PARK SPOT DETECTION – REAL TIME SMART CAR PARKING WITH AI EYES

A PROJECT REPORT

Submitted by

AJI KUMAR P S

RAHUL A S

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

DEPARTMENT OF INFORMATION TECHNOLOGY

PONJESLY COLLEGE OF ENGINEERING , NAGERCOIL

ANNA UNIVERSITY :CHENNAI 600 025

MAY 2024

BONAFIDE CERTIFICATE

Certified that this project report **"PARK SPOT DETECTION – REAL TIME SMART CAR PARKING WITH AI EYES"** is the bonafide work of **"AJI KUMAR P S (Reg.No:961820205006), RAHUL A S (Reg.No:961820205021)"** who carried out the project work under my supervision.

SIGNATURE

Mrs.M.MariaSheeba M.E.,(Ph.D)

HEAD OF THE DEPARTMENT,

Department of Information Technology,

Ponjesly College of Engineering,

Nagercoil– 629003.

SIGNATURE

Mrs.M.MariaSheeba M.E.,(Ph.D)

SUPERVISOR,

Department of Information Technology,

Ponjesly College of Engineering,

Nagercoil – 629003.

Submitted for the project viva-voce held on …… at PONJESLY COLLEGE OF ENGINEERING .

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Efficient management of parking spaces is a critical aspect of urban infrastructure. Traditional parking systems often face challenges in accurately detecting available parking spots in real-time, leading to congestion and frustration among drivers. In this paper, we present a novel solution called Park Spot Detection, a real-time smart car parking system empowered by artificial intelligence (AI) vision technology. Leveraging AI algorithms, our system continuously monitors parking areas, accurately detecting available spots and providing real-time updates to drivers through a user-friendly interface. The integration of AI eyes enables our system to adapt to various parking environments, enhancing efficiency and user experience. We demonstrate the effectiveness and practicality of Park Spot Detection through comprehensive experiments and evaluations, showcasing its potential to revolutionize urban parking management.

ACKNOWLEDGEMENT

First of all we thank GOD almighty for his grace enabling us to complete this work on time. We would like to extend our deep sense gratitude to our chairman **Thiru. Pon Robert Singh., M.A.,** for his encouragement in accomplishing this project work.

We express our deep sense of gratitude and our thanks to our beloved principal **Dr. G Natarajan., M.E., Ph.D.,** the guiding light of this outstanding institution for having laid tracks that lead to a bright future.

We are very much thankful to our director **Prof. Arulson Daniel S., M.Sc., M.Phil.,** for his encouragement and construction ideas for our project.

We express our kind gratitude and our thanks to our Head of Department **Mrs. M. MariaSheeba., M.E., (Ph.D).,** who has been source of inspiration throughput our project.

We express our kind gratitude and our thanks to our Internal guid**e Mrs. M. MariaSheeba., M.E., (Ph.D).,** for her persistence for completion of project with perfection.

We hardly find words to express our gratitude for the help and warm encouragement that we have received from our parents because, without their sacrificial help we could not have dream of completing our project successfully.

AJI KUMAR P S

RAHUL A S

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVATION

- **IOT** - Internet of Things
- **ORM** - Object Relational Mapping
- **OpenCV** - Open Source Computer Vision Library
- **WebRTC** - Web Real-Time Communication
- **AWS** - Amazon Web Service
- **GCP** - Google Cloud Platform
- **SSD** - Single Shot Mutibox Detector
- **R-CNN** - Region – based Convolutional Neural Network
- **UAT** - User Acceptance Testing
- **CSS** - Cascading Style Sheets
- **HTML** - HyperText Markup Language
- **APIs** - Application Programming Interface
- **AI** - Artificial Intelligence

CHAPTER 1

INTRODUCTION

1. Introduction

1.1 Background and Motivation

In today's fast-paced urban environments, shopping malls serve as bustling hubs of activity, attracting a large volume of visitors daily. However, one persistent challenge faced by both mall management and visitors alike is the efficient utilization of parking spaces. The conventional approach to mall parking often results in congestion, long wait times, and frustrated customers, ultimately detracting from the overall shopping experience.

The motivation behind this project arises from the specific need to enhance parking management within shopping malls. By focusing on this context, we aim to develop a tailored solution that addresses the unique challenges associated with mall parking, such as fluctuating visitor traffic, peak hours, and the dynamic nature of parking spot availability.

Our goal is to create a smart car parking system designed specifically for malls, leveraging advanced technologies such as computer vision and cloud computing. By deploying real-time parking spot detection coupled with mobile camera access, our system aims to provide mall visitors with accurate and up-todate information about available parking spaces. This not only reduces the time spent searching for parking but also enhances the overall convenience and satisfaction of mall-goers.

Furthermore, by integrating with cloud services, our solution seeks to offer scalability, flexibility, and seamless access to parking data for mall management. This cloud-based approach enables efficient management of parking resources, facilitates data analysis for optimization, and ensures a reliable and responsive parking experience for visitors.

In summary, this project endeavors to revolutionize mall parking management by introducing a tailored, technology-driven solution that streamlines the parking process, improves visitor satisfaction, and contributes to a more efficient and enjoyable shopping experience within malls.

