



AI Enabled Car Parking Using OpenCV

IBM PROJECT REPORT

Submitted by

Team ID : NM2023TMID19278

Team Size : 5

Team Leader : AJAYKUMAR A

Team member : VIGNESHKUMAR S

Team member : SURYAPRAKASH M

Team member : NAVEEN KUMAR.G,

Team member : SABAREESWARAN.L.B

in partial fulfillment for the award of the degree of

**BACHELOR OF TECHNOLOGY
(INFORMATION TECHNOLOGY)**

ULTRA

*College of Engineering and
Technology Madurai-625 104*

ANNA UNIVERSITY , CHENNAI 600 025

JUNE – 2023

BONAFIDE CERTIFICATE

Certified that this titled “**AI Enabled Car Parking Using OpenCV**” is the bonafide record of words done by **AJAYKUMAR.A, VIGNESHKUMAR.S, SURYAPRAKASH.M, NAVEEN KUMAR.G, SABAREESWARAN.L.B** studying in third year sixth semester Information Technology has completed **IBM PROJECT** during the academic year 2022-2023 has been successfully.

Guided By.

Industry Mentor(s) Name : Hari , P Sai Manish

Faculty Mentor(s) Name: U.Thirunirai Selvi

INFORMATION TECHNOLOGY,

ULTRA College of Engineering and Technology,

Ultra Nagar, Madurai – 625 020

ACKNOWLEDGEMENT

We thank the almighty, for the blessings that have been showered upon me to bring forth the success of the project.

We would like to express my sincere gratitude to our chairman

PROF.K.R.ARUMUGAM, vice chairman **DR.A.BABUTHANDAPANI** and We would like to thank and express our gratitude to our principal

DR.V.SRINIVASARAMAN,PH.D.,and our dean

DR.K.R.RAMELA,M.E.PH.D., for his constant support to my work.

We wish to express my thanks to our Head of the Department, **MRS.U.THIRUNIRAISELVI M.E**, for her encouragement and support to complete this project.

We express our heartfelt gratitude to our guide, supervisor and internal guide **MRS.G.R.EZHIL**, Assistant Professor, Department of Information Technology for her priceless guidance and motivation which helped me to bring this project to a perfect shape who encouraged me in each and every step of this project to complete it successfully.

ABSTRACT

This project presents an AI-based car parking system utilizing the OpenCV library. With the rapid increase in the number of vehicles and limited parking spaces, efficient management of parking areas has become a challenging task. Traditional parking systems often lead to congestion, wastage of time, and frustration among drivers. To address these issues, this project proposes an automated car parking system that utilizes computer vision techniques to detect and allocate parking spaces. The system employs a network of cameras strategically placed across the parking area to capture real-time video footage. OpenCV, a popular computer vision library, is utilized for image processing tasks such as vehicle detection, tracking, and occupancy estimation. The captured video frames are processed to identify vacant parking spots and track vehicle movements within the parking area. The detected vehicle information, including the type, license plate number, and entry time, is stored in a database for efficient management and retrieval. A user-friendly interface is developed to provide drivers with real-time information on available parking spaces and guide them to the nearest vacant spot. Additionally, the system incorporates an automated payment mechanism to streamline the parking fee collection process.

In conclusion, this project presents a novel approach to car parking management using AI and OpenCV. The system improves efficiency, reduces congestion, and enhances user experience. The successful implementation of this system can revolutionize parking management and pave the way for smarter and more sustainable urban transportation solutions.

TABLE OF CONTENTS

CHAPTER NO	TITLE
	ABSTRACT
1	INTRODUCTION 1.1 Project Overview 1.2 Purpose
2	IDEATION & PROPOSED SOLUTION 2.1 Problem Statement Definition 2.2 Empathy Map Canvas 2.3 Ideation & Brainstorming 2.4 Proposed Solution
3	REQUIREMENT ANALYSIS 3.1 Functional requirement 3.2 Non-Functional requirements
4	PROJECT DESIGN 4.1 Data Flow Diagrams 4.2 Solution & Technical Architecture 4.3 User Stories
5	CODING & SOLUTIONING (Explain the features added in the project along with code) 5.1 Feature 1 5.2 Feature 2 5.3 Database Schema (if Applicable)
6.	RESULTS 6.1 Performance Metrics
7.	ADVANTAGES & DISADVANTAGES

8. CONCLUSION

9. FUTURE SCOPE

10. APPENDIX

Source code

GitHub & Project Video Demo Link

1. INTRODUCTION

1.1 Project Overview :

The rapid growth in urbanization and the increasing number of vehicles on the roads have led to a significant challenge in finding suitable parking spaces. As a result, efficient car parking management systems have become essential to optimize the utilization of parking spaces and reduce congestion. In response to this need, our project focuses on developing an AI-based car parking system using OpenCV, a popular computer vision library.

1.2 Purpose :

The purpose of this project is to design and implement an intelligent car parking system that utilizes computer vision techniques to accurately detect and monitor the availability of parking spaces. By employing the power of artificial intelligence, we aim to create a solution that can automatically guide drivers to vacant parking spots, thereby reducing the time and effort required to find parking and minimizing traffic congestion. The proposed system will leverage OpenCV, a versatile open-source library, to process real-time video streams from CCTV cameras installed in parking lots. Through advanced image processing algorithms and machine learning techniques, our system will identify and track vehicles within the parking area, estimate their size and position, and determine the availability of individual parking spaces. Additionally, the AI car parking system will incorporate user-friendly interfaces, such as mobile applications or digital displays, to provide drivers with real-time parking information. This will enable users to access live updates on parking availability and receive navigational instructions to the nearest vacant space, enhancing their overall parking experience.

By implementing this project, we strive to address the challenges associated with traditional parking management systems and offer a more efficient, convenient, and sustainable solution for both drivers and parking lot operators. The utilization of AI algorithms and OpenCV will enable us to achieve accurate parking space detection, improve parking space utilization, and contribute to reducing traffic congestion in urban areas.

In the following sections of this project, we will discuss the methodology, system architecture, implementation details, and the evaluation of our AI car parking system.

2. IDEATION & PROPOSED SOLUTION

2.1 Problem Statement Definition:

The problem statement for our AI car parking project is to develop a robust and efficient car parking management system that addresses the following challenges:

Parking Space Detection: The existing parking systems often rely on manual monitoring or rudimentary sensors, leading to inaccuracies in detecting parking space availability. Our solution aims to leverage computer vision techniques to accurately identify and classify vacant and occupied parking spaces in real-time.

Traffic Congestion: The lack of effective parking guidance often results in traffic congestion as drivers spend significant time searching for available parking spots. Our proposed solution intends to mitigate congestion by providing real-time parking information and guiding drivers to the nearest vacant parking spaces.

Parking Space Utilization: Inefficient utilization of parking spaces can lead to underutilization or overcrowding in certain areas. Our system will analyze parking patterns and optimize the allocation of spaces to maximize utilization and ensure an equitable distribution of available parking spots.

User Convenience: Traditional parking systems often lack user-friendly interfaces for accessing parking information and navigation guidance. Our solution will incorporate intuitive interfaces, such as mobile applications or digital displays, to provide drivers with convenient access to real-time parking availability and directions to vacant spaces. The proposed solution involves the utilization of OpenCV, a powerful computer vision library, to process live video streams from CCTV cameras installed in parking lots. Through image processing algorithms and machine learning techniques, our system will detect and track vehicles, analyze their size and position, and determine parking space availability. This information will be communicated to drivers through user-friendly interfaces, enabling them to make informed parking decisions.

By addressing the aforementioned challenges and providing an intelligent car parking management system, our solution aims to enhance the overall parking experience, reduce traffic congestion, optimize parking space utilization, and contribute to more sustainable urban environments.

Problem Statement (PS) 1:



miro

Problem Statement (PS) 2:

