenCV Technology is a cutting-edge project that aims to revolutionize the way parking lots operate. This system uses OpenCV, a popular computer vision library, to enable vehicles to park autonomously.
The OpenCV library analyzes the footage and identifies the available parking spaces in the lot.
The system is designed to be highly accurate, and it can detect small and large vehicles, even in low-light conditions.

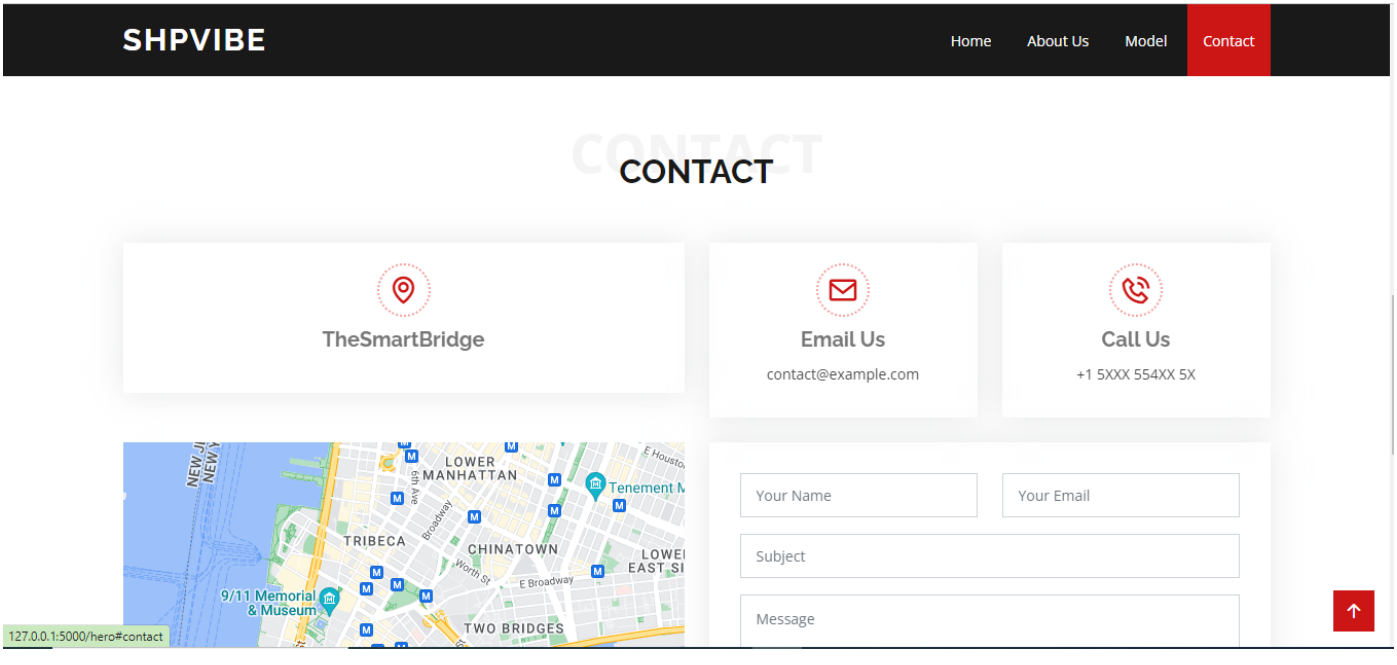
New Tab Gmail Maps YouTube II CSE Semester 3 IBM-Project-17704-... Folder - Google Dri... IBM NextStep- Tata Con... WhatsApp https://auth0.opena...



New Tab Gmail Maps YouTube II CSE Semester 3 IBM-Project-17704-... Folder - Google Dri... IBM NextStep- Tata Con... WhatsApp https://auth0.opena...



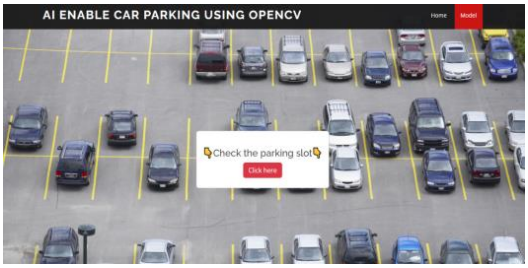
New Tab Gmail Maps YouTube II CSE Semester 3 IBM-Project-17704-... Folder - Google Dri... IBM NextStep- Tata Con... WhatsApp https://auth0.opena...



8.RESULTS

8.1 performance metrices

Model Performance Testing:

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Model Building	
2.	Accuracy	Validation Accuracy	
3.	Confidence Score (Only Yolo Projects)	Class Detected	

9. ADVANTAGES

- Real time monitoring
- Urban Space Management
- Time saving and vehicle management
- Real time availability information
- Reduce Traffic flows on residential streets.
- Vehicle secured

9.DIS-ADVANTAGES

- Expensive Construction & Installation
- Privacy and data security concerns
- Dependence on power and connectivity
- Limited scalability
- Requires Regular Maintenance
- Integration and compatibility issues

10.CONCLUSION

- AI-enabled car parking systems utilizing OpenCV (Open Source Computer Vision) technology offer a promising solution to the challenges of parking management. By leveraging computer vision algorithms and machine learning techniques, these systems provide advanced capabilities for parking space detection, occupancy monitoring, and overall parking efficiency.

- The development of intelligent parking systems by offering a wide range of image and video processing functions. It provides the necessary tools for object detection, recognition, and tracking, which are essential components of AI-enabled car parking systems.

- It enhances the accuracy and speed of parking space detection, reducing the time spent by drivers in searching for available spots. Real-time monitoring of parking occupancy enables better space management, maximizing the utilization of parking facilities

9. FUTURE WORK

For circle shape detection using Open CV python ,Real time detection from the video where a camera is used to capture that video, for parking slot detection is very useful for future work as automatic parking slot detection in a smart city,where time and energy both can save using a focus on theParking Deposit system. From this work as a future work trying to develop Graphical User Interface based simulation so get parking slot availability with the help of the message.Develop algorithms with minimum all correct detection of specific shapes, with minimum time .

That can be detected using OpenCV Python. So trying to reduce all this error as minimum as possible in future work. They may integrate with autonomous driving technologies, allowing vehicles to navigate parking lots autonomously and locate available spaces with minimal human intervention. This would not only enhance the convenience for drivers but also contribute to optimizing traffic flow and reducing congestion.

10.APPENDIX

GITHUB LINK:

https://github.com/naanmudhalvan-SI/PBL-NT-GP--5672-1680797853

PROJECT DEMO LINK:

https://youtu.be/9XJfFV5qcVg

THANK YOU !..