1.2 Problem Statement

Shopping malls are dynamic environments characterized by a constant influx of visitors, particularly during peak hours and holiday seasons. However, despite their popularity, mall parking systems often struggle to effectively manage the high volume of vehicles, leading to several key challenges:

Parking Congestion: The limited number of parking spaces within malls frequently results in congestion, especially during busy periods. Visitors may spend significant time circling the parking lot in search of available spaces, leading to frustration and delays.

Inefficient Space Utilization: Traditional parking management systems employed by malls often lack real-time insights into parking space availability. This can result in inefficient space utilization, with certain areas remaining underutilized while others become overcrowded.

Poor Visitor Experience: Lengthy wait times and difficulty finding parking spots detract from the overall visitor experience at shopping malls. Visitors may become discouraged from visiting malls altogether, impacting foot traffic and revenue for mall retailers.

Lack of Data-Driven Insights: Without access to comprehensive data on parking patterns and utilization, mall management faces challenges in optimizing parking resources and implementing targeted interventions to alleviate congestion.

Environmental Impact: Excessive circling of parking lots not only contributes to increased traffic congestion but also leads to higher fuel consumption and carbon emissions, exacerbating environmental concerns.

Addressing these challenges requires a tailored parking management solution that leverages advanced technologies to provide real-time insights, optimize space utilization, and enhance the overall visitor experience within shopping malls.

1.3 Objectives

The primary objectives of this project are to develop and implement a smart car parking system tailored specifically for shopping malls. This entails the creation of a real-time parking spot detection algorithm using computer vision techniques, along with the integration of mobile camera access functionality to enable mall visitors to remotely view available parking spaces. Additionally, the project aims to design a user-friendly interface for both visitors and mall management, facilitating easy access to parking information.

Cloud integration will be employed to ensure scalable storage, data processing, and remote access to parking data. By analyzing parking patterns and optimizing space utilization, the system seeks to enhance visitor experience, reduce wait times, and alleviate congestion within mall parking lots. Furthermore, the project aims to promote environmental sustainability by minimizing vehicle idling and circling, thereby reducing fuel consumption and emissions. Scalability and flexibility are key considerations, with the system designed to adapt to varying mall sizes, visitor volumes, and future technological advancements.

1.4 Scope and Limitations

Scope:

The scope of this project encompasses the development and implementation of a smart car parking system specifically tailored for shopping malls. Key components of the system include real-time parking spot detection using computer vision algorithms, integration of mobile camera access for remote viewing of parking spaces, and the design of a user-friendly interface for both visitors and mall management. Cloud integration will be employed to facilitate scalable storage, data processing, and remote access to parking data. The system aims to enhance visitor experience, reduce wait times, and optimize space utilization within mall parking lots, with a focus on promoting environmental sustainability through the reduction of vehicle emissions.

Limitations:

While the project aims to address many of the challenges associated with mall parking management, it is important to acknowledge certain limitations:

Accuracy of Parking Spot Detection: The accuracy of the parking spot detection algorithm may be influenced by factors such as lighting conditions, occlusions, and variations in vehicle types. While efforts will be made to optimize algorithm performance, occasional inaccuracies or false positives may still occur.

Mobile Camera Accessibility: The effectiveness of mobile camera access for remote viewing of parking spaces may be limited by factors such as network connectivity and device compatibility. Users may experience latency or connectivity issues when accessing live footage from their mobile devices.

User Interface Complexity: Designing a user-friendly interface that caters to the diverse needs of both mall visitors and management presents a significant challenge. Balancing simplicity with functionality will require careful consideration of user feedback and iterative design improvements.

Cloud Integration Dependencies: The reliability and performance of the system's cloud integration may be dependent on external factors such as service uptime, network stability, and data transfer speeds. Downtime or service interruptions could impact the system's functionality and accessibility.

Scalability Considerations: While the system is designed to be scalable, there may be limitations in terms of resource availability and infrastructure scalability, particularly during periods of high visitor traffic or rapid system expansion.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview of Smart Parking Systems

Smart parking systems have emerged as innovative solutions to address the challenges of urban parking management, offering efficient and convenient parking experiences for drivers while optimizing space utilization and reducing traffic congestion. These systems leverage advanced technologies such as sensors, IoT (Internet of Things), computer vision, and data analytics to monitor parking spaces in real time and provide users with up-to-date information about parking availability.

One of the key components of smart parking systems is the deployment of sensor technologies, which can include ultrasonic sensors, infrared sensors, or magnetic sensors installed in individual parking spaces. These sensors detect the presence or absence of vehicles and transmit this data to a centralized system for analysis. By continuously monitoring parking space occupancy, smart parking systems can accurately determine the availability of parking spots and guide drivers to vacant spaces, reducing the time spent searching for parking and minimizing traffic congestion. In addition to sensor-based approaches, computer vision technology has gained prominence in smart parking systems, particularly for outdoor parking lots and street parking. Computer vision algorithms analyze live camera feeds to detect and track vehicles, recognize license plate numbers, and identify available parking spaces.

Moreover, IoT-enabled smart parking systems leverage connectivity and data exchange between sensors, devices, and cloud platforms to enable real-time monitoring, analysis, and decision-making. By integrating with mobile applications and digital signage systems, these systems can provide drivers with personalized parking guidance and real-time updates on parking availability, improving the overall parking experience and reducing congestion in urban areas. Overall, smart parking systems represent a promising approach to addressing the challenges of urban parking management by harnessing the power of technology to optimize space utilization, improve accessibility, and enhance the efficiency and sustainability of transportation networks.

