Project Report

Date	8th November2023
Team ID	592813
Team Members	Ninad Nilesh Sugandhi 21BCE5724 Aadhith M 21BAI1192
Project Name	AI Enabled Car Parking System using OpenCV

1. INTRODUCTION

1.1 Project Overview:

The AI Enabled Car Parking System is an innovative solution leveraging OpenCV and artificial intelligence technologies to optimize parking management. By employing computer vision and machine learning algorithms, the system automates the parking process, enhances security, and improves user experience. It ensures efficient utilization of parking spaces while reducing human intervention.

1.2 Purpose:

The purpose of the AI Enabled Car Parking System is to revolutionize conventional parking management. It aims to address challenges related to congestion, security, and user convenience in parking lots. By integrating advanced technologies, the system simplifies parking for users, optimizes space allocation, and provides real-time insights for efficient parking lot management.

2. LITERATURE SURVEY

2.1 Existing problem:

Traditional parking systems often suffer from inefficiencies such as inadequate space utilization, manual errors in parking management, and lack of real-time data for decision-making. These challenges lead to congestion, frustration among users, and inefficient use of available parking space.

2.2 References:

Build a simple smart parking project using python and OpenCV | by Razvan Vilceanu | The Startup | Medium

AI-Enabled-Car-Parking-System-Using-OpenCV/ParkingSpacePicker.py at main · rahul13289/AI-Enabled-Car-Parking-System-Using-OpenCV · GitHub

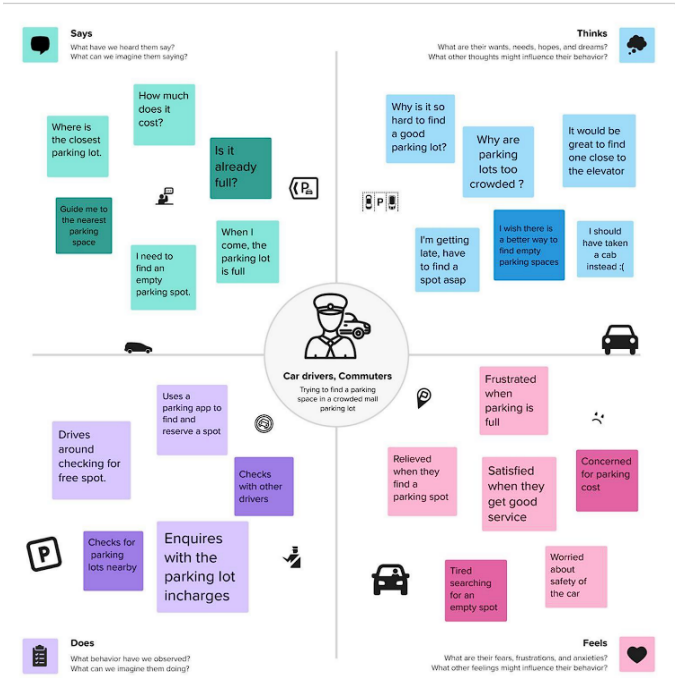
2.3 Problem Statement Definition:

In urban and commercial areas, the scarcity of parking spaces often leads to significant inconveniences for drivers and challenges for parking facility operators. The lack of an efficient system for managing parking spaces results in frustrated drivers circling parking lots in search of available spots, leading to wasted time, increased traffic congestion, and elevated stress levels. Additionally, parking facility owners face difficulties in optimizing space utilization, managing peak hours, and ensuring a positive customer experience.

To address these challenges, there is a pressing need for an AI-enabled car parking system utilizing OpenCV technology. This system must be capable of real-time monitoring and analysis of parking spaces, accurately detecting available spots. By leveraging the power of AI and OpenCV, this solution aims to revolutionize the parking experience, alleviating the frustrations of drivers and improving the overall management of parking facilities.

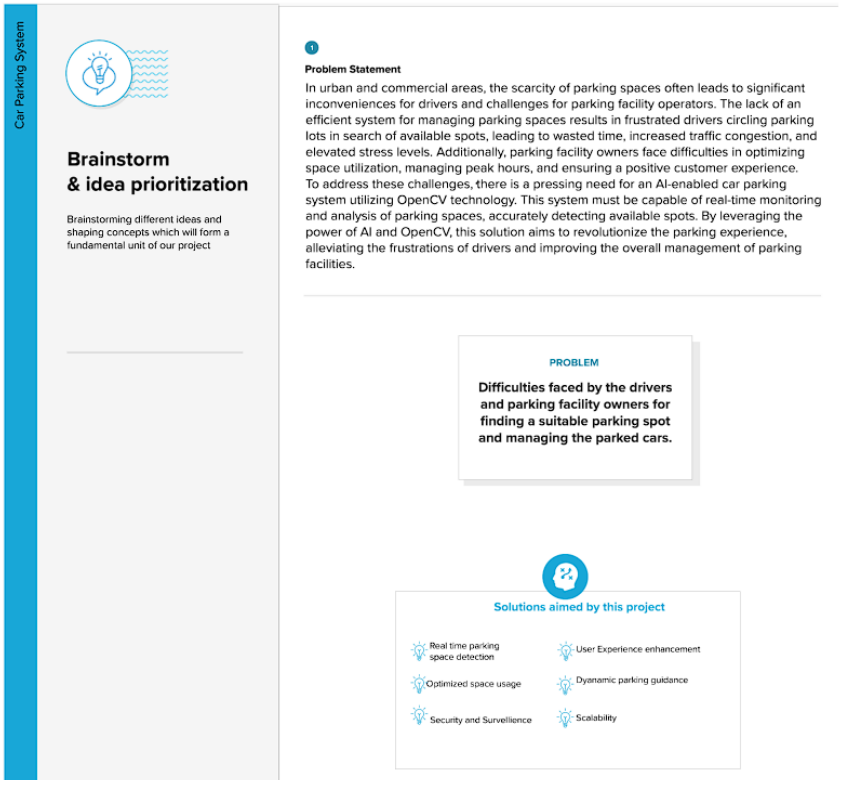
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

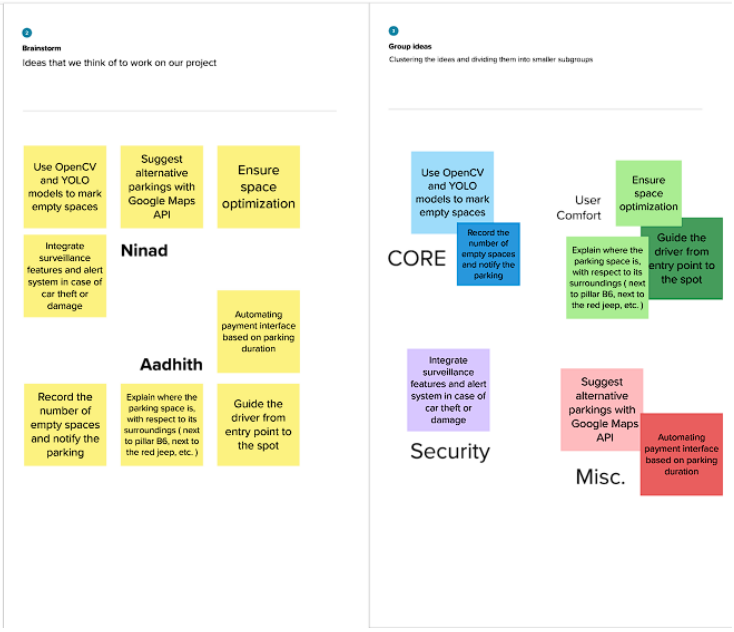


3.2 Ideation & Brainstorming

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



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritisation

4. REQUIREMENT ANALYSIS

4.1. Functional requirement

4.1.1. Automated Parking Space Allocation:

The system should automatically allocate parking spaces to vehicles entering the parking lot based on their size, type, and availability, ensuring optimal space utilization.

4.1.2. Real-time Parking Occupancy Monitoring:

The system must provide real-time updates on parking space occupancy to both the parking management staff and users. Users should be able to check the availability of parking spaces through a mobile app or a web interface.