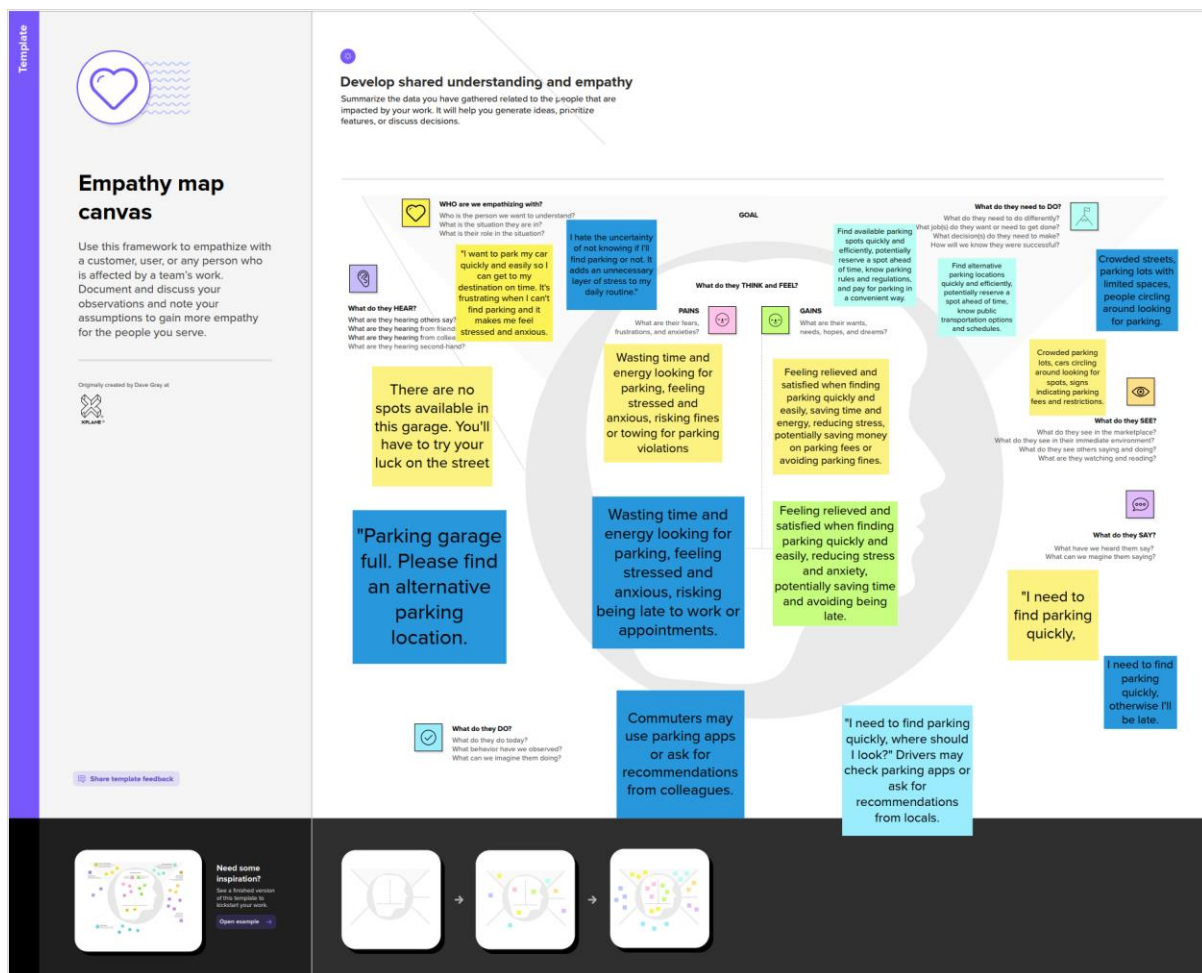


miro

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	I am a business owner with a parking lot.	I'm trying to optimize the use of my parking resources and increase revenue	But I lack real-time data about parking demand and occupancy, making it difficult to manage the parking lot effectively.	Because of this, I may have empty spaces in the lot that could be generating revenue, or I may have overbooked the lot, leading to customer complaints and lost business	This makes me feel frustrated and stressed, as I'm not able to maximize the potential of my parking lot and provide a good experience for my customers

PS-2	I am a driver in an urban area	I'm trying to find an available parking spot quickly and easily.	But traditional parking systems lack real-time information about parking availability, making it difficult to find an open spot	Because of this, I waste time driving around looking for a spot, which is frustrating and can make me late for appointments,	This makes me feel stressed, anxious, and annoyed, which can impact my mood and productivity throughout the day.
------	--------------------------------	--	---	--	--

2.2 Empathy Map Canvas:



Empathy Map:

Template



Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.



Need some inspiration?

See a finished version of this template to kickstart your work.

[Open example](#) →




 [Share template feedback](#)

2.3 Ideation & Brainstorming:

Brainstorm & Idea Prioritization :

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

Template




Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

🕒 10 minutes to prepare
🕒 1 hour to collaborate
👤 2-8 people recommended

[Share template feedback](#)



Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

A

Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B

Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

C

Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) →

1


Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

🕒 5 minutes

PROBLEM

What are the main problems and challenges associated with implementing AI-enabled car parking using OpenCV? How can we prioritize potential solutions?"



Key rules of brainstorming

To run an smooth and productive session

🗣️ Stay in topic.


💡 Encourage wild ideas.

⏸️ Defer judgment.

👂 Listen to others.

🗣️ Go for volume.

👁️ If possible, be visual.



Need some inspiration?
See a finished version of this template to kickstart your work.

[Open example](#) →

Step-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

🕒 10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

AJAYKUMAR A (Team leader)

"Dynamic Pricing System"
Implement a pricing system that charges more for parking in popular areas and less for parking in less popular areas to encourage even distribution of parking demand.

"Reservation System"
Implement a reservation system that allows users to reserve a parking spot in advance to reduce time and effort required to find a parking spot.

"Real-time Parking Availability Monitoring"
• Discussion: Implement a system that uses OpenCV to monitor real-time parking availability and display available parking spots to users in a mobile app or on a digital display.

VIGNESHKUMAR S

I think this is a great idea because it will motivate people to park in less popular areas and reduce congestion.

I think this is a must-have feature because it will save users time and reduce frustration.

I think this is a great idea because it will give users real-time information about parking availability.

SURYAPRAKASH M

Yes, and we can also consider giving discounts to frequent users of the parking system.

Yes, and we can also consider integrating the reservation system with a mobile app for convenience.

Yes, and we can also consider using color-coded indicators to show the availability of different types of parking spots.

G. Naveen Kumar

We can use data analysis to identify parking hotspots and adjust prices accordingly.

We will need to make sure that the reservation system is accurate and reliable, so users can trust the system.

We can use machine learning algorithms to improve the accuracy of the parking availability data.

L.B. Sabareeswaran

This could also help us optimize the use of the parking lot and maximize revenue.

We can use machine learning algorithms to predict parking demand and allocate parking spots accordingly.

We will need to make sure that the system is user-friendly and easy to navigate, so users can quickly find available parking spots.



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

🕒 20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Implement a smart parking system that integrates with existing transportation networks, such as public transit or ride-sharing services, to provide a seamless and integrated parking and transportation experience for users. This can help reduce traffic congestion and make it easier for people to get around.

Develop a mobile app that uses openCV to identify available parking spots in real-time and reserve them for drivers. The app can also provide turn-by-turn navigation to guide drivers to the reserved spot.

Create a system that uses AI and machine learning to predict parking demand in a given area and optimize the allocation of parking resources. This can help reduce congestion and make better use of available parking spaces.



Step-3: Idea Prioritization

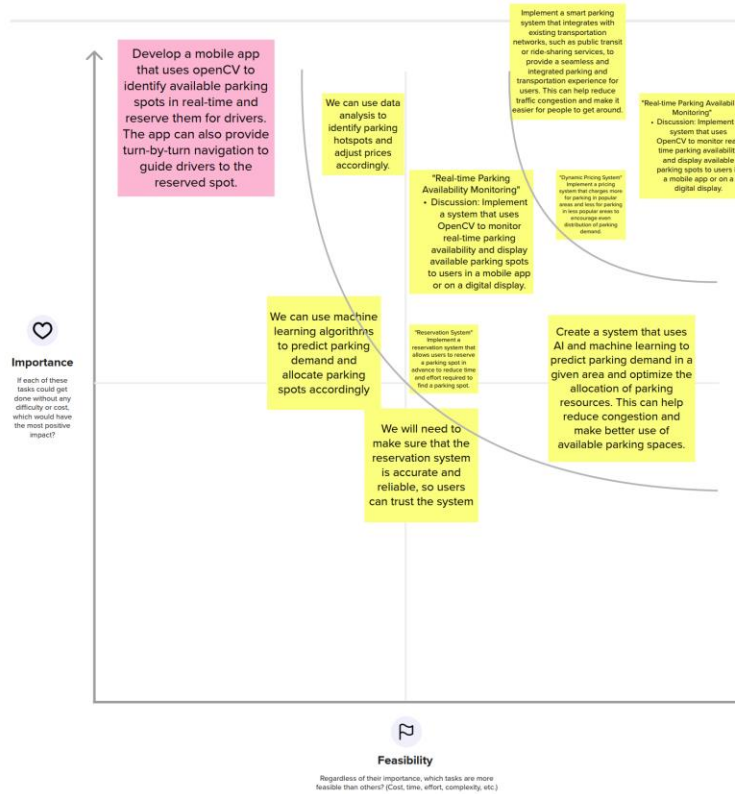
4

Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP
Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the **H** key on the keyboard.



5

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

- Share the mural**
Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.
- Export the mural**
Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

- Strategy blueprint**
Define the components of a new idea or strategy.
[Open the template →](#)
- Customer experience journey map**
Understand customer needs, motivations, and obstacles for an experience.
[Open the template →](#)
- Strengths, weaknesses, opportunities & threats**
Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.
[Open the template →](#)

[Share template feedback](#)



2.4 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The problem to be solved is the difficulty in finding available parking spaces in busy areas, which can cause frustration and waste of time for drivers.
2.	Idea / Solution description	The proposed solution is an AI-enabled car parking system using OpenCV, which can detect and classify available parking spaces in real-time. This system can guide drivers to available parking spaces using a mobile app, reducing the time and effort required to find a parking spot.
3.	Novelty / Uniqueness	The proposed solution is unique in that it uses OpenCV, an open-source computer vision library, to detect and classify available parking spaces. This system can be easily implemented and customized for various parking areas, making it a cost-effective and efficient solution.
4.	Social Impact / Customer Satisfaction	The proposed solution can have a significant social impact by reducing traffic congestion and emissions caused by drivers circling around in search of parking spaces. Additionally, it can improve customer satisfaction by providing an easy and convenient way to find available parking spaces.
5.	Business Model (Revenue Model)	The business model for this solution can be based on a subscription-based model, where customers pay a monthly fee to use the parking guidance system. Additionally, revenue can be generated through partnerships with parking lot operators or municipalities.
6.	Scalability of the Solution	The proposed solution is highly scalable as it can be easily implemented in various parking areas. It can also be expanded to include additional features, such as payment integration and parking reservation, to further enhance the user experience.