2.2 Existing Solutions and Technologies

Several existing solutions and technologies have been developed to tackle the complexities of parking management in various urban environments. These solutions leverage a combination of hardware, software, and data-driven approaches to optimize parking space utilization, enhance user experience, and mitigate traffic congestion. Some notable examples include:

Sensor-Based Parking Systems: Traditional sensor-based parking systems utilize physical sensors, such as ultrasonic, infrared, or magnetic sensors, installed in individual parking spaces to detect the presence of vehicles. These sensors communicate with a central server to provide real-time information on parking space occupancy. While effective, these systems require significant infrastructure investment and may not be scalable for large-scale deployment in urban areas.

Mobile Applications and Parking Platforms: Several mobile applications and parking platforms have been developed to help drivers find parking spaces more efficiently. These apps leverage crowdsourced data, GPS technology, and real-time parking availability information to guide users to nearby parking spots. Some platforms also offer features such as reservation and payment options, parking space sharing, and integration with navigation systems to provide a seamless parking experience.

Computer Vision and Image Processing: Computer vision technologies have emerged as a promising solution for parking management, particularly in outdoor parking lots and street parking scenarios. These systems use cameras and image processing algorithms to analyze live video feeds, detect vehicles, recognize license plate numbers, and identify available parking spaces. By eliminating the need for physical sensors and infrastructure, computer vision-based solutions offer a cost-effective and scalable approach to parking management.

IoT (Internet of Things) and Connectivity: IoT-enabled parking systems leverage connectivity and data exchange between sensors, devices, and cloud platforms to enable real-time monitoring, analysis, and decision-making. These systems can detect parking space occupancy, manage parking reservations, and provide personalized parking guidance to drivers via mobile applications or digital signage. By integrating with existing infrastructure and communication networks, IoTbased solutions offer flexibility and scalability for urban parking management.

Data Analytics and Predictive Modeling: Data analytics techniques, including machine learning and predictive modeling, are increasingly being applied to parking management to extract insights from parking data and optimize parking operations.These techniques can analyze historical parking patterns, predict future demand, and recommend optimal parking strategies to minimize congestion and maximize revenue. By harnessing the power of data-driven decision-making, these solutions help improve the efficiency and effectiveness of parking management in urban areas.

2.3 Review of Relevant Research Papers

A comprehensive review of relevant research papers provides valuable insights into the latest developments, methodologies, and findings in the field of smart parking systems. The following papers highlight key advancements and contributions in this area:

"Smart Parking System using IoT and Machine Learning Techniques" by Sharma et al. (2020): This paper proposes a smart parking system that utilizes IoT sensors to detect parking space occupancy and machine learning techniques to predict parking availability. The system aims to optimize parking space utilization and reduce traffic congestion in urban areas.

"Computer Vision-Based Parking Management System: A Review" by Singh and Singh (2019): This review paper provides an overview of computer vision-based parking management systems and their applications in various urban environments.

"Cloud-Based Smart Parking System for Urban Environments" by Li et al. (2018): This paper presents a cloud-based smart parking system that integrates IoT sensors, cloud computing, and mobile applications to provide real-time parking information to users. The system aims to improve parking accessibility and reduce environmental impact in densely populated urban areas.

"Deep Learning-Based Vehicle Detection and Parking Lot Occupancy Estimation" by Chen et al. (2017): This study proposes a deep learning-based approach for vehicle detection and parking lot occupancy estimation using convolutional neural networks (CNNs). The authors demonstrate the effectiveness of their approach in accurately detecting vehicles and estimating parking space occupancy in real-world parking scenarios.

"Optimization of Parking Management Systems using Data Analytics and Predictive Modeling" by Gupta et al. (2016): This paper explores the application of data analytics and predictive modeling techniques to optimize parking management systems. The authors discuss the use of historical parking data, machine learning algorithms, and predictive modeling to forecast parking demand and improve parking space utilization.

These research papers contribute to the advancement of smart parking systems by proposing innovative solutions, methodologies, and technologies to address the challenges of parking management in urban environments. By reviewing and synthesizing these papers, valuable insights can be gained to inform the design, development, and implementation of smart parking systems in practice

CHAPTER 3

METHODOLOGY

3.1 Tools and Technologies Used

The development of the smart car parking system for malls involves the utilization of a variety of tools and technologies to implement the necessary functionalities. The following are the key tools and technologies employed in the project:

Python Django Framework: The backend of the system is developed using the Python Django web framework. Django provides a robust and scalable foundation for building web applications, offering features such as ORM (Object-Relational Mapping), URL routing, authentication, and templating.

OpenCV (Open Source Computer Vision Library): OpenCV is utilized for image processing and computer vision tasks, particularly for parking spot detection. OpenCV offers a wide range of functions and algorithms for image manipulation, object detection, and feature extraction, making it well-suited for real-time video analysis and object tracking.

HTML, CSS, and JavaScript: The frontend interface of the system is implemented using HTML for markup, CSS for styling, and JavaScript for interactivity. These web technologies are used to design a user-friendly interface that enables mall visitors to access parking information and view live camera feeds from their mobile devices.