4.1.3. User Registration and Authentication:

Users should be able to register their vehicles and create accounts in the system. Secure authentication mechanisms, such as two-factor authentication, should be implemented to ensure user identity verification.

4.2 Non-Functional requirements

4.2.1. Performance:

The system should be capable of handling a high volume of simultaneous users and vehicles without experiencing performance degradation. Response times for user interactions and data processing should be minimal.

4.2.2. Scalability:

The system architecture should be scalable, allowing for the addition of more parking spaces, users, and devices without requiring significant modifications to the existing infrastructure.

4.2.3. Reliability:

The system should be highly reliable, ensuring continuous operation 24/7. It should have backup and failover mechanisms to handle hardware failures or network issues without disrupting services.

4.2.4. Security:

The system should implement robust security measures, including data encryption, secure communication protocols, and access controls, to protect user data, payment information, and system integrity.

4.2.5. Usability:

The user interfaces (mobile app, website, payment kiosks) should be intuitive and user-friendly. Users should be able to navigate through the system easily, make reservations, and complete payments without confusion.

4.2.6. Compliance:

The system should comply with local regulations and standards related to parking management, data privacy, and security. It should also adhere to industry best practices and guidelines.

4.2.7. Maintainability:

The system should be easy to maintain and update. Regular maintenance tasks, software updates, and security patches should be applied without causing significant downtime or disruptions to the service.

4.2.8. Availability:

The system should have a high level of availability, ensuring that users can access the parking services at any time. Scheduled maintenance activities should be communicated to users in advance to minimize inconvenience.

4.2.9. Integration:

The system should be able to integrate with third-party services, such as payment gateways, mapping services, and emergency services communication systems, to enhance its functionality and user experience.

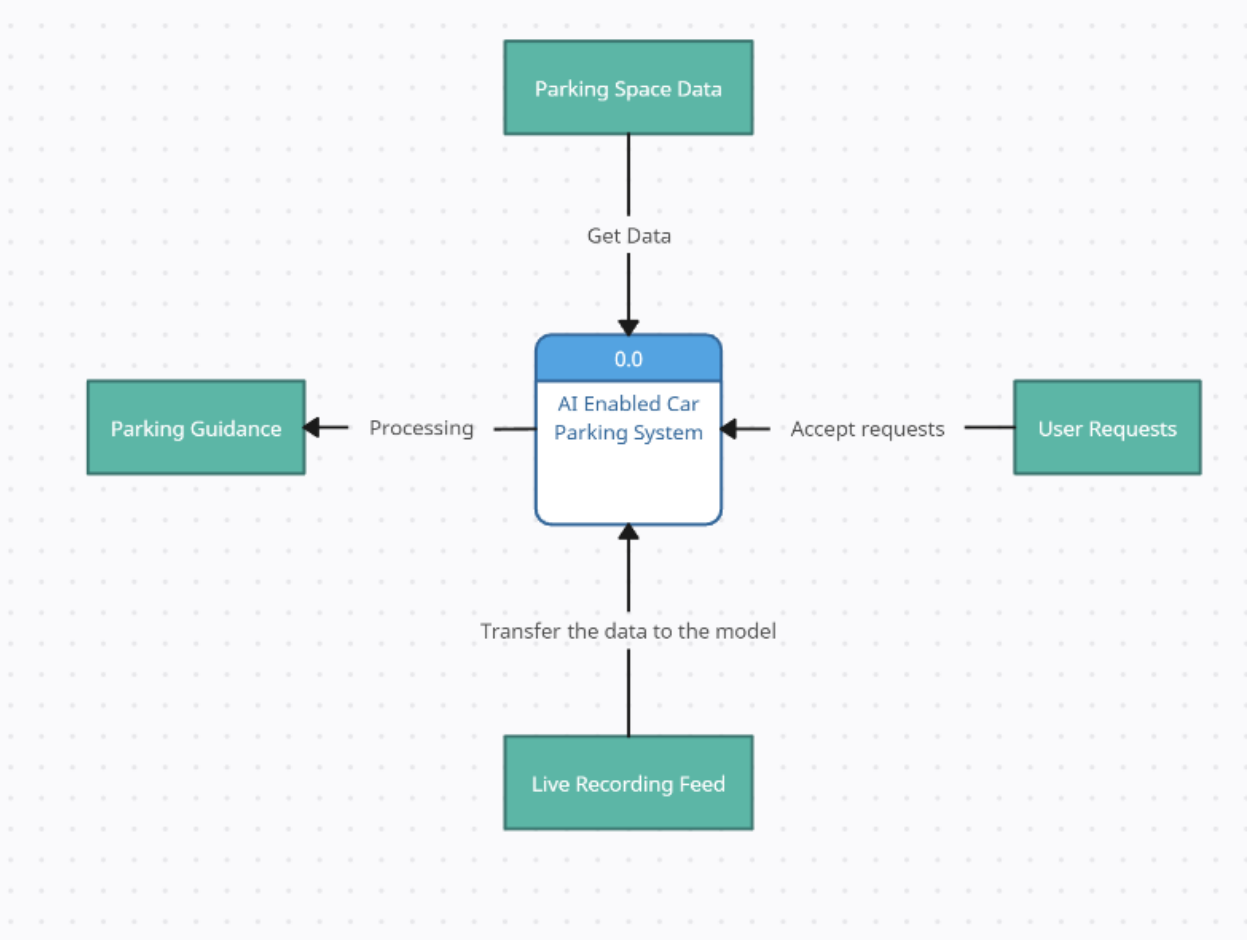
5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories

LEVEL 0 DFD:

A Level 0 Data Flow Diagram (DFD) is a visual representation of a system that provides a fundamental overview of the flow of data within the system.

The Level 0 DFD is given below:



Description:

The DFD has three external entities:

1. User Requests: This entity represents the requests that users make to the parking system, such as requests to find a parking spot or to reserve a parking spot.

2. Live Recording Feed: This entity represents the live video feed from the parking lot. This feed can be used to monitor the parking lot and to identify available parking spots.

3. Parking Guidance: This entity represents the guidance that the parking system provides to users, such as directions to the parking lot or directions to an available parking spot.

The DFD also has three internal processes:

1. Accept requests: This process accepts the requests from the User Requests entity.

2. Transfer the data to the model: This process transfers the data from the User Requests entity and the Live Recording Feed entity to the parking space model.

3. Processing: This process processes the data in the parking space model to generate the Parking Guidance entity.

The DFD also has two data stores:

1. Parking Space Data: This data store stores information about the parking spaces in the parking lot, such as the location of each parking space and whether or not it is available.

2. Parking Space Model: This data store stores the parking space model, which is a representation of the parking lot that is used to generate the Parking Guidance entity.

The following is a step-by-step explanation of how the DFD works:

1. The user makes a request to the parking system, such as a request to find a parking spot.

2. The “Accept requests” process accepts the request from the user.

3. The “Transfer the data to the model” process transfers the data from the User Requests entity and the Live Recording Feed entity to the parking space model.

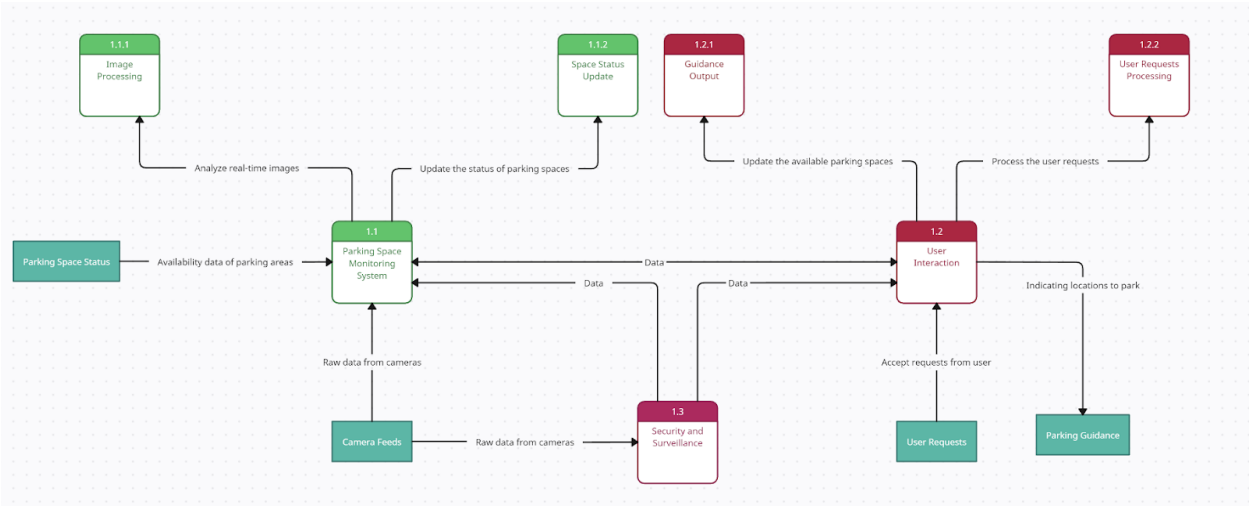
4. The “Processing” process processes the data in the parking space model to generate the Parking Guidance entity.