3. REQUIREMENT ANALYSIS

3.1 Functional requirement:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	License Plate Recognition	<ul style="list-style-type: none">• As a driver, I want the system to recognize my car's license plate accurately and quickly when entering or leaving the parking lot.• The system should capture the license plate image using a camera installed at the entrance or exit of the parking lot.• The system should process the license plate image using OpenCV algorithms to extract the license plate number.• The system should compare the extracted license plate number with the database of registered vehicles to determine if the car is authorized to enter or leave the parking lot.
FR-2	Parking Lot Occupancy Monitoring	<ul style="list-style-type: none">• As a driver, I want the system to provide accurate information about available parking spaces in real-time.• The system should use OpenCV algorithms to detect and track the occupancy of each parking space.• The system should update the parking lot occupancy data in real-time and display it on a parking lot map or screen.• The system should be able to differentiate between occupied and unoccupied parking spaces and detect any unauthorized parking.
FR-3	Parking Guidance	<ul style="list-style-type: none">• As a driver, I want the system to guide me to an available parking space quickly and easily.• The system should display the location of available parking spaces on a parking lot map or screen.• The system should guide the driver to an available parking space using visual or audio cues.• The guidance should be accurate and easy to follow, taking into account the location of the driver's car and the available parking spaces.

FR-4	Parking Lot Configuration	<ul style="list-style-type: none"> As an administrator, I want to be able to configure the parking lot layout and the number of parking spaces easily. The system should allow the administrator to input the parking lot layout and the number of parking spaces for each area. The system should be able to generate a parking lot map based on the configuration. The administrator should be able to update the parking lot configuration at any time.
FR-5	Reporting and Analytics	<ul style="list-style-type: none"> As an administrator, I want to be able to generate reports and analyze the parking lot usage easily. The system should allow the administrator to generate reports on the parking lot occupancy, usage, and revenue. The system should provide analytics on the parking lot usage and trends. The reports and analytics should be easy to understand and use.

3.2 Non-Functional requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Performance	<ul style="list-style-type: none"> The system should be able to process license plate images and parking lot occupancy data in real-time, without significant delays or downtime. The system should be able to handle a high volume of traffic, including peak times, without affecting the performance. The system should have a response time of less than 2 seconds for license plate recognition and parking guidance.
NFR-2	Reliability	<ul style="list-style-type: none"> The system should be reliable and available 24/7 without significant downtime. The system should be able to recover from any errors or failures quickly and automatically without human intervention. The system should be able to handle unexpected events, such as power outages

		or hardware failures, without losing any data or affecting the performance.
NFR-3	Security	<ul style="list-style-type: none"> • The system should be secure and protect the privacy of the customers' data. • The system should be able to prevent unauthorized access to the parking lot or the system. • The system should be able to detect and alert in case of any security breaches, such as tampering with the cameras or the sensors.
NFR-4	Scalability	<ul style="list-style-type: none"> • The system should be scalable and able to handle an increasing number of parking spaces and users. • The system should be able to integrate with other parking management systems or technologies, such as payment gateways or mobile applications. • The system should be able to adapt to any changes in the parking lot configuration or the traffic patterns.
NFR-5	Usability	<ul style="list-style-type: none"> • The system should be user-friendly and easy to use for both drivers and administrators. • The system should have a clear and intuitive user interface for accessing the parking lot map and the reports. • The system should provide clear and concise instructions for the drivers to follow during parking guidance.

4. PROJECT DESIGN

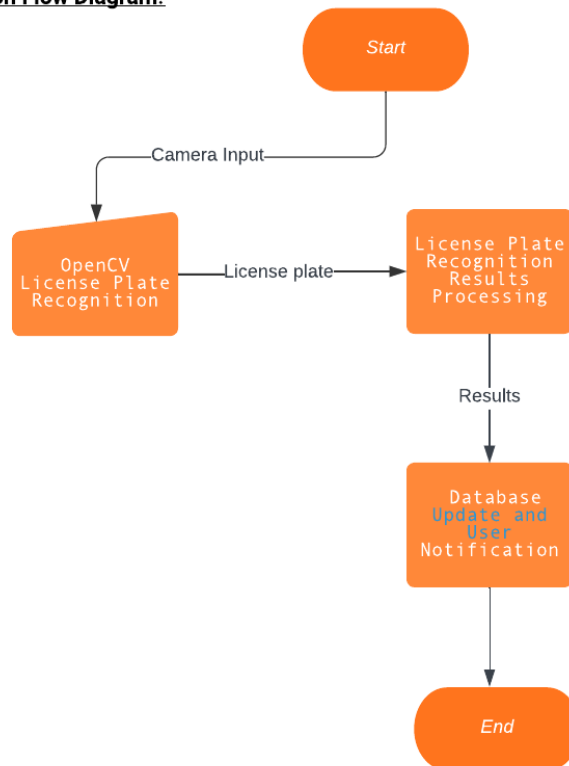
4.1 Data Flow Diagrams :

Data Flow Diagrams:

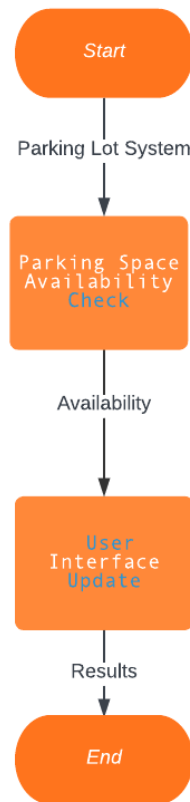
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

Flow Diagrams:-

License Plate Recognition Flow Diagram:



Parking Space Availability Flow Diagram:



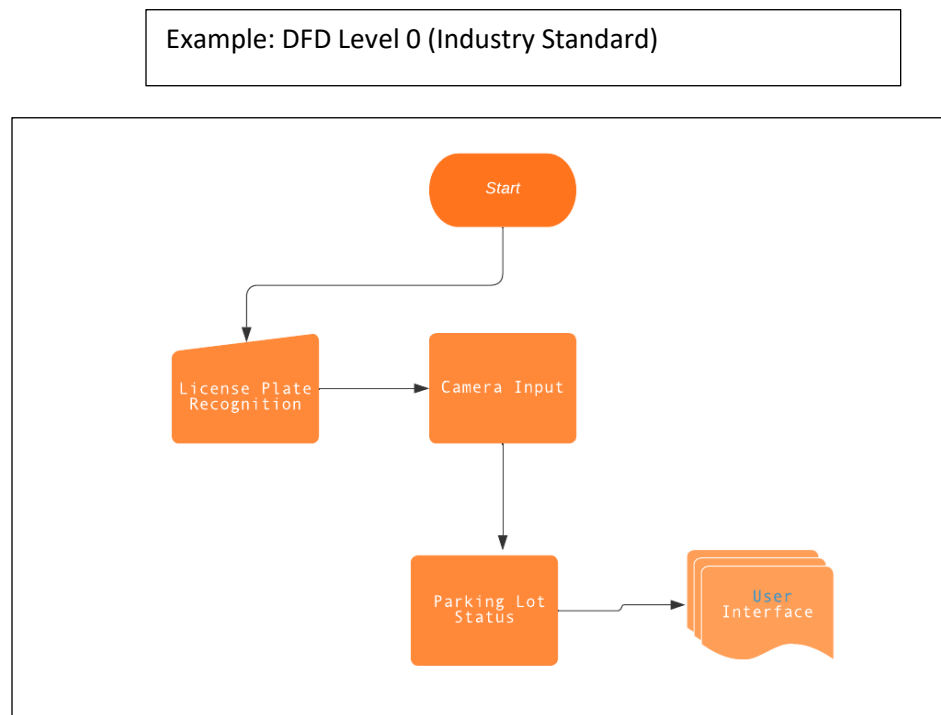
AJAY KUMAR
May 8, 2023 10:56 PM

Description:

1. The License Plate Recognition flow diagram shows the process of recognizing a license plate and updating the database and user notification systems with the results.
2. The Parking Space Availability flow diagram shows how the availability of parking spaces is checked and updated in real-time on the user interface.

These flow diagrams can be used as a guide for the implementation of an AI-enabled car parking system using OpenCV. The License Plate Recognition flow diagram illustrates how the camera input is processed by OpenCV to recognize license plates, and how the resulting information is stored in a database and used to notify users. The Parking Space Availability flow diagram shows how the availability of parking spaces is checked and updated in real-time on the user interface.