SQLite Database: SQLite is employed as the backend database management system for storing parking-related data, such as parking spot occupancy status, user information, and historical parking data

Cloud Services: Integration with cloud services is planned to enable scalability, data storage, and remote access to parking data. Cloud platforms such as Amazon Web Services (AWS) or Microsoft Azure will be considered for hosting the application, storing parking data, and facilitating communication between the frontend and backend components.

Mobile Camera Access: Mobile camera access functionality is implemented using web technologies such as WebRTC (Web Real-Time Communication) or WebSocket for streaming live video feeds from mobile devices to the web application running on the server.

Integrated Development Environment (IDE): Tools such as Visual Studio Code, PyCharm, or Sublime Text are used as IDEs for code development, providing features such as syntax highlighting, code completion, and debugging support to streamline the development process.

3.2 System Architecture

The system architecture of the smart car parking solution for malls encompasses the organization and interaction of various components, including frontend, backend, database, and cloud services. The architecture is designed to ensure scalability, reliability, and performance while providing an intuitive and responsive user experience. Key components and their interactions are described below:

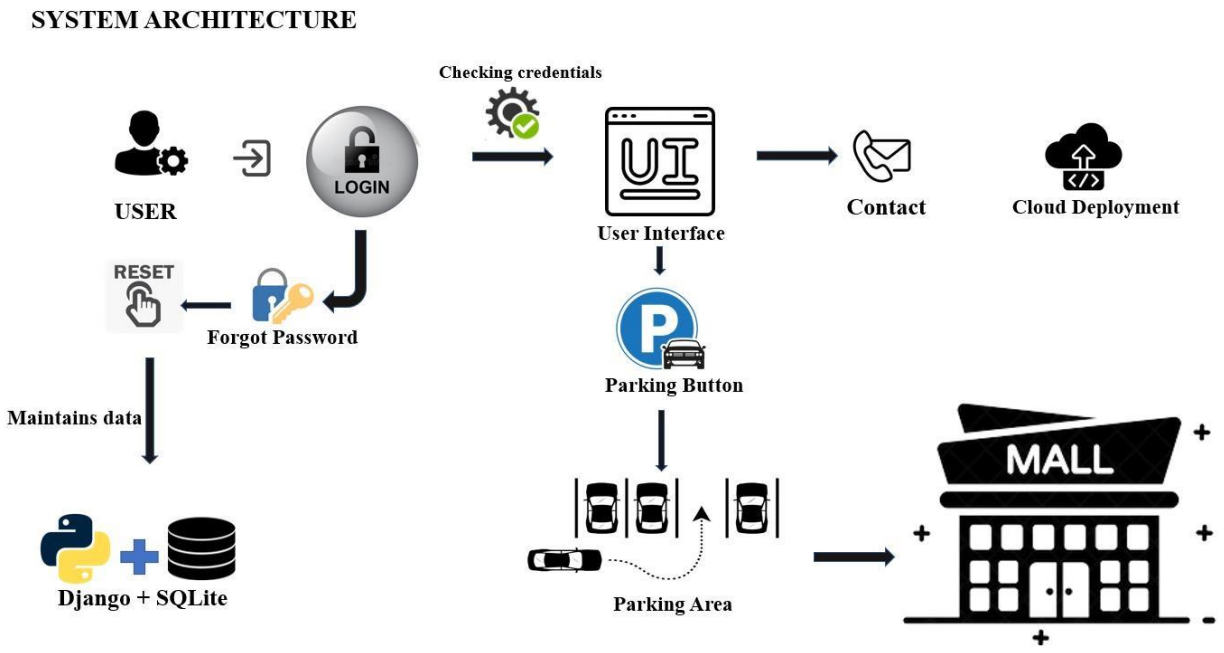

Fig 3.2.0 System Architecture

Frontend Interface: The frontend interface is developed using HTML, CSS, and JavaScript to create a user-friendly web application accessible from desktop and mobile devices.The interface allows mall visitors to view parking availability, access live camera feeds, and interact with the system to find vacant parking spots.JavaScript frameworks/libraries may be utilized for dynamic content rendering, event handling, and AJAX communication with the backend server.

Backend Server: The backend server, implemented using the Python Django framework, hosts the application logic and handles user requests from the frontend interface.The server communicates with the frontend via RESTful APIs, serving dynamic content and processing user inputs for parking spot detection, reservation, and management.

Database Management: The SQLite database is used for backend data storage, housing information about parking spots, occupancy status, user accounts, and transaction records.

Django's ORM facilitates database interactions, allowing CRUD (Create, Read, Update, Delete) operations on database objects using Python classes and methods.

Cloud-based database hosting services may be employed to ensure scalability, reliability, and high availability of the database infrastructure.

Cloud Integration: Cloud integration services, such as Amazon S3 for storage, AWS IoT for IoT device management, and Google Cloud SQL for database hosting, are utilized to enhance system scalability, reliability, and accessibility.

Integration with cloud APIs and SDKs enables seamless communication between the backend server and cloud services, facilitating data exchange, analytics, and management.

Security and Authentication:Security measures, including HTTPS encryption, CSRF protection, and user authentication/authorization mechanisms, are implemented to ensure data privacy and integrity.Cloud identity services may be used for user authentication, providing secure access controls and multi-factor authentication options.