5. The Parking Guidance entity is provided to the user.

LEVEL 1 DFD:

At Level 1, the DFD delves into each major process from Level 0, breaking them down into sub-processes and data flows.

The Level-1 DFD is given below:



Description:

The DFD has three external entities:

1. Raw data from cameras: This entity represents the raw video feed from the cameras in the parking lot.

2. User requests: This entity represents the requests that users make to the parking system, such as requests to find a parking spot or to reserve a parking spot.

3. Guidance: This entity represents the guidance that the parking system provides to users, such as directions to the parking lot or directions to an available parking spot.

The DFD also has three internal processes:

1. Analyse real-time images: This process analyses the raw video feed from the cameras to identify available parking spots and to track the movement of vehicles in the parking lot.

2. Update the status of parking spaces: This process updates the status of the parking spaces in the parking system based on the information from the Analyze real-time images process.

3. Process user requests: This process processes the requests from the User requests entity and generates the Guidance entity.

The DFD also has one data store:

Parking space availability: This data store stores information about the availability of the parking spaces in the parking lot.

The following is a step-by-step explanation of how the DFD works:

1. The “Analyse real-time” images process analyses the raw video feed from the cameras to identify available parking spots and to track the movement of vehicles in the parking lot.

2. The “Update the status of parking spaces” process updates the status of the parking spaces in the parking system based on the information from the Analyze real-time images process.

3. The “Process user requests” process processes the requests from the User requests entity and generates the Guidance entity.

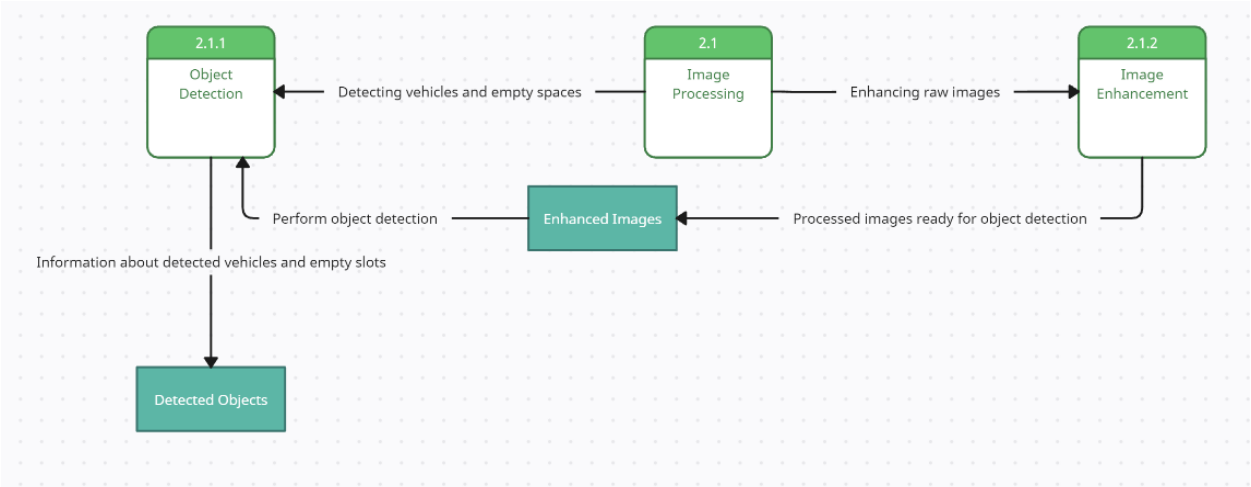
4. The Guidance entity is provided to the user.

LEVEL 2 DFD:

At Level 2, the DFD further breaks down Level 1 processes into detailed sub-processes, providing a granular view of the system's operations.

The Level-2 DFDs are given below:

2.1. Image Processing Sub-Process:



Description:

The DFD has three external entities:

1. Input image: This entity represents the image that is input to the system.

2. Output image: This entity represents the image that is output from the system.

3. Detected objects: This entity represents the objects that have been detected in the image.

The DFD also has three internal processes:

1. Enhance image: This process enhances the input image to improve the accuracy of the object detection process.

2. Detects objects: This process detects the objects in the input image.

3. Label objects: This process labels the detected objects in the image.

The DFD also has one data store:

Object database: This data store contains information about the objects that the system can detect.

The following is a step-by-step explanation of how the DFD works:

1. The “Enhance image” process enhances the input image to improve the accuracy of the object detection process.

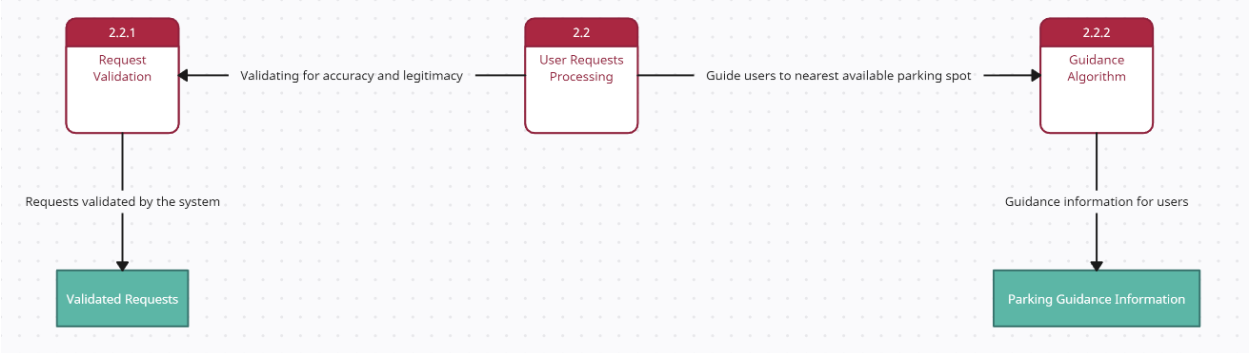
2. The “Detect objects” process detects the objects in the input image.

3. The “Label objects” process labels the detected objects in the image.

4. The “Output image” entity is output from the system.

5. The “Detected objects” entity is output from the system.

2.2. User Requests Processing Sub-Process:



Description:

The DFD has three external entities:

1. User requests: This entity represents the requests that users make to the parking system, such as requests to find a parking spot or to reserve a parking spot.

2. Parking lot data: This entity represents the data about the parking lot, such as the location of the parking lot, the number of parking spaces in the parking lot, and the availability of parking spaces in the parking lot.

3. Guidance: This entity represents the guidance that the parking system provides to users, such as directions to the parking lot or directions to an available parking spot.

The DFD also has three internal processes:

1. Process user requests: This process processes the requests from the User requests entity and generates a query to the Parking lot data entity.

2. Query parking lot data: This process queries the Parking lot data entity to determine the availability of parking spaces.

3. Generate guidance: This process generates guidance for the user based on the results of the Query parking lot data process.

The DFD also has one data store:

Parking lot data: This data store stores information about the parking lot, such as the location of the parking lot, the number of parking spaces in the parking lot, and the availability of parking spaces in the parking lot.