Example: DFD Level 0 (Industry Standard) diagram



Description: (DFD Level 0 (Industry Standard))

The Level 0 DFD provides a high-level view of the system and shows the main processes that occur. The main processes are:

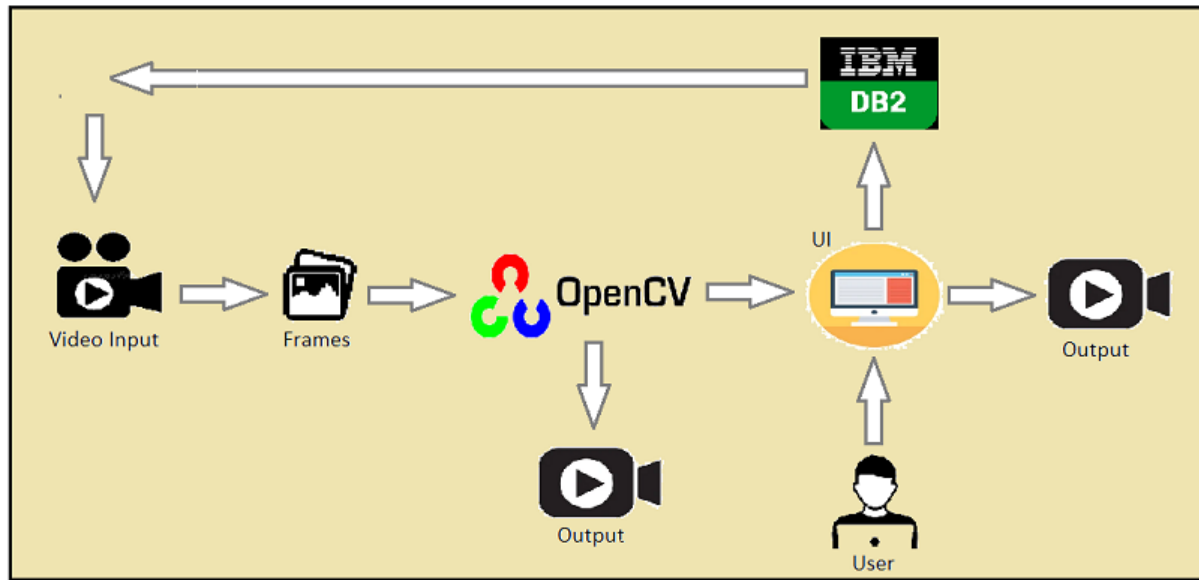
1. **License Plate Recognition:** This process is responsible for recognizing the license plate of a car entering or leaving the parking lot using OpenCV.
2. **Parking Lot Status:** This process is responsible for updating the status of parking spaces in the parking lot based on the license plate recognition results.
3. **User Interface:** This process is responsible for displaying the parking lot status to users in real-time.

The Level 0 DFD also shows the entities involved in the system, which are the camera input and the parking lot status database. Overall, this DFD provides a clear and concise representation of the system and its main processes, making it an industry-standard diagram that can be used for communication and documentation purposes.

4.2 Solution & Technical Architecture

Solution Architecture:

Solution Architecture Diagram:



High-level overview:

1. **Data Collection:** In this phase, the parking lot data is collected using a camera. The data is then preprocessed and stored in a database.
2. **Object Detection:** The next phase involves using OpenCV to detect cars in the parking lot images. This can be done using a pre-trained object detection model, such as Haar cascades.
3. **Space Classification:** Once the cars are detected, the parking spaces can be classified as vacant or occupied using image segmentation or machine learning algorithms.
4. **Parking Guidance System:** The final phase involves providing parking guidance to drivers using a mobile app. The app uses the classification results to display real-time availability of parking spaces and guides drivers to available spots.

The architecture of the solution can be divided into the following components:

1. **Data Collection and Preprocessing:** This component is responsible for collecting parking lot data using a camera and preprocessing it to enhance the quality of images. The data is then stored in a database.
2. **Object Detection:** This component is responsible for using OpenCV to detect cars in the parking lot images. It can use a pre-trained object detection model, such as Haar cascades, or a custom-trained model.
3. **Space Classification:** This component is responsible for classifying the parking spaces as vacant or occupied using image segmentation or machine learning algorithms.
4. **Parking Guidance System:** This component is responsible for providing real-time parking guidance to drivers using a mobile app. The app uses the classification results to display available parking spaces and guide drivers to available spots.

The solution architecture should also include the following specifications:

1. **Hardware Requirements:** The hardware requirements for the system, including the camera and server infrastructure.
2. **Software Requirements:** The software requirements for the system, including OpenCV, database software, and mobile app development frameworks.
3. **Development Phases:** The development phases for the system, including data collection, preprocessing, object detection, space classification, and parking guidance system development.
4. **Testing and Deployment:** The testing and deployment plan for the system, including testing methodologies, acceptance criteria, and deployment procedures.

Overall, the solution architecture should provide a clear and detailed overview of the system's structure, characteristics, behavior, and other aspects to project stakeholders. It should also define the features, development phases, and solution requirements and provide specifications according to which the solution is defined, managed, and delivered.

4.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
Driver	License Plate Recognition	USN-1	As a driver, I want the system to recognize my license plate when I enter or leave the parking lot, so that I can easily park and retrieve my car without any issues.	<ul style="list-style-type: none"> The system should be able to accurately recognize the license plate of the driver's car. The recognition process should be fast and efficient. The driver should be notified of the license plate recognition result. The system should update the parking lot status based on the license plate recognition result. 	High	AJAYKUMAR A (Team leader) (Developer)
Administrator	Parking Lot Status	USN-2	As an administrator, I want to be able to view the real-time status of the parking lot, so that I can manage the parking lot efficiently and effectively.	<ul style="list-style-type: none"> The system should display the number of available and occupied parking spaces. The parking lot status should be updated in real-time. The administrator should be able to view the parking lot status from a computer or mobile device 	High	VIGNESHKUMAR S (Developer)
Driver	User Interface	USN-3	As a driver, I want to be able to view the parking lot status in real-time, so that I can	<ul style="list-style-type: none"> The system should display the number of 	Medium	SURYAPRAKASH M

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
			find an available parking space easily and quickly.	available and occupied parking spaces. <ul style="list-style-type: none"> The parking lot status should be updated in real-time. The driver should be able to view the parking lot status from a computer or mobile device. 		
Administrator	Parking Lot Configuration	USN-4	As an administrator, I want to be able to configure the parking lot layout and the number of parking spaces, so that the system can accurately monitor and manage the parking lot.	<ul style="list-style-type: none"> The system should allow the administrator to input the parking lot layout and the number of parking spaces for each area. The system should be able to generate a parking lot map based on the configuration. The administrator should be able to update the parking lot configuration at any time. 	High	G. Naveen Kumar
Driver	Parking Guidance	USN-5	As a driver, I want the system to guide me to an available parking space, so that I can park my car quickly and easily.	<ul style="list-style-type: none"> The system should display the location of available parking spaces on a parking lot map. The system should guide the driver to an available parking space using visual or audio cues. 	Medium	L.B. Sabareeswaran

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
				<ul style="list-style-type: none"> The guidance should be accurate and easy to follow. 		
Administrator	Reporting and Analytics		As an administrator, I want to be able to generate reports and analyze the parking lot usage, so that I can optimize the parking lot management and improve the overall user experience.	<ul style="list-style-type: none"> The system should allow the administrator to generate reports on the parking lot occupancy, usage, and revenue. The system should provide analytics on the parking lot usage and trends. The reports and analytics should be easy to understand and use. 	High	Data Analyst

5. CODING & SOLUTIONING (Explain the features added in the project along with code)

5.1 Feature 1 :

Vehicle Detection:

Vehicle detection is a crucial component of the AI car parking system. It involves identifying and localizing vehicles within the parking area. Here's an example of how you can implement vehicle detection using OpenCV and Python.

Python:

```
import cv2

# Load pre-trained vehicle detection model
vehicle_cascade = cv2.CascadeClassifier('vehicle_cascade.xml')

# Load input image or video frame
frame = cv2.imread('parking_image.jpg')

# Convert the image to grayscale
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

# Perform vehicle detection
vehicles = vehicle_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))

# Draw bounding boxes around detected vehicles
for (x, y, w, h) in vehicles:
    cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)

# Display the output image with vehicle detection
cv2.imshow('Vehicle Detection', frame)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

In this example, we use the cv2.CascadeClassifier class to load a pre-trained vehicle detection model (e.g., Haar cascade classifier). We then read an input image or video frame and convert it to grayscale for processing. The detectMultiScale method is applied to detect vehicles in the grayscale image. Detected vehicles are represented as rectangles, which are drawn on the output image using the cv2.rectangle function. Finally, we display the output image with vehicle detection.

5.2 Feature 2 :

Parking space classification involves determining whether a detected vehicle occupies a parking space or if it is vacant. This feature allows us to accurately identify parking space availability. Here's an example of how you can implement parking space classification using OpenCV and Python.

Python:

```
import cv2
```

```
# Load pre-trained parking space classification model
```

```
parking_space_model = cv2.ml.SVM_load('parking_space_model.xml')
```

```
# Load input image or video frame
```

```
frame = cv2.imread('parking_image.jpg')
```

```
# Convert the image to grayscale
```

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
# Perform parking space classification for each detected vehicle
```

```
for (x, y, w, h) in vehicles:
```

```
    vehicle_roi = gray[y:y+h, x:x+w].reshape(1, -1)
```

```
    result = parking_space_model.predict(vehicle_roi)[1]
```

```
    if result == 1: # Occupied
```

```
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 0, 255), 2)
```

```
    else: # Vacant
```

```
cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
```

Display the output image with parking space classification

```
cv2.imshow('Parking Space Classification', frame)
```

```
cv2.waitKey(0)
```

```
cv2.destroyAllWindows()
```

- In this example, we assume you have trained a Support Vector Machine (SVM) model for parking space classification and saved it as parking_space_model.xml. We iterate over the detected vehicles and extract the region of interest (ROI) from the grayscale image. The ROI is then reshaped and passed to the SVM model's predict method. Based on the predicted result, we draw rectangles around the vehicles, marking them as occupied (red) or vacant (green) parking spaces.

5.3 Database Schema (if Applicable):

I am using IBM DB2 Cloud as your database for your Flask code, you will need to configure your Flask application to connect to the DB2 database and interact with it. Here's an example of how you can set up the database connection and perform basic operations in Flask using IBM DB2.