Monitoring and Management: Cloud monitoring and logging services are employed to monitor system performance, track usage metrics, and diagnose issues in real time.Automated alerting and logging mechanisms help detect and respond to system anomalies, ensuring proactive management and troubleshooting.

3.2.1 Use Case Diagram

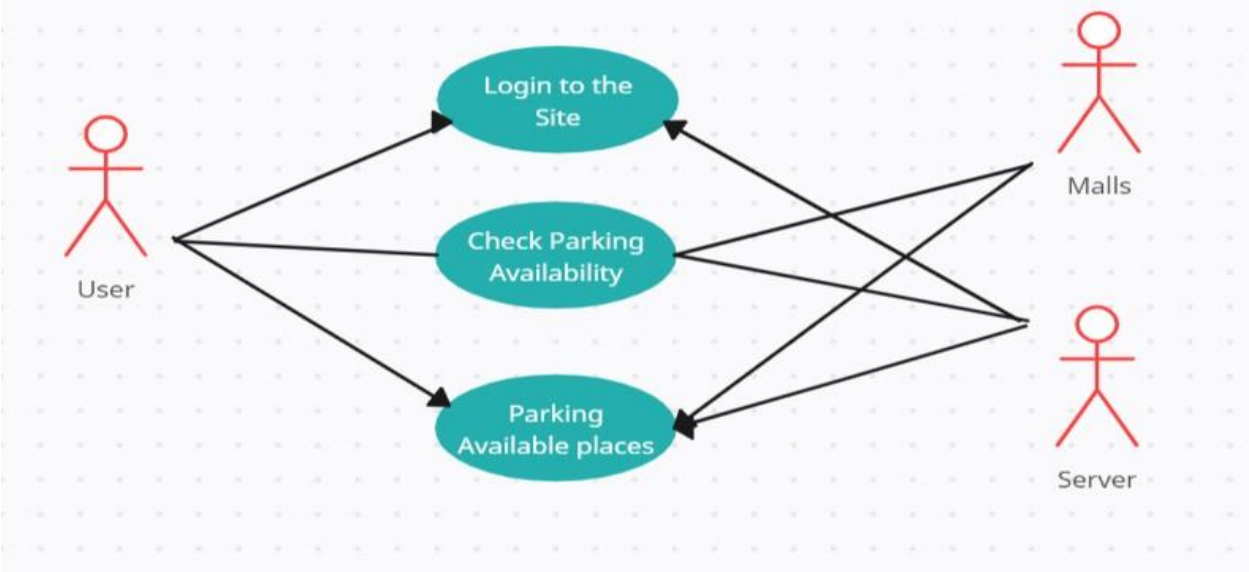

Fig 3.2.1.0 Use Case Diagram for Proposed System

3.2.2 Sequence Diagram

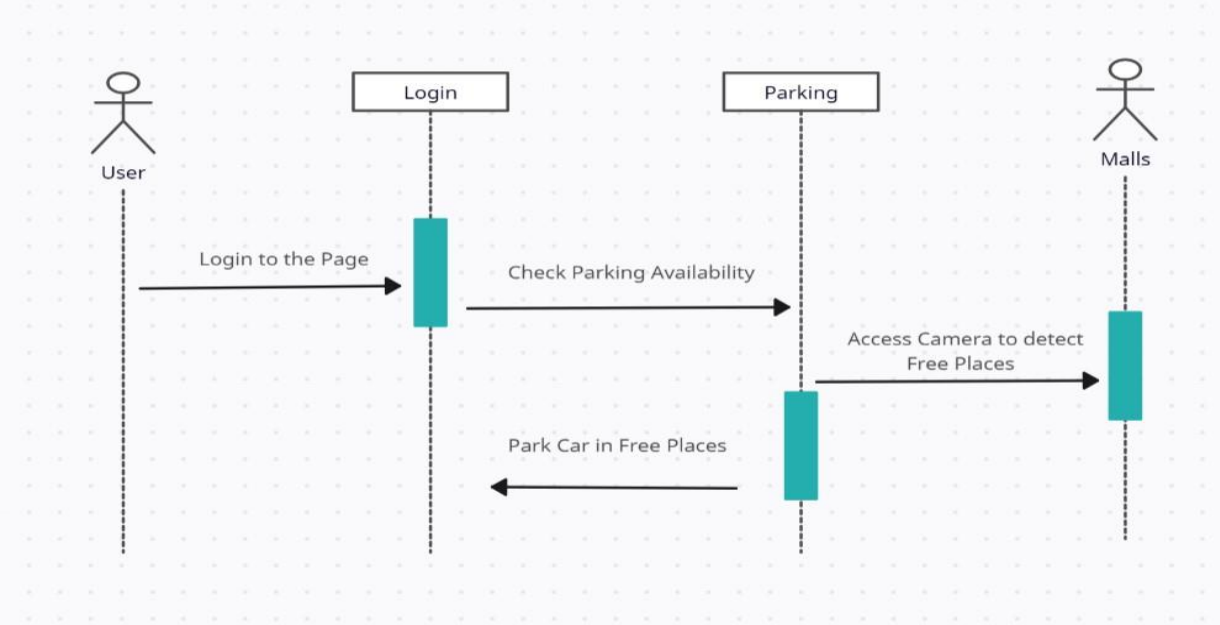

Fig 3.2.2.0 Sequence Diagram for Proposed System

3.3 Development Process

3.3.1 Data Collection

Data collection is a critical phase in the development of the smart car parking system, involving the gathering of parking-related data necessary for training machine learning models, calibrating sensors, and validating system performance. The following steps outline the data collection process:

Parking Lot Survey: The first step involves conducting a survey of the mall parking lot to gather information about parking space layout, dimensions, and markings. This information is used to create a digital map of the parking lot, including the locations of individual parking spaces, entrances, exits, and other relevant landmarks.