The following is a step-by-step explanation of how the DFD works:

1. The user makes a request to the parking system, such as a request to find a parking spot.

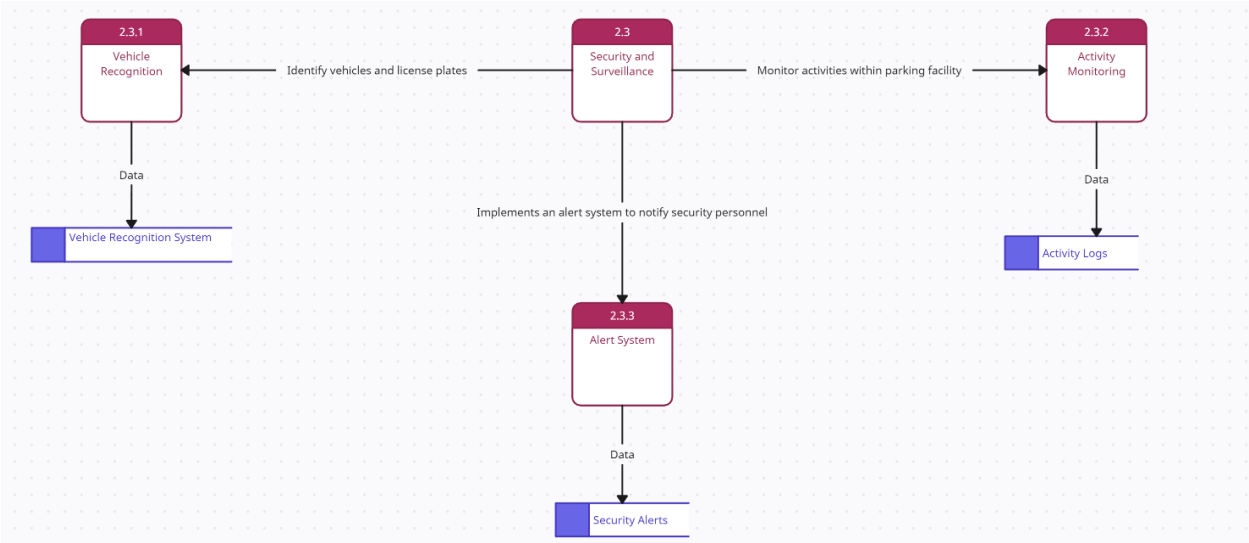
2. The “Process user requests” process processes the request from the user and generates a query to the Parking lot data entity.

3. The “Query parking lot data” process queries the Parking lot data entity to determine the availability of parking spaces.

4. The “Generate guidance” process generates guidance for the user based on the results of the Query parking lot data process.

5. The Guidance entity is provided to the user.

2.3. Security and Surveillance Sub-Process:



The DFD has three external entities:

1. Raw data from cameras: This entity represents the raw video feed from the cameras in the parking lot.

2. User requests: This entity represents the requests that users make to the parking system, such as requests to find a parking spot or to reserve a parking spot.

3. Guidance: This entity represents the guidance that the parking system provides to users, such as directions to the parking lot or directions to an available parking spot.

The DFD also has three internal processes:

1. Analyse real-time images: This process analyses the raw video feed from the cameras to identify available parking spots and to track the movement of vehicles in the parking lot.

2. Update the status of parking spaces: This process updates the status of the parking spaces in the parking system based on the information from the Analyze real-time images process.

3. Process user requests: This process processes the requests from the User requests entity and generates the Guidance entity.

The DFD also has one data store:

Parking space availability: This data store stores information about the availability of the parking spaces in the parking lot.

The following is a step-by-step explanation of how the DFD works:

1. The “Analyse real-time images” process analyses the raw video feed from the cameras to identify available parking spots and to track the movement of vehicles in the parking lot.

2. The “Update the status of parking spaces” process updates the status of the parking spaces in the parking system based on the information from the Analyze real-time images process.

3. The "Process user requests” process processes the requests from the User requests entity and generates the Guidance entity.

4. The Guidance entity is provided to the user.

User Stories

The below given table describes the user stories for the product

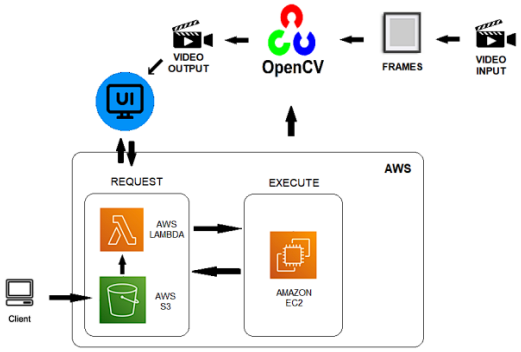
User	Functional	User	User Story/ Task	Acceptance Criteria	Priority	Release

Type	Requirements (Epic)	Story Number				
New User	Registration	USN-1	As a new user, I want to create an account on the mobile app to access parking services.	The registration process should include fields for name, email, phone number, and password. Upon successful registration, the user should receive a confirmation email.	High	Sprint-1.0
Registered User	Login	USN-2	As a registered user, I want to log into the mobile app to access parking information and services.	Users should be able to log in using their email and password. The system should validate the credentials and grant access upon successful login.	High	Sprint-1.0
Driver	Parking Space Availability	USN-3	As a driver, I want to easily find an available parking space so that I can park my car without hassle.	The system should provide real-time updates on available parking spaces. The driver should receive clear and accurate guidance to the selected parking space.	High	Sprint-1.0
	Reservation System	USN-4	As a driver, I want to reserve a parking space in advance so that I can ensure a spot during peak hours.	The system should allow users to reserve parking spaces through the mobile app. It should confirm the reservation and provide a QR code for entry. If the user doesn't arrive within the specified time, the reservation should be cancelled and the space made available for others.	Medium	Sprint-1.2
	Profile Management	USN-5	As a driver, I want to manage my profile information such as name, contact details, and vehicle information.	Users should be able to edit their profile details. Changes made should be reflected in the system immediately.	Medium	Sprint-1.1

	Payment Management	USN-6	As a driver, I want to manage my payment methods for parking services.	Users should be able to add, edit, or remove payment methods. The system should securely store payment information and allow users to select a default payment method.	Medium	Sprint-1.1
	Parking History	USN-7	As a driver, I want to view my parking history, including previous parking sessions and payments.	Users should have access to a detailed parking history section displaying past sessions, payment details, and duration of each session.	Medium	Sprint-1.2
	Favourites & Reminders	USN-8	As a driver, I want to mark certain parking facilities as favourites and set reminders for specific parking sessions.	Users should be able to mark preferred parking facilities as favourites for quick access. The system should allow setting reminders for upcoming parking sessions, sending notifications to users.	Low	Sprint-1.2
Parking Facility Operator	Space Utilisation Optimization	USN-9	As a parking facility operator, I want to optimise parking space allocation to accommodate maximum vehicles and enhance revenue.	The system should analyse parking data to suggest optimal space allocations. It should provide reports indicating the optimised space utilisation and revenue increase.	High	Sprint-1.1
	Real-time Occupancy Dashboard	USN-10	As a parking facility operator, I want a real-time dashboard displaying occupancy and availability data.	The system should provide a live dashboard showing the current occupancy status, including the number of occupied and available parking spaces. The dashboard should update in real-time.	High	Sprint-1.0
	User Feedback Analysis	USN-11	As a parking facility operator, I want to collect and analyse user feedback to improve services.	The system should allow users to provide feedback within the app. Feedback data should be collected and analysed for insights. Regular reports should be generated based on feedback analysis.	Medium	Sprint-1.1