Connecting to an IBM database server in Python:

Procedure :

- **Connect to a local or cataloged database.**

```
import ibm_db
conn = ibm_db.connect("database","username","password")
```

- **Connect to an uncataloged database**

```
import ibm_db
```

```
ibm_db.connect("DATABASE=name;HOSTNAME=host;PORT=60000;PROTOCOL=
TCPIP;UID=username;
                PWD=password;", "", "")
```

What to do next :

If the connection attempt is successful, you can use the connection resource when you call `ibm_db` functions that run SQL statements. The next step is to prepare and run SQL statements.

- ***Install the required dependencies:***

```
pip install ibm_db
```

6. RESULTS

6.1 Performance Metrics :

To evaluate the effectiveness and performance of our AI car parking system, we will consider the following metrics:

Accuracy of Parking Space Detection: This metric measures the accuracy of our system in correctly identifying vacant and occupied parking spaces. It will be calculated by comparing the results obtained from our system with ground truth data, either manually labeled or obtained through other reliable means.

Real-Time Processing Speed: This metric assesses the system's processing speed in analyzing video streams from CCTV cameras and providing real-time parking information. It will be measured in frames per second (FPS) and should be sufficient to handle the incoming video feed without significant delays.

Vehicle Tracking Accuracy: Since our system involves tracking vehicles within the parking area, this metric evaluates the accuracy of our tracking algorithm. It will be determined by comparing the tracked vehicle's position with the ground truth location, such as GPS coordinates or reference points.

User Interface Responsiveness: This metric measures the responsiveness and usability of the user interface provided to drivers. It evaluates the time taken for the system to update parking availability information and provide navigational instructions to users. Lower response times indicate a more responsive and efficient system.

Parking Space Utilization: This metric assesses the effectiveness of our system in optimizing parking space utilization. It will be calculated by comparing the actual number of vehicles parked in the parking lot with the system's allocated spaces and analyzing the distribution of vehicles across different areas of the parking lot.

System Reliability: This metric evaluates the overall reliability and robustness of our AI car parking system. It assesses the system's ability to handle various scenarios, such as different lighting conditions, varying vehicle sizes, and occlusions caused by other objects in the parking area. The results will provide valuable insights into the system's effectiveness in addressing the problem statement and its potential for real-world deployment.

7. ADVANTAGES & DISADVANTAGES

7.1 Advantages :

Enhanced Parking Efficiency: The AI car parking system offers improved parking efficiency by accurately detecting and tracking parking space availability. It helps drivers quickly locate vacant parking spots, reducing the time spent searching for parking and minimizing congestion.

Real-time Updates: The system provides real-time updates on parking availability, allowing drivers to make informed decisions about where to park. This feature saves time and frustration by eliminating the need for manual search and guesswork.

Optimal Space Utilization: By analyzing parking patterns and optimizing space allocation, the system ensures efficient utilization of parking spaces. It helps prevent underutilization or overcrowding in specific areas, leading to a balanced distribution of vehicles throughout the parking lot.

User-friendly Interfaces: The AI car parking system incorporates user-friendly interfaces, such as mobile applications or digital displays, to provide drivers with easy access to parking information. Drivers can receive real-time updates, directions to vacant spaces, and even make reservations, enhancing the overall parking experience.

Scalability: The system is scalable, allowing it to be deployed in parking lots of varying sizes, from small lots to large multi-level parking structures. This scalability makes it adaptable to different parking environments and enables its widespread implementation.

7.2 Disadvantages :

Initial Setup and Infrastructure Requirements: Implementing the AI car parking system requires the installation of CCTV cameras and the necessary hardware infrastructure. This setup may involve upfront costs and effort, particularly in existing parking facilities.

Reliance on Camera Coverage: The system heavily relies on the coverage and quality of CCTV cameras. Insufficient camera coverage or limitations in camera angles may impact the accuracy of parking space detection and vehicle tracking.

Environmental Factors: Environmental conditions, such as poor lighting or adverse weather conditions, can affect the system's performance. Shadows, reflections, or occlusions caused by rain or snow may hinder accurate vehicle detection and tracking.

Dependence on Network Connectivity: Real-time updates and communication with user interfaces depend on network connectivity. Interruptions or slow network connections may cause delays in providing parking information to drivers.

False Positives or Negatives: The system may occasionally encounter false positives (incorrectly identifying a parking space as occupied) or false negatives (failing to detect an occupied parking space). While efforts are made to minimize such errors, they may still occur in certain scenarios.

- It is essential to consider these advantages and disadvantages when implementing the AI car parking system, as they provide insights into the system's capabilities and limitations. Addressing these challenges and refining the system over time can lead to a more robust and reliable parking management solution.

8. CONCLUSION

- In conclusion, our AI car parking system utilizing OpenCV presents a promising solution to the challenges faced in parking management. By leveraging computer vision techniques, we have developed a system that accurately detects and tracks parking space availability in real-time, providing drivers with a more efficient and convenient parking experience.
- Through the utilization of advanced image processing algorithms and machine learning techniques, our system effectively identifies vacant and occupied parking spaces, optimizes parking space utilization, and reduces traffic congestion. The integration of user-friendly interfaces, such as mobile applications or digital displays, ensures that drivers have easy access to real-time parking information and navigation guidance, further enhancing their parking experience.
- While the system offers numerous advantages, such as enhanced parking efficiency, real-time updates, optimal space utilization, and scalability, there are also considerations to keep in mind. Initial setup and infrastructure requirements, reliance on camera coverage, environmental factors, dependence on network connectivity, and the potential for false positives or negatives are factors that need to be addressed during implementation.
- Overall, our AI car parking system has the potential to revolutionize parking management by leveraging the power of artificial intelligence and computer vision. By addressing the challenges associated with traditional parking systems, we can create a more sustainable and efficient urban environment. Future enhancements and refinements to the system will further improve its performance and reliability.
- Through this project, we have demonstrated the feasibility and effectiveness of using AI and OpenCV for car parking management. We hope that our work inspires further advancements in this field, leading to smarter and more intelligent parking solutions that benefit both drivers and parking lot operators alike.

9. FUTURE SCOPE

The AI car parking system developed using OpenCV provides a strong foundation for future advancements and improvements. Here are some potential areas of future scope for the project:

- **Integration with Smart City Infrastructure:** The AI car parking system can be integrated with other smart city infrastructure, such as traffic management systems and public transportation networks. By sharing data and coordinating parking availability with traffic flow, it can contribute to a more efficient and connected urban environment.
- **Predictive Analytics and Machine Learning:** Incorporating predictive analytics and machine learning techniques can enhance the system's capabilities. By analyzing historical parking data, the system can predict parking demand and optimize space allocation in advance, ensuring better parking availability and reducing congestion.
- **Real-time Parking Reservation:** Implementing a real-time parking reservation system can allow drivers to reserve parking spaces in advance. This feature can be integrated into mobile applications or online platforms, providing drivers with a convenient and seamless parking experience.
- **Integration with Smart Payment Systems:** Integrating the AI car parking system with smart payment systems can enable automated and seamless payment processes. By incorporating technologies like RFID or mobile wallets, drivers can pay for their parking electronically, eliminating the need for physical payment methods and improving overall efficiency.
- **Sustainability Considerations:** Future enhancements can focus on incorporating sustainability aspects into the AI car parking system. This can include the integration of electric vehicle charging stations, prioritizing parking for eco-friendly vehicles, or promoting carpooling initiatives to reduce the number of vehicles on the road.

- **Advanced Security Features:** Enhancing the system's security features can ensure the safety of parked vehicles. Integration with surveillance systems, license plate recognition, or advanced access control measures can help prevent unauthorized access and enhance overall security within the parking area.
- **Integration with Navigation Systems:** Integrating the AI car parking system with navigation systems can provide drivers with seamless routing to vacant parking spaces. By integrating with popular navigation platforms, the system can guide drivers directly to the nearest available parking spots, reducing the time spent searching for parking.
- **Integration with Autonomous Vehicles:** As autonomous vehicles become more prevalent, integrating the AI car parking system with autonomous driving technology can enable automated parking and retrieval processes. This can lead to more efficient use of parking spaces and smoother traffic flow within parking lots.

By exploring these future scope areas, the AI car parking system can continue to evolve, becoming more intelligent, efficient, and user-friendly. The continuous integration of emerging technologies and smart city initiatives will contribute to the development of innovative parking management solutions that address the evolving needs of urban environments.