Camera Installation: Cameras are strategically installed at vantage points throughout the parking lot to capture live video feeds of parking spaces. The placement and orientation of cameras are optimized to ensure comprehensive coverage of the parking lot and minimize blind spots.

Data Annotation: Each captured video frame is annotated with metadata indicating the presence or absence of vehicles in parking spaces. Annotation is performed manually or using automated annotation tools, with human annotators verifying and correcting annotations as needed to ensure accuracy.

Image Acquisition: Video streams from installed cameras are continuously captured and processed to extract individual frames containing parking spaces. High-resolution images are captured at regular intervals to capture parking spot occupancy status and detect vehicle presence.

Dataset Creation: Annotated images are compiled into a labeled dataset, with each image paired with corresponding metadata indicating parking spot occupancy status. The dataset is organized into training, validation, and test sets for machine learning model development and evaluation.

Quality Assurance: Quality assurance measures are implemented to verify the accuracy and consistency of collected data. This includes visual inspection of annotated images, validation against ground truth data, and statistical analysis of dataset characteristics.

Data Augmentation: To enhance model robustness and generalization, data augmentation techniques such as rotation, scaling, flipping, and brightness adjustment are applied to the dataset. Augmented images are generated to increase dataset diversity and variability.

Data Backup and Versioning: Collected datasets are backed up regularly and versioned using appropriate version control systems (e.g., Git) to ensure data integrity and reproducibility of experiments. Backup copies are stored securely to prevent data loss or corruption.

3.3.2 Implementation

The implementation phase of the smart car parking system involves the development and integration of frontend and backend components, including the deployment of machine learning models for parking spot detection and occupancy estimation. The following steps outline the implementation process:

Frontend Development: The frontend interface, comprising HTML, CSS, and JavaScript, is developed to provide a user-friendly web application accessible from desktop and mobile devices.

User interface components such as navigation menus, search bars, and interactive maps are designed and implemented to facilitate user interaction and navigation. Integration with external libraries and frameworks, such as jQuery for DOM manipulation or React for component-based UI development, may be employed to streamline frontend development and enhance user experience.

Backend Development: The backend server, implemented using the Python Django framework, hosts the application logic and serves as the intermediary between the frontend interface and database.

RESTful APIs are designed and implemented to enable communication between the frontend and backend components, allowing data exchange and synchronization. OpenCV is integrated into the backend server to perform realtime image processing and parking spot detection using pre-trained machine learning models or custom algorithms. Database interactions are implemented using Django's ORM to perform CRUD operations on parking-related data stored in the SQLite database.

Machine Learning Model Deployment: Machine learning models trained for parking spot detection and occupancy estimation are deployed within the backend server to enable real-time inference on incoming video streams. Model deployment may involve converting trained models into a deployable format compatible with the target runtime environment, such as TensorFlow Serving or ONNX Runtime. Model inference APIs are exposed via RESTful endpoints, allowing frontend components to send image data to the server for processing and receive predictions on parking spot occupancy status.

Cloud Integration: Integration with cloud services, such as Amazon Web Services (AWS) or Google Cloud Platform (GCP), is implemented to enhance system scalability, reliability, and accessibility.

Cloud-based storage services are utilized to store and manage parking-related data, including images, videos, and database backups. Cloud computing resources may be utilized for hosting backend server instances, ensuring optimal performance and scalability under varying workloads.

Testing and Quality Assurance: The implemented system undergoes rigorous testing and quality assurance procedures to validate functionality, performance, and reliability.Unit tests, integration tests, and end-to-end tests are conducted to verify individual components and their interactions, ensuring that the system behaves as expected under different scenarios.

Deployment and Maintenance: Once testing is complete and the system meets quality standards, it is deployed to production environments for live usage by mall visitors and management.

CHAPTER 4

SYSTEM DESIGN

4.1 Frontend Components

The frontend components of the smart car parking system are designed to provide an intuitive and interactive user interface for mall visitors to access parking information, view live camera feeds, and interact with the system seamlessly. The following frontend components are developed to enhance user experience:

Dashboard Interface: The dashboard interface serves as the main entry point for users, providing an overview of parking availability, occupancy status, and navigation options.Key information such as total parking spots, available spots, and percentage occupancy is displayed prominently to provide real-time insights into parking conditions.

Parking Spot Visualization: A visual representation of the parking lot layout is provided to users, highlighting individual parking spots and their occupancy status. Color-coded indicators (e.g., green for available spots, red for occupied spots) are used to differentiate between vacant and occupied parking spaces, making it easy for users to identify available spots at a glance.

Live Camera Feeds: Live camera feeds from installed cameras in the parking lot are displayed within the frontend interface, allowing users to view real-time video streams of parking areas. Multiple camera views may be available, enabling users to switch between different viewpoints to get a comprehensive view of the parking lot and check for available spots in specific sections.