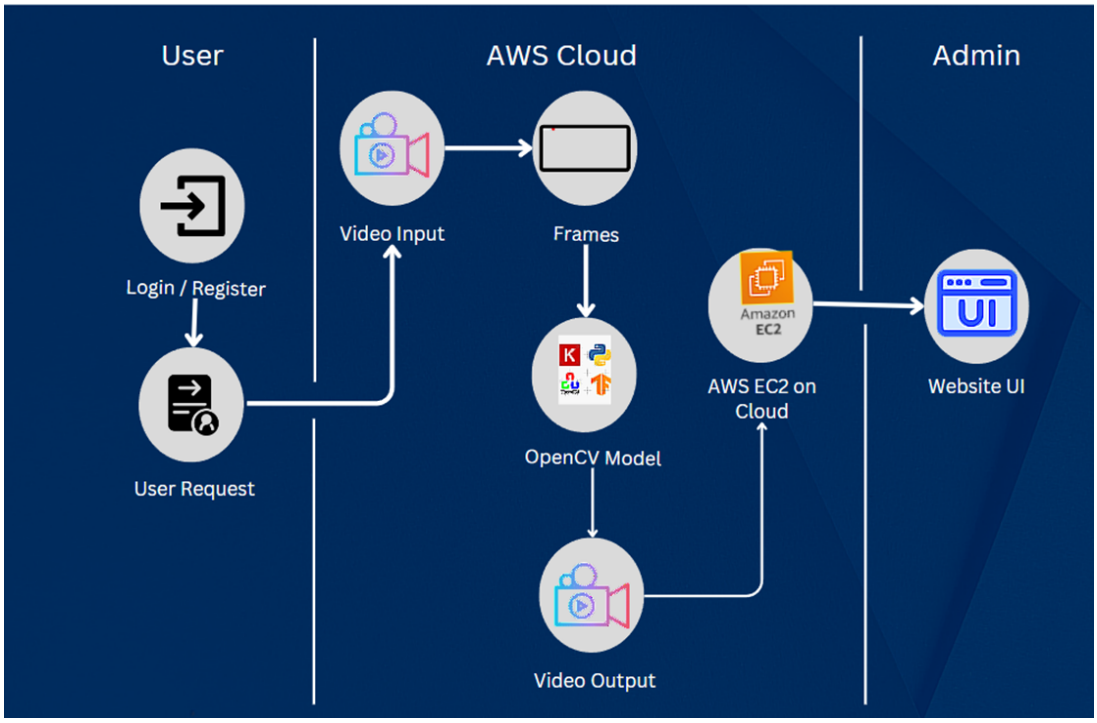
Security Personnel	Suspicious Activity Detection	USN-12	As a security personnel, I want to be alerted about any suspicious activities within the parking facility to ensure a secure environment.	The system should monitor parking activities and send real-time alerts in case of unauthorised access or suspicious behaviour.	High	Sprint-1.1
Application Administrator	Data Analytics and Insights	USN-13	As an application administrator, I want to access detailed analytics and insights to understand user behaviour and improve system performance.	The system should provide comprehensive analytics reports, including user interaction patterns, popular parking times, and frequently used features. It should allow exporting these reports in various formats for further analysis.	Medium	Sprint-1.1
	System Monitoring	USN-14	As an administrator, I want to monitor the system's performance, including server health and user activity.	The system should provide logs and reports on server health, user interactions, and error occurrences. Administrators should receive notifications for critical issues.	High	Sprint-1.0
UI/UX Designer	UI Enhancements	USN-15	As a UI/UX designer, I want to enhance the user interface with intuitive design elements and seamless navigation.	The app should undergo design improvements, including user-friendly menus, intuitive icons, and smooth transitions between screens. Users should find the app visually appealing and easy to use.	High	Sprint-1.1

5.2 Solution Architecture:



6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture:



6.2 Sprint Planning & Estimation:

Sprint	Functional Requirements (Epic)	User Story Number	User Story/ Task	Story Points	Priority	Team Members
Sprint-1.0	Registration	USN-1	As a new user, I want to create an account on the mobile app to access parking services.	10	High	2
Sprint-1.0	Login	USN-2	As a registered user, I want to log into the mobile app to access parking information and services.	10	High	2
Sprint-1.0	Parking Space Availability	USN-3	As a driver, I want to easily find an available parking space so that I can park my car without hassle.	15	High	2

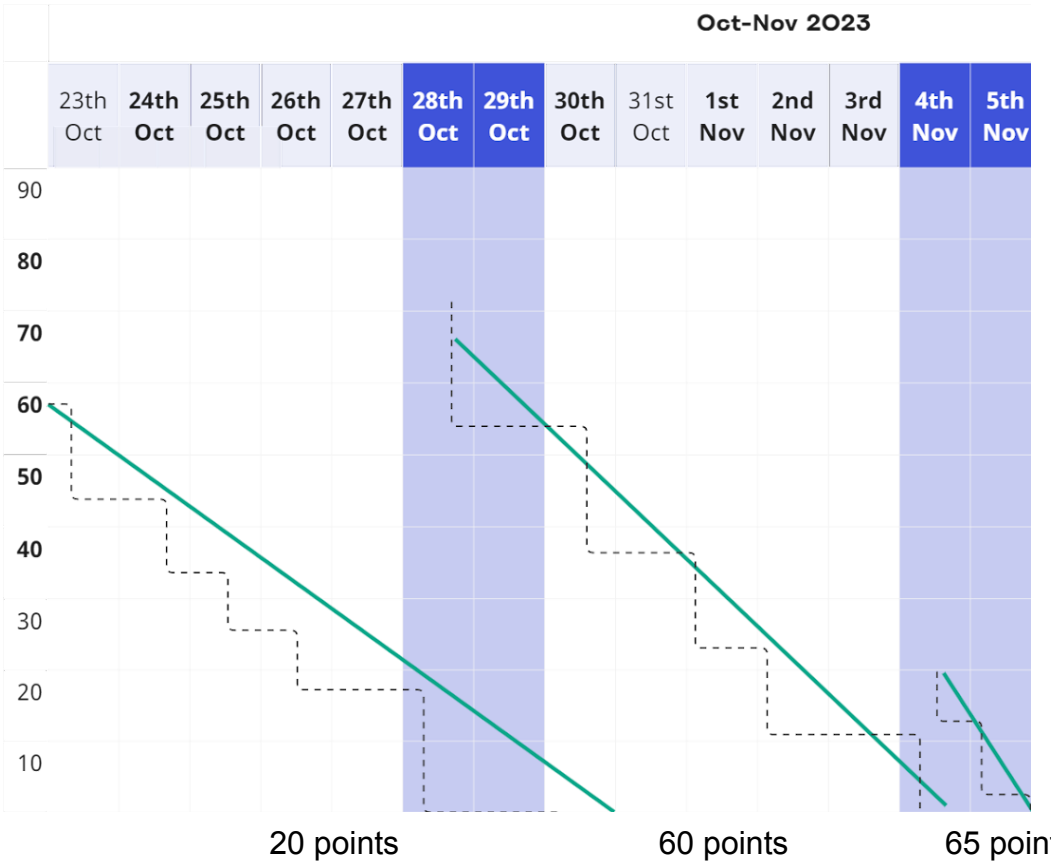
Sprint-1.1	Profile Management	USN-4	As a driver, I want to manage my profile information such as name, contact details, and vehicle information.	10	Medium	2
Sprint-1.1	Space Utilisation Optimization	USN-5	As a parking facility operator, I want to optimise parking space allocation to accommodate maximum vehicles and enhance revenue.	10	High	2
Sprint-1.1	Suspicious Activity Detection	USN-6	As a security personnel, I want to be alerted about any suspicious activities within the parking facility to ensure a secure environment.	10	High	2
Sprint-1.1	Data Analytics and Insights	USN-7	As an application administrator, I want to access detailed analytics and insights to understand user behaviour and improve system performance.	5	Medium	2
Sprint-1.1	Payment Management	USN-8	As a driver, I want to manage my payment methods for parking services.	10	Medium	2
Sprint-1.2	Reservation System	USN-9	As a driver, I want to reserve a parking space in advance so that I can ensure a spot during peak hours.	5	Medium	2
Sprint-1.2	Parking History	USN-10	As a driver, I want to view my parking history, including previous parking sessions and payments.	10	Medium	2
Sprint-1.2	Favourites & Reminders	USN-11	As a driver, I want to mark certain parking facilities as favourites and set reminders for specific parking sessions.	5	Low	2