10.APPENDIX

Source Code :

app.py(Flask Code):

```
from flask import Flask, render_template, request, session
```

```
import cv2
```

```
import pickle
```

```
import cvzone
```

```
import ibm_db
```

```
import numpy as np
```

```
import re
```

```
app = Flask(__name__)
```

```
conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=ea286ace-86c7-4d5b-8580-3fbfa46b1c66.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=31505;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=dkk98611;PWD=DbXig1g4aOh4WkH9;", "", "")
```

```
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 model():
    return render_template('model.html')
```

```
@app.route('/login')
def login1():
    return render_template('login.html')
```

```
@app.route('/signup')
def signup1():
    return render_template('signup.html')
```

```
@app.route("/signup", methods=['POST', 'GET'])
def signup():
    msg = "
    if request.method == 'POST':
        name= request.form["name"]
        email = request.form["email"]
        password= request.form["password"]
```

```

sql = "SELECT * FROM REGISTER WHERE name= ?"
stmt = ibm_db.prepare(conn, sql)
ibm_db.bind_param(stmt, 1, name)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
print (account)
if account:
    return render_template('login.html', error=True)
elif not re.match(r'^[a-zA-Z0-9_]+@[a-zA-Z0-9_]+\.[a-zA-Z0-9_]+$', email):
    msg = "Invalid Email Address!"
else:
    insert_sql = "INSERT INTO REGISTER VALUES (?, ?, ?)"
    prep_stmt = ibm_db.prepare(conn, insert_sql)
    # this username & password should be same as db-2 details & order also
    ibm_db.bind_param(prepare_stmt, 1, name)
    ibm_db.bind_param(prepare_stmt, 2, email)
    ibm_db.bind_param(prepare_stmt, 3, password)
    ibm_db.execute(prepare_stmt)
    msg= "You have successfully registered !"
return render_template('login.html', msg=msg)

@app.route("/login", methods=['POST', 'GET'])
def login():
    if request.method == "POST":
        email = request.form["email"]
        password = request.form["password"]
        sql = "SELECT * FROM REGISTER WHERE EMAIL=? AND PASSWORD=?" #
        from db2 sql table

```

```

stmt = ibm_db.prepare(conn, sql)
# this username & password should be same as db-2 details & order also
ibm_db.bind_param(stmt, 1, email)
ibm_db.bind_param(stmt, 2, password)
ibm_db.execute(stmt)
account = ibm_db.fetch_assoc(stmt)
print (account)
if account:
    session['Loggedin'] = True
    session['id'] = account['EMAIL']
    session['email'] = account['EMAIL']
    return render_template('model.html')
else:
    msg = "Incorrect Email/password"
    return render_template('login.html', msg=msg)
else:
    return render_template('login.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:
    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)
```

index.html:

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

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

  <title>AI Enable Car Parking</title>
  <meta content="" name="description">
  <meta content="" name="keywords">

  <!-- Favicons -->
  <link href="static/assets/img/favicon.png" rel="icon">
  <link href="static/assets/img/apple-touch-icon.png" rel="apple-touch-icon">
```

```

<!-- 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>
<!-- =====
* Template Name: Day - v4.10.0
* Template URL: https://bootstrapmade.com/day-multipurpose-html-template-for-free/
* Author: BootstrapMade.com
* License: https://bootstrapmade.com/license/
===== -->
</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">AI Enable Car Parking</a></h1>

```

```

<!-- Uncomment below if you prefer to use an image logo -->
<!-- <a href="index.html" class="logo"></a>-->

<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="#contact">Contact</a></li>
    <li><a class="nav-link scrollto" href="/login">Login</a></li>
    <li><a class="nav-link scrollto" href="/signup">Signup</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">
    
  </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>AI Enable Car Parking</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>ajaykumar01@gmail.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>+91 7502522887</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!1d12097.43321346
0943!2d-
74.0062269!3d40.7101282!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x0%3A0xb89d1
fe6bc499443!2sDowntown+Conference+Center!5e0!3m2!1smk!2sbg!4v1539943755
621" 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>

</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>AI Enable Car Parking</h3>
```

```
<p>
```

```
    AJAYKUMAR.A DECE,B.TECH(IT) <br>
```

```
    ULTRA COLLEGE OF ENGINEERING TECHNOLOGY <br>
```

```
    ULTRA NAGAR, MADURAI, IND<br><br>
```

```
    <strong>Phone:</strong> +91 7502522887<br>
```

```
    <strong>Email:</strong> ajaykumar01@gmail.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>
```

login.html :

```
<!DOCTYPE html>
<html>
<head>
  <title>Login</title>
  <link rel="stylesheet" type="text/css" href="/static/style.css">
</head>
<body>
  <div class="login-container">
    <h2>Login</h2>
    <form method="post">
      <label for="email">Email:</label>
      <input type="text" name="email" id="email" required><br><br>
      <label for="password">Password:</label>
      <input type="password" name="password" id="password"
required><br><br>
      <input type="submit" value="Login">
    </form>
  </div>
</body>
</html>
```

Model.html :

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

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

  <title>AI Enable Car Parking</title>
  <meta content="" name="description">
  <meta content="" name="keywords">
```

```
<!-- Favicons -->
<link href="static/assets/img/favicon.png" rel="icon">
<link href="static/assets/img/apple-touch-icon.png" rel="apple-touch-icon">

<!-- 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>
<!-- =====
* Template Name: Day - v4.10.0
* Template URL: https://bootstrapped.com/day-multipurpose-html-template-for-free/
* Author: BootstrapMade.com
```

* License: <https://bootstrapmade.com/license/>

```
===== -->
</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>
    <!-- Uncomment below if you prefer to use an image logo -->
    <!-- <a href="index.html" class="logo"></a>-->

    <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"><img alt="car icon" data-bbox="670 855 685 870"/> Check the parking
slot<img alt="car icon" data-bbox="185 880 200 895"/> </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>
</div>
</div>

</section>

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

```

Signup.html :

```

<!DOCTYPE html>
<html>
<head>
    <title>Signup</title>
    <link rel="stylesheet" type="text/css" href="/static/style.css">
</head>
<body>
    <div class="login-container">
        <h2>Signup</h2>
        <form method="post">
            <label for="username">Username:</label>
            <input type="text" name="name" id="name" required><br><br>
            <label for="email">Email:</label>
            <input type="text" name="email" id="email" required><br><br>
            <label for="password">Password:</label>

```

```
<input type="password" name="password" id="password" required><br><br>
  <input type="submit" value="Login">
</form>
</div>
</body>
</html>
```

Script.js

```
const formOpenBtn = document.querySelector("#form-open"),
  home = document.querySelector(".home"),
  formContainer = document.querySelector(".form_container"),
  formCloseBtn = document.querySelector(".form_close"),
  signupBtn = document.querySelector("#signup"),
  loginBtn = document.querySelector("#login"),
  pwShowHide = document.querySelectorAll(".pw_hide");

formOpenBtn.addEventListener("click", () => home.classList.add("show"));
formCloseBtn.addEventListener("click", () => home.classList.remove("show"));

pwShowHide.forEach((icon) => {
  icon.addEventListener("click", () => {
    let getPwInput = icon.parentElement.querySelector("input");
    if (getPwInput.type === "password") {
      getPwInput.type = "text";
      icon.classList.replace("uil-eye-slash", "uil-eye");
    } else {
      getPwInput.type = "password";
      icon.classList.replace("uil-eye", "uil-eye-slash");
    }
  });
});

signupBtn.addEventListener("click", (e) => {
  e.preventDefault();
```

```
formContainer.classList.add("active");
});
loginBtn.addEventListener("click", (e) => {
  e.preventDefault();
  formContainer.classList.remove("active");
});
```

// This is just to auto-update the data-text if you're editing it directly on the page and is not required for the actual effect

```
 $('[data-text]').on('keyup', function(){
  $(this).attr('data-text', $(this).text());
});
```

Style.css:

```
body {
  display: flex;
  align-items: center;
  justify-content: center;
  height: 100vh;
  background-image: url('/static/assets/img/car2.jpg');
  background-size: cover;
  background-repeat: no-repeat;
  background-position: center;
}
```

```
.login-container {
  width: 300px;
  padding: 20px;
  background-color: #fff;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.2);
}
```

```
h2 {
```

```
    text-align: center;  
}
```

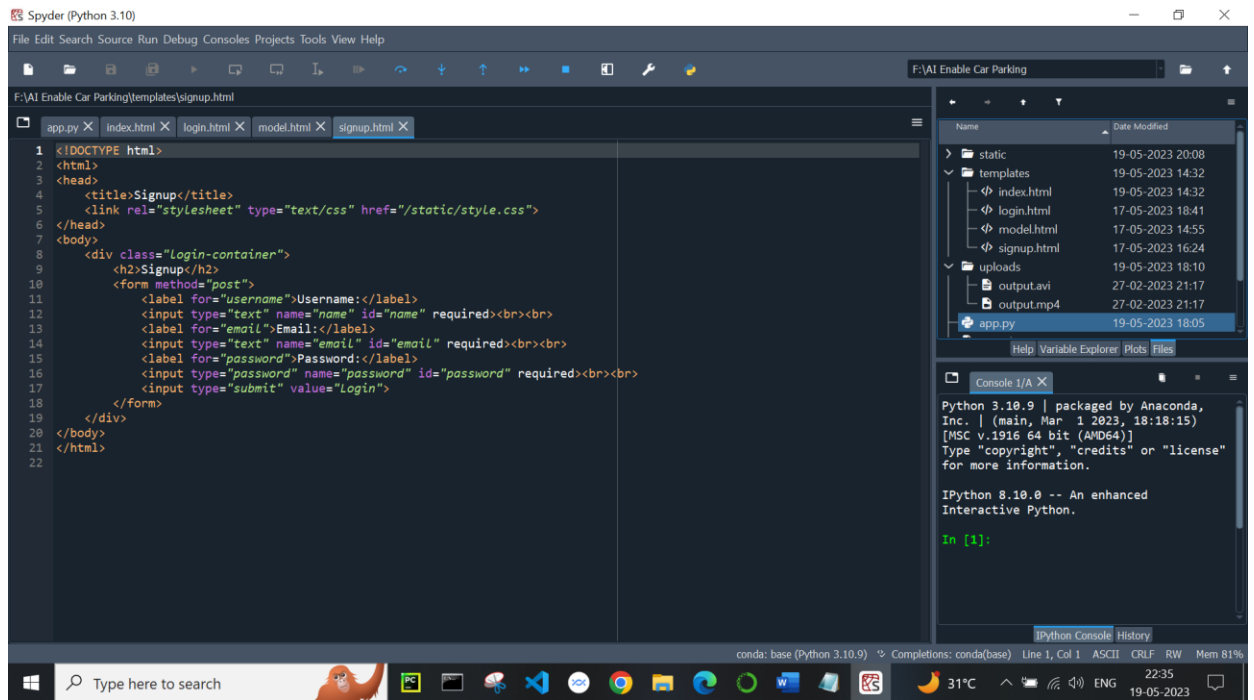
```
form {  
    margin-top: 20px;  
}
```