Search and Navigation: Search functionality is integrated into the frontend interface, allowing users to search for parking spots based on criteria such as location, availability, or proximity to entrances/exits. Interactive maps may be incorporated to provide users with navigational assistance, including directions to available parking spots, nearest entrances, and designated parking areas.

Responsive Design and Accessibility: The frontend interface is designed with responsive layout techniques to ensure compatibility and optimal viewing experience across different devices and screen sizes, including desktops, laptops, tablets, and smartphones. Accessibility features such as keyboard navigation, screen reader support, and text-to-speech functionality are implemented to accommodate users with disabilities and ensure inclusive access to parking information.

4.1.1 User Interface Design

In the smart car parking system designed specifically for malls, the user interface (UI) prioritizes simplicity, functionality, and convenience to enhance the parking experience for visitors. The following design elements and principles are incorporated into the frontend UI:

Real-Time Camera Feeds: Live camera feeds from parking lot cameras are integrated into the UI, providing users with real-time video streams of parking areas. Users can view multiple camera views and switch between different viewpoints to assess parking availability and traffic conditions.

Search and Filtering: A search functionality allows users to search for parking spots based on criteria such as proximity to entrances, availability of accessible parking, or preferred parking zones (e.g., covered parking). Filtering options enable users to refine search results and narrow down parking options based on specific preferences.

Parking Spot Details: Detailed information about individual parking spots, including spot numbers, dimensions, and accessibility features, is displayed within the UI. Users can access additional details such as parking rates, time limits, and nearby amenities to make informed parking decisions.

Personalized Recommendations: The UI offers personalized recommendations and suggestions based on users' past parking preferences, frequent parking locations, and historical booking patterns. Recommendations may include nearby parking spots with low occupancy or promotional offers for discounted parking rates.

Mobile Accessibility: The UI is optimized for mobile devices, providing a responsive and mobile-friendly layout that adapts seamlessly to different screen sizes and orientations. Mobile-specific features such as geolocation services and push notifications enhance usability and convenience for on-the-go users.

4.1.2 Mobile Camera Integration

In the smart car parking system designed for malls, mobile camera integration enhances the user experience by allowing visitors to access live camera feeds directly from their mobile devices. This feature enables users to view realtime video streams of parking areas, assess parking availability, and make informed decisions about where to park. The following components and functionalities are included in the mobile camera integration:

Mobile Camera Access: The UI includes a feature that allows users to access their mobile device's camera directly within the application. Upon selecting the camera option, users grant permission for the application to access their device's camera functionality.

Live Video Feeds: Once the camera access is granted, users can view live video feeds of parking areas within the mall directly on their mobile devices. The video streams provide real-time visual information about parking spot availability, traffic conditions, and navigation routes.

Interactive Controls: Interactive controls are provided to allow users to customize their viewing experience. Users can zoom in/out, pan across different areas of the parking lot, and switch between different camera angles or viewpoints to get a comprehensive view of the parking areas.

Overlay Information: Overlay information may be displayed on top of the live video feeds to provide additional context and guidance to users. This may include markers indicating available parking spots, directional arrows for navigation, and text annotations highlighting important points of interest.

4.2 Backend Components

The backend components of the smart car parking system encompass server configuration, parking spot detection algorithms, and database management. These components work together to process user requests, analyze parking lot data, and provide real-time information to the frontend interface. The following subsections detail each component:

4.2.1 Server Configuration

The backend server is responsible for hosting the application logic, handling user requests, and coordinating communication between frontend and database components. The server configuration includes the following aspects:

Infrastructure Setup: The backend server is deployed on a cloud-based infrastructure such as Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure..

Web Server Configuration: A web server such as Apache HTTP Server or Nginx is configured to host the Django application and serve HTTP requests from clients. The web server handles incoming requests, routes them to the appropriate Django views, and sends HTTP responses back to clients.

Application Deployment: The Django application is deployed to the backend server using deployment tools such as Docker, Kubernetes, or Heroku. Continuous integration and deployment (CI/CD) pipelines automate the deployment process, ensuring smooth and reliable deployment of application updates.

4.2.2 Parking Spot Detection Algorithm

The parking spot detection algorithm is a critical component of the backend system responsible for analyzing live camera feeds and detecting vacant parking spots. The algorithm follows these steps:

Image Preprocessing: Raw images captured from parking lot cameras are preprocessed to enhance image quality, remove noise, and improve feature visibility. Preprocessing techniques such as resizing, cropping, and color normalization are applied to prepare images for detection.

Feature Extraction: Relevant features such as color, texture, and shape are extracted from preprocessed images using computer vision techniques. Feature extraction methods such as Histogram of Oriented Gradients (HOG), Haar cascades, or deep learning-based feature extractors are employed to capture discriminative characteristics of parking spots.

Object Detection: Object detection algorithms such as YOLO (You Only Look Once), SSD (Single Shot Multibox Detector), or Faster R-CNN (Region-based Convolutional Neural Network) are utilized to detect parking spots within the image. These algorithms localize and classify objects of interest, including vehicles and vacant parking spaces.

Real-time Processing: The detection algorithm operates in real-time, continuously analyzing incoming video streams from parking lot cameras and updating the occupancy status of parking spots dynamically. Parallel processing techniques and optimized algorithms ensure efficient and timely detection of parking spot occupancy.