Sprint-1.0	Real-time Occupancy Dashboard	USN-12	As a parking facility operator, I want a real-time dashboard displaying occupancy and availability data.	15	High	2
Sprint-1.1	User Feedback Analysis	USN-13	As a parking facility operator, I want to collect and analyse user feedback to improve services.	10	Medium	2
Sprint-1.0	System Monitoring	USN-14	As an administrator, I want to monitor the system's performance, including server health and user activity.	10	High	2
Sprint-1.1	UI Enhancements	USN-15	As a UI/UX designer, I want to enhance the user interface with intuitive design elements and seamless navigation.	10	High	2

Project Tracker:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
1.0	60	7 Days	23rd Oct 2023	30th Oct 2023	60	30th Oct 2023
1.1	65	7 Days	28th Oct 2023	4th Nov 2023	60	4th Nov 2023
1.2	20	3 Days	3rd Nov 2023	5th Nov 2023	20	6th Nov 2023

6.3 Sprint Delivery Schedule:



20 points 60 points 65 points

7. CODING & SOLUTIONING

7.1. Feature 1 - User Authentication:

1. Users can register and create accounts using their email and password.

2. Secure authentication mechanisms are implemented for user identity verification.

7.2. Feature 2 - User Interface:

1. The application provides intuitive and user-friendly interfaces for login, registration, and model predictions.

2. Redirects users to appropriate pages based on their actions (e.g., successful login redirects to the model page).

7.3. Feature 3 - Database Interaction:

1. The application interacts with a MySQL database hosted on Amazon RDS for user data storage.

2. Supports user registration, login, and validation of user credentials.

7.4. Feature 4 - Real-time Parking Space Detection:

1. Utilizes OpenCV for real-time processing of video input from a specified file ("carParkingInput.mp4").

2. Detects and highlights free parking spaces within the video frames using image processing techniques.

7.5. Database Schema:

Column Name	Data Type	Constraints
ID (Primary Key)	INT	Auto-increment
NAME	VARCHAR(255)	Not Null
EMAIL	VARCHAR(255)	Not Null, Unique
PASSWORD	VARCHAR(255)	Not Null

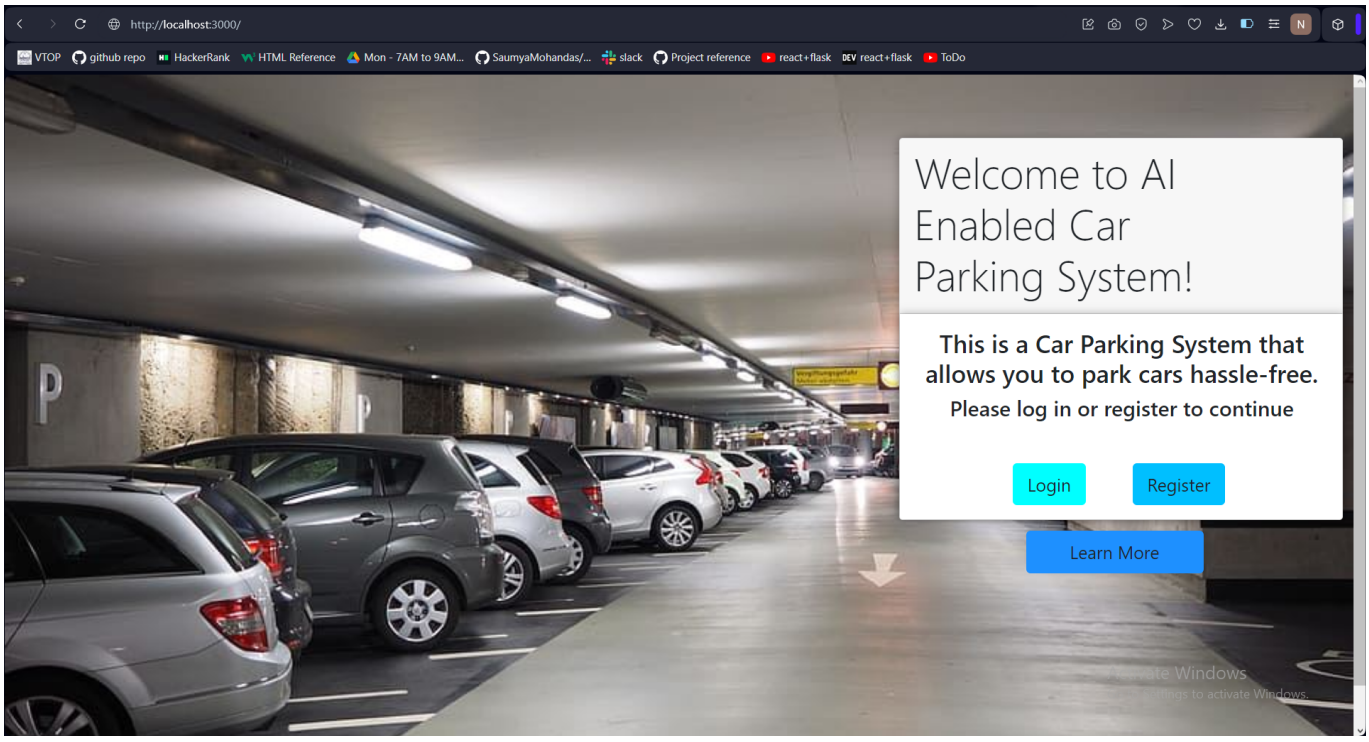
Table Name: REGISTER

ID: Unique identifier for each user (auto-generated).
NAME: User's full name.
EMAIL: User's email address (unique for each user, used for login).
PASSWORD: User's password (hashed and stored securely).
This database schema allows for the storage of user information, enabling user registration, login, and secure authentication within the AI-enabled car parking system.

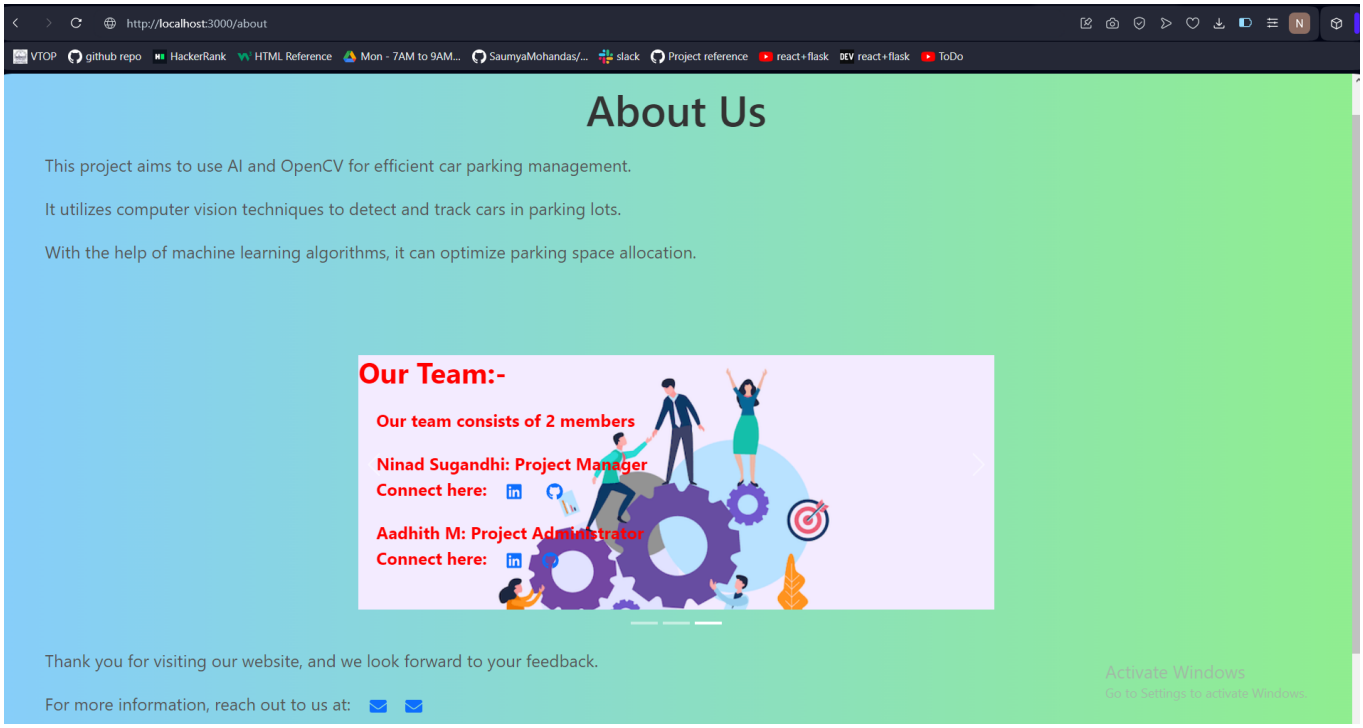
9. RESULTS:

Output Screenshots -

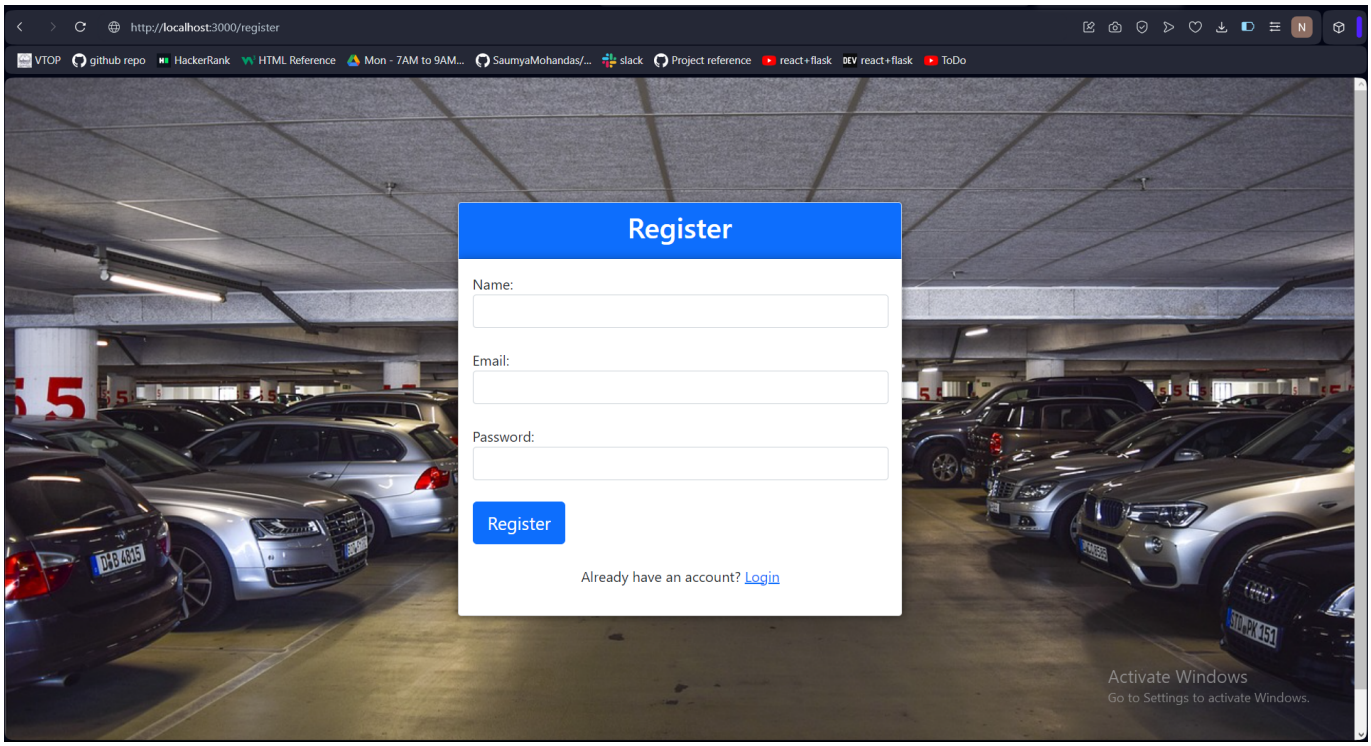
9.1. Home Page:



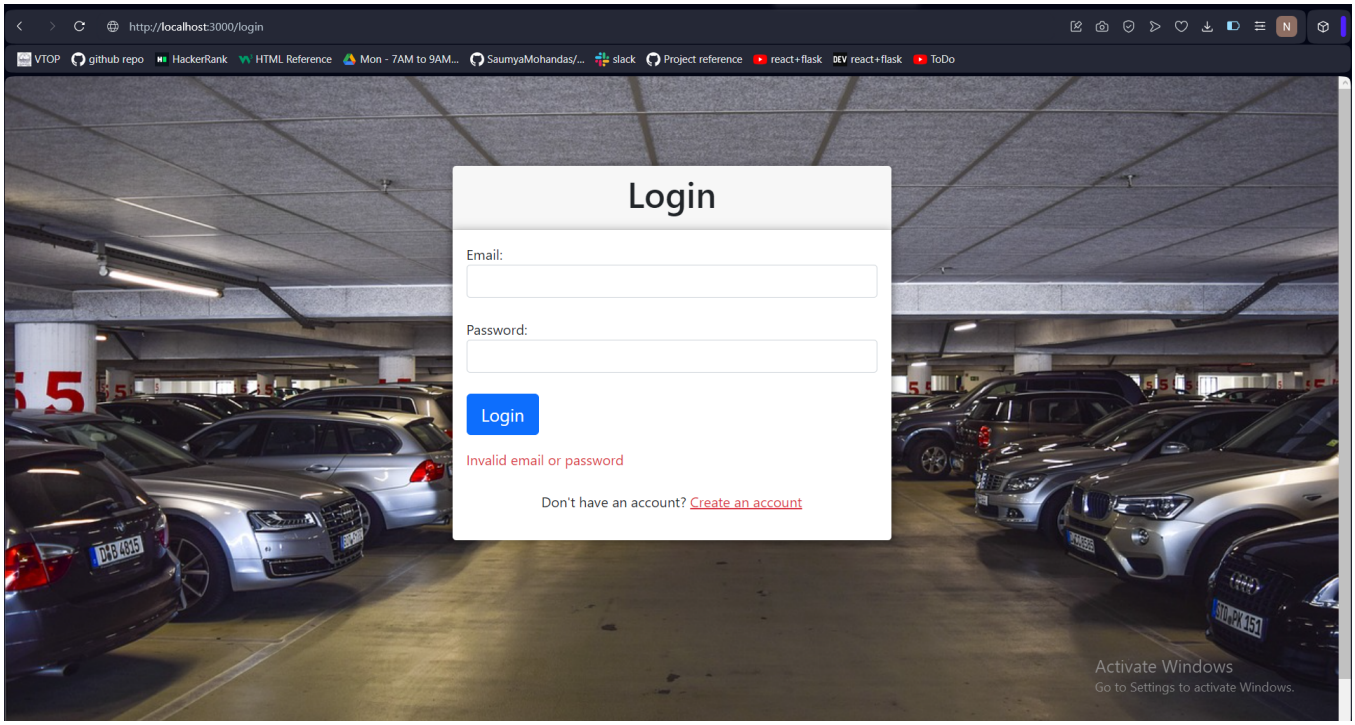
9.2.About Us Page:



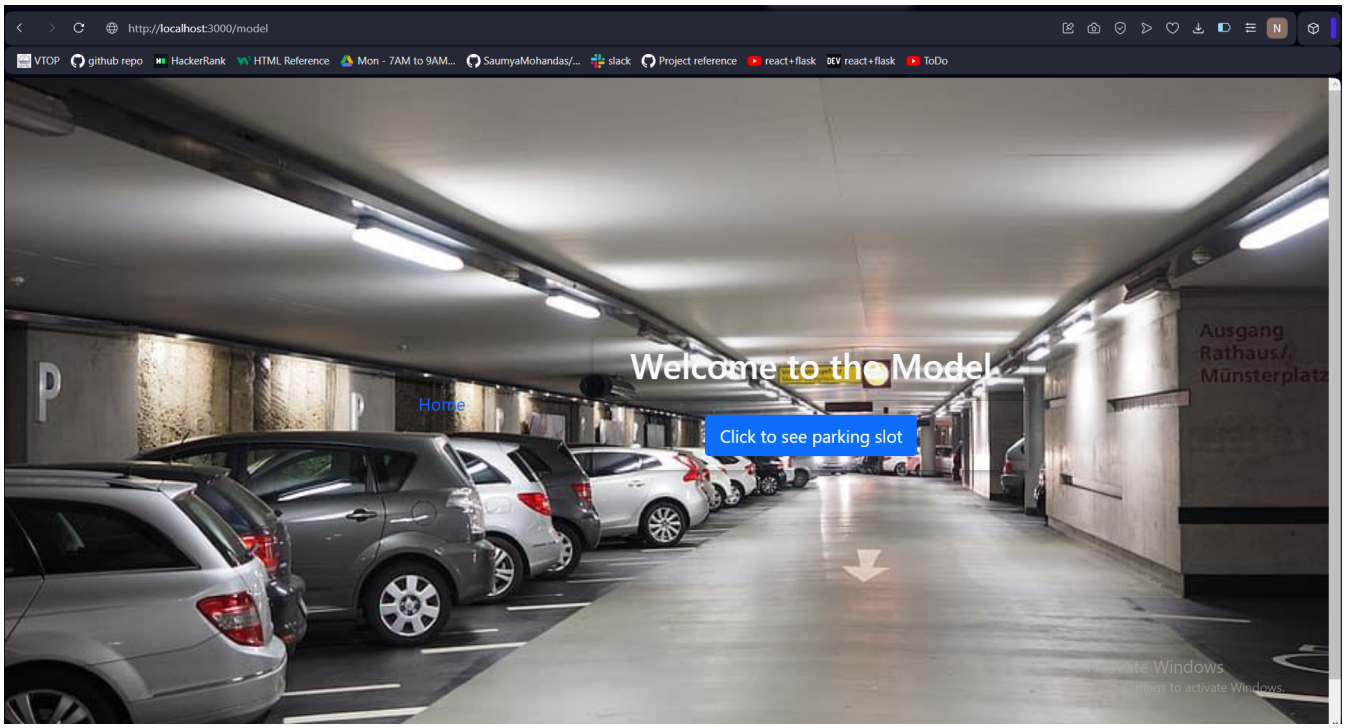
9.3. Register Page:



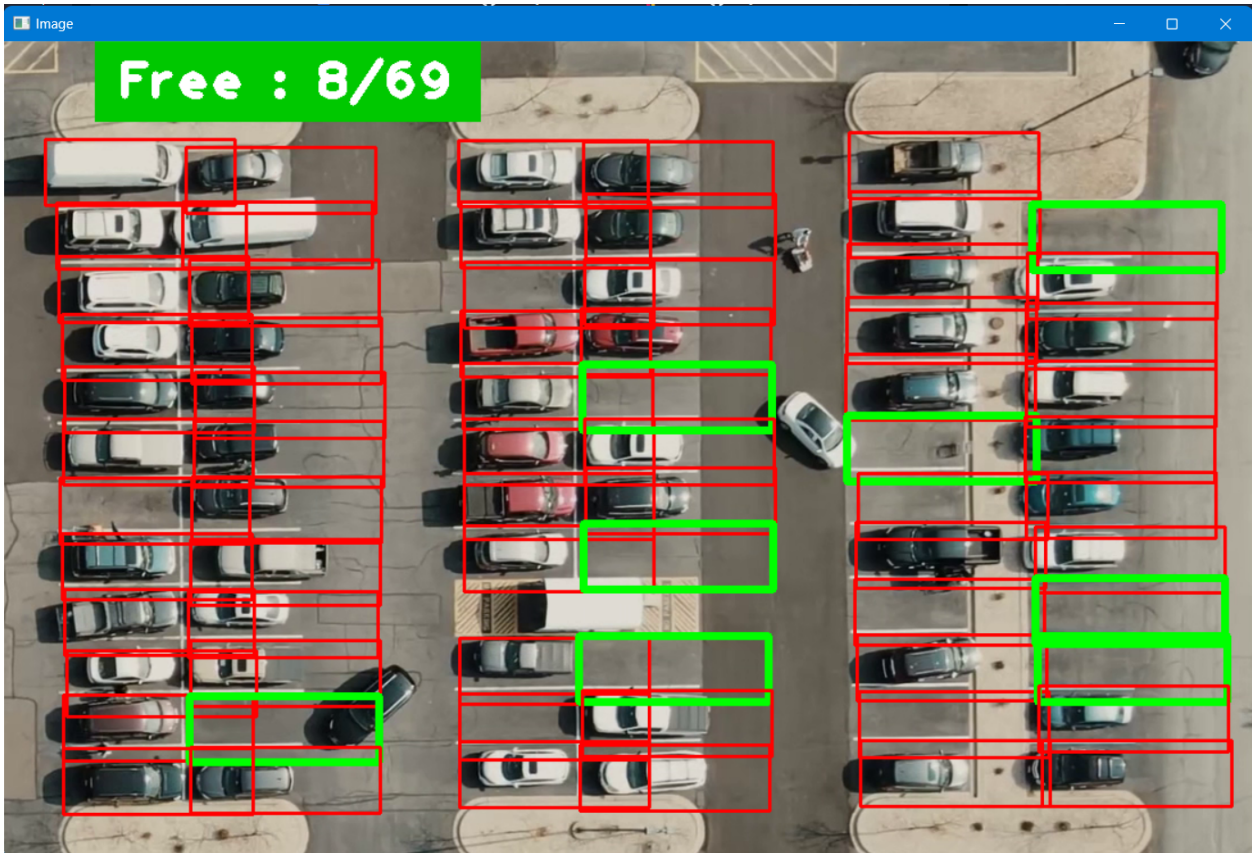
9.4. Login Page:



9.5. Model Page:



9.6. Predictions:



10. ADVANTAGES & DISADVANTAGES

10.1. ADVANTAGES:

10.1.1. Efficient Parking Management:

Provides real-time information about available parking spaces, optimizing parking management and reducing congestion.

10.1.2. User Convenience:

Enhances user experience by providing easy access to parking availability information through a user-friendly interface.

10.1.3. Cost-Efficiency:

Optimizes parking space utilization, reducing operational costs and maximizing revenue generation for parking operators.

10.1.4. Time-Saving:

Saves users' time by enabling them to quickly locate and secure parking spaces without unnecessary searching.

10.2. DISADVANTAGES:

10.2.1. Dependency on Video Input:

Relies on a video feed for parking space detection, which might not work optimally in low-light or adverse weather conditions.

10.2.2. Accuracy Challenges:

The accuracy of parking space detection might be affected by video quality and environmental factors, leading to false positives or negatives.

10.2.3. Limited Integration:

Current version lacks integration with additional features like payment systems, Google Maps API, or security surveillance, limiting overall functionality.

10.2.4. Maintenance:

Requires regular maintenance to ensure video feeds are clear, and the system operates efficiently, incurring additional costs.

11. CONCLUSION

The AI-enabled car parking system presents a significant advancement in parking management, offering users real-time information about available parking spaces. While it enhances user convenience and operational efficiency, there are challenges such as dependency on video input quality and limited integration with other essential services. Addressing these challenges is crucial to creating a robust and reliable parking solution.

12. FUTURE SCOPE

12.1. Payment Integration:

Integrate payment gateways to enable users to pay for parking spaces online, enhancing user convenience and providing a seamless experience.

12.2. Google Maps API Integration:

Implement Google Maps API to provide users with directions to available parking spaces, optimizing their route based on real-time traffic data.

12.3. Robotics Integration:

Explore the integration of robotics for automated parking assistance, allowing for more efficient use of parking spaces and reduced human intervention.

12.4. Enhanced Security Features:

Implement advanced security features such as surveillance cameras, automatic number plate recognition (ANPR), and alarm systems to enhance the overall safety and security of the parking area.

12.5. Smart Parking Guidance System:

Develop a smart parking guidance system that directs users to the nearest available parking spot in real-time, minimizing search time and reducing congestion.

13. APPENDIX

GitHub Link:

https://github.com/smartinternz02/SI-GuidedProject-603184-1697647448

Project Demo Link:

https://drive.google.com/file/d/11l_vLy-NqIMp5udb1Z8SzDqNdEVs_DJo/view?usp=drive_link