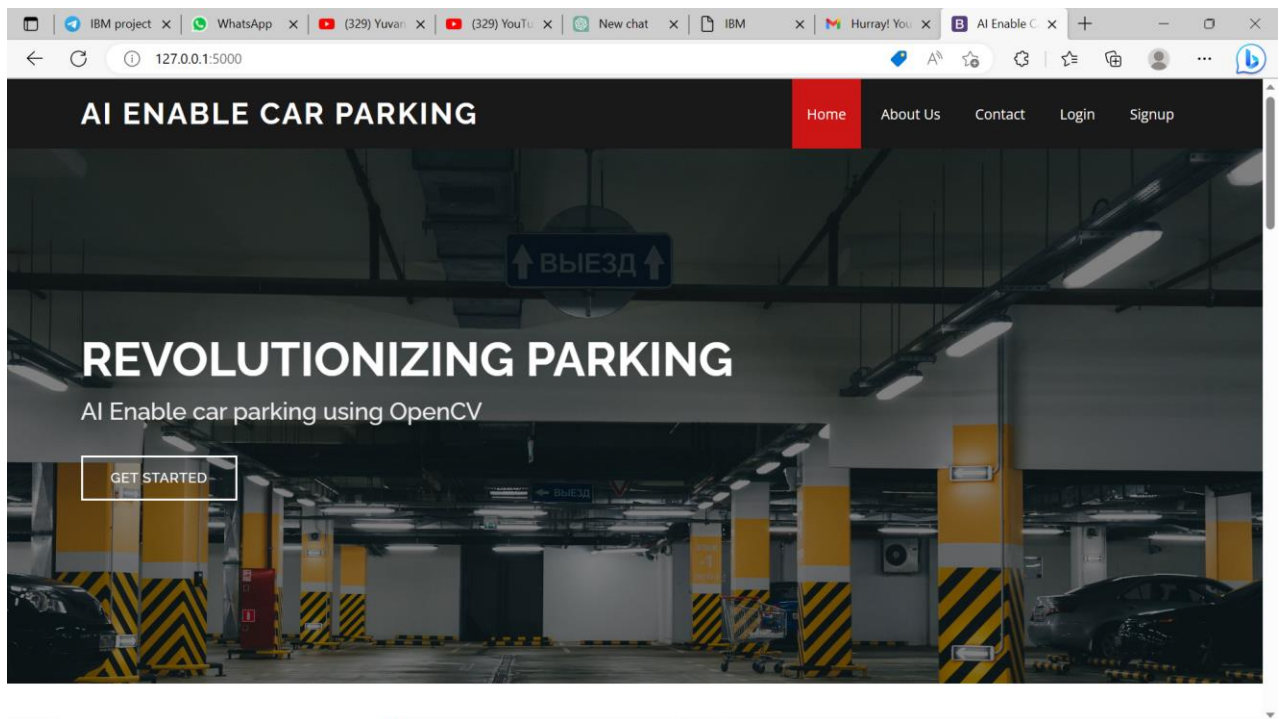
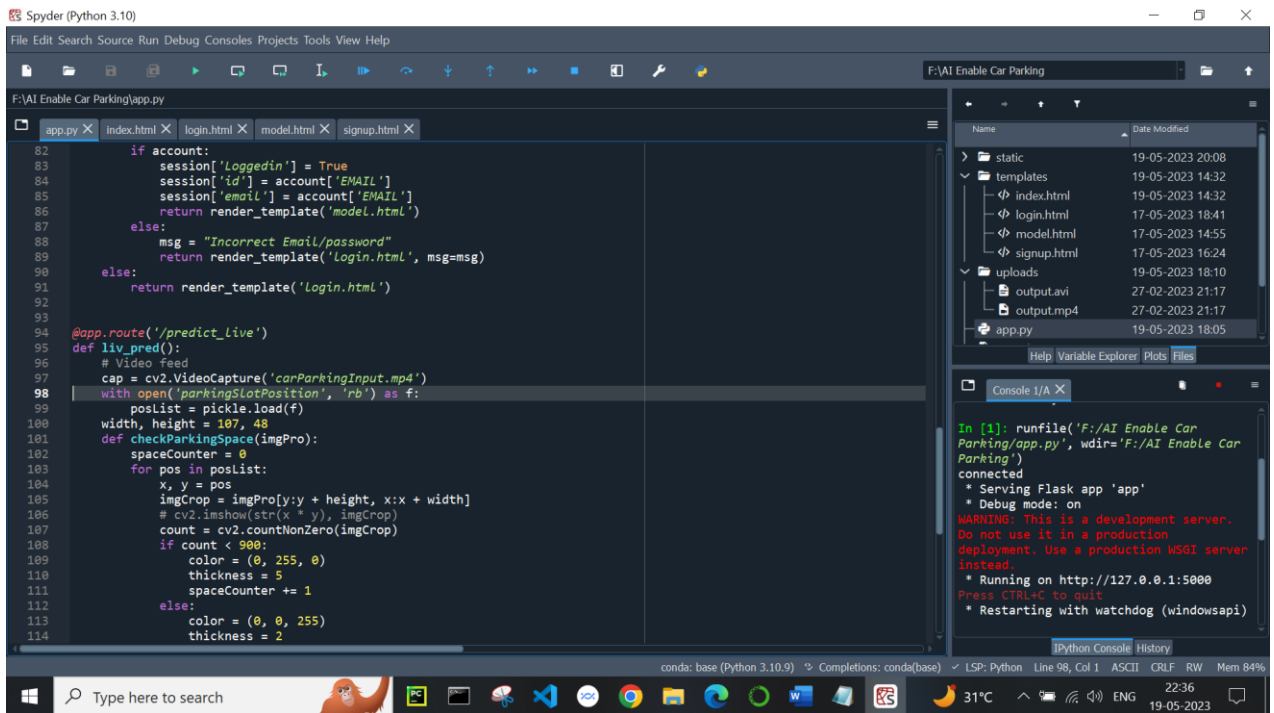
```
label {  
    display: block;  
    margin-bottom: 5px;  
}
```

```
input[type="text"],  
input[type="text"],  
input[type="password"] {  
    width: 100%;  
    padding: 5px;  
    margin-bottom: 10px;  
}
```

```
input[type="submit"] {  
    width: 100%;  
    padding: 10px;  
    background-color: #333;  
    color: #fff;  
    border: none;  
    cursor: pointer;  
}
```


Project Running Screenshots:





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.



CONTACT



AI Enable Car Parking



Email Us

ajaykumar01@gmail.com



Call Us

+91 7502522887

Send Message

AI Enable Car Parking

AJAYKUMAR.A DECE,B.TECH(IT)
ULTRA COLLEGE OF ENGINEERING
TECHNOLOGY
ULTRA NAGAR, MADURAI, IND

Phone: +91 7502522887
Email: ajaykumar01@gmail.com



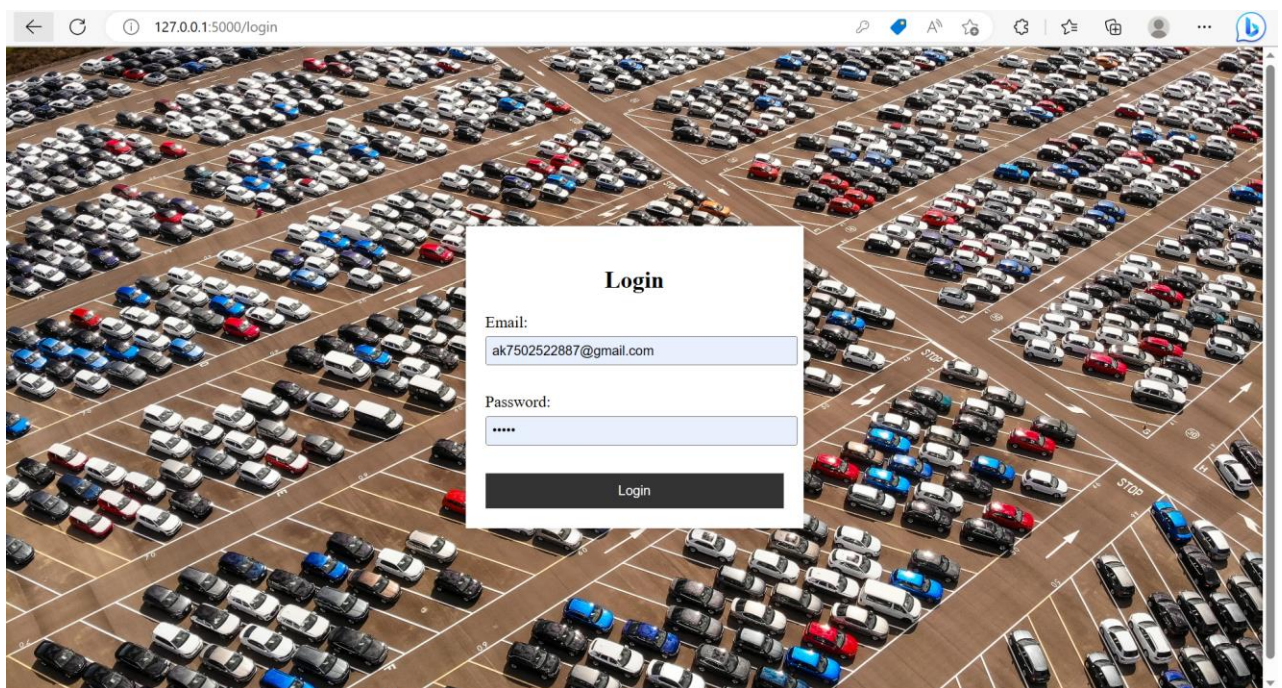
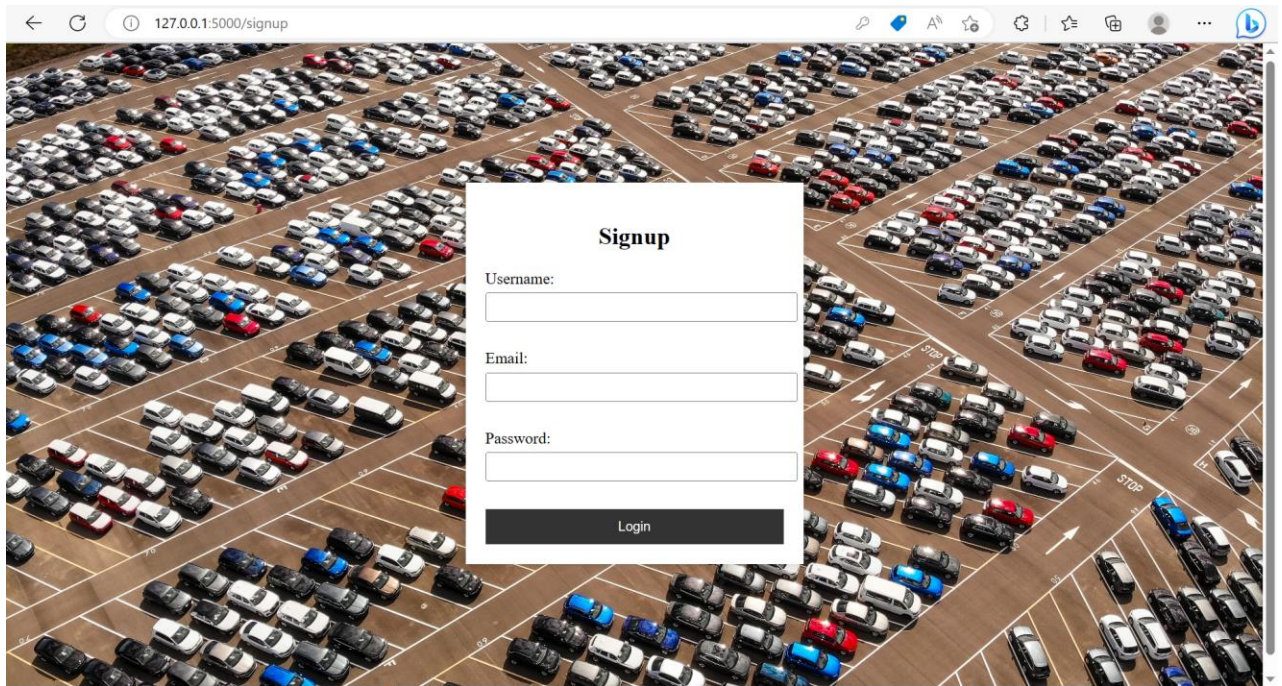
Useful Links

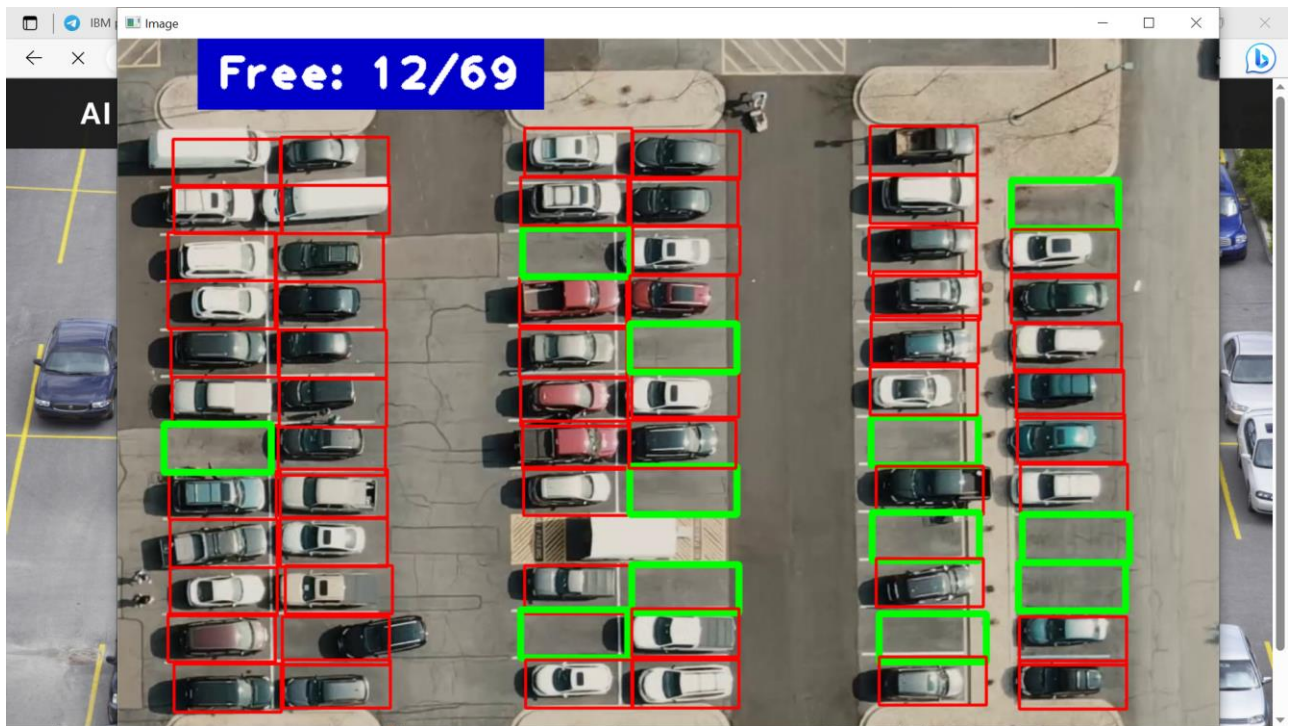
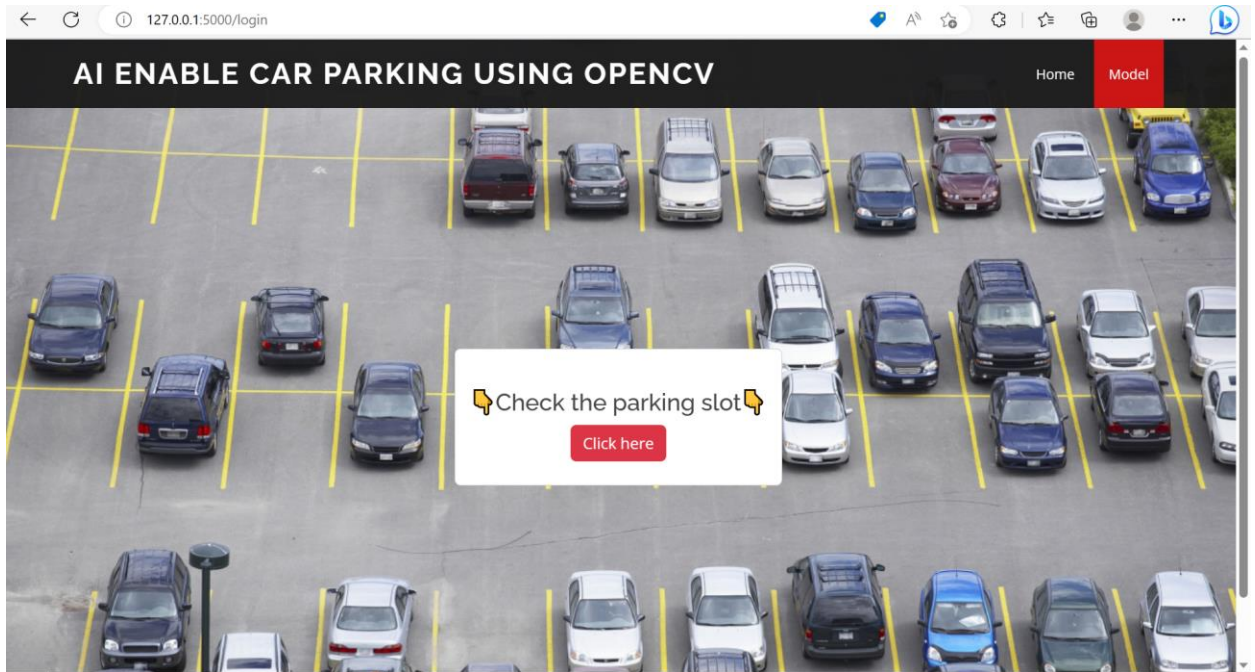
- > Home
- > About us
- > Demo
- > Contact

Our Newsletter

Subscribe







GitHub:

<https://github.com/naanmudhalvan-SI/IBM--18527-1682584903>

Project Video Demo Link:

[Click here](#)