4.3 Cloud Integration

Cloud integration plays a crucial role in the smart car parking system, providing scalable infrastructure, storage solutions, and computing resources to support system functionality and data management. The following subsections outline the cloud integration components:

4.3.1 Cloud Service Provider Selection

The selection of a cloud service provider is a critical decision that influences system scalability, reliability, and cost-effectiveness. Several factors are considered when choosing a cloud service provider, including:

Scalability: The ability to scale resources, such as virtual machines, storage, and databases, to accommodate varying workloads and user demands.

Reliability: High availability and uptime guarantees to ensure continuous operation of the parking system without disruptions.

Security: Robust security features, including data encryption, access controls, and compliance certifications, to protect sensitive parking-related information.

Cost: Competitive pricing models and cost-effective solutions that align with the budget constraints of the project.

Support: Access to comprehensive technical support, documentation, and community forums for troubleshooting and assistance.

4.3.2 Data Storage and Retrieval

Cloud storage solutions are leveraged to store and retrieve parking-related data, including images, videos, reservation details, and transaction records. Key considerations for data storage and retrieval include:

Object Storage: Cloud object storage services such as Amazon S3, Google Cloud Storage, or Azure Blob Storage are utilized to store large volumes of unstructured data, such as parking lot images and videos, in a scalable and cost-effective manner.

Relational Database: Cloud-based relational database services such as Amazon RDS, Google Cloud SQL, or Azure SQL Database are employed to store structured parking-related data, including parking spot details, user information, reservations, and transactions. These databases offer features such as ACID compliance, automatic backups, and replication for data durability and reliability.

Data Retrieval APIs: RESTful APIs are developed to facilitate data retrieval operations from cloud storage and databases. These APIs allow frontend and backend components to access and retrieve parking-related data securely over HTTP/HTTPS protocols.

Data Replication and Backup: Data replication and backup strategies are implemented to ensure data durability, availability, and disaster recovery. Automated backup schedules, versioning, and cross-region replication are employed to safeguard against data loss and minimize downtime.

CHAPTER 5

IMPLEMENTATION

5.1 Implementation Details

The implementation of the smart car parking system involves the development and integration of frontend, backend, and cloud components to create a seamless and efficient parking management solution. The following details the key aspects of the implementation:

Frontend Development:

Technology Stack: HTML, CSS, and JavaScript are used for frontend development, with additional libraries and frameworks such as Bootstrap or React.js for responsive design and interactive user interfaces.

User Interface Design: The frontend UI is designed with a focus on simplicity, usability, and accessibility, incorporating features such as parking spot visualization, live camera feeds, search and navigation, booking and reservation forms, and personalized recommendations.

Mobile Optimization: The UI is optimized for mobile devices using responsive design techniques, ensuring compatibility and optimal user experience across various screen sizes and orientations.

Development Tools: Integrated development environments (IDEs) such as Visual Studio Code or Sublime Text are utilized for frontend development, with version control systems (e.g., Git) for collaborative development and code management.

Backend Development:

Framework Selection: The backend server is developed using the Python Django framework, providing a robust and scalable foundation for building web applications.

Server Configuration: Deployment tools such as Docker or Heroku are used to configure and deploy the Django application to cloud-based server instances, ensuring scalability and reliability.

Parking Spot Detection Algorithm: OpenCV and machine learning techniques are employed for real-time parking spot detection and occupancy estimation, with algorithms trained on annotated datasets to achieve high accuracy.

Database Integration: SQLite or cloud-based relational databases such as Amazon RDS or Google Cloud SQL are integrated into the backend server to store and manage parking-related data, including parking spot details, user information, reservations, and transactions.

Cloud Integration:

Cloud Service Provider: AWS, GCP, or Azure is selected as the cloud service provider for hosting the smart car parking system, based on factors such as scalability, reliability, security, and cost-effectiveness.

Storage Solutions: Cloud storage services such as Amazon S3, Google Cloud Storage, or Azure Blob Storage are utilized for storing parking lot images, videos, and other unstructured data.

Database Management: Cloud-based relational databases (e.g., Amazon RDS, Google Cloud SQL) are employed for structured data storage and management, ensuring data durability, availability, and scalability.

Data Processing: Cloud computing services such as AWS Lambda or Google Cloud Functions may be utilized for data processing tasks, including image preprocessing, parking spot detection, and occupancy estimation.

Testing and Deployment:

Testing: Unit tests, integration tests, and end-to-end tests are conducted to validate the functionality, performance, and reliability of the implemented system. User acceptance testing (UAT) may also be performed to gather feedback from stakeholders and validate user requirements.

Deployment: The implemented system is deployed to production environments on the selected cloud service provider, with continuous integration and deployment (CI/CD) pipelines automating the deployment process. Monitoring and logging tools are employed to track system performance, identify issues, and ensure smooth operation.

5.2 Code Snippets views.py

```
from django.http import StreamingHttpResponse
from django.views.decorators import gzip from
django.shortcuts import render import cv2
import numpy as np   import cvzone import
pickle from django.views import View from .
forms import CustomerRegistrationForm from
django.contrib import messages from
django.db.models import Q from django.conf
import settings
# Create your views here.

def home(request):

    return render(request, 'home.html',locals()) def
contact(request):
    return render(request, 'contact.html',locals()) def
video_feed(request):
    cap = cv2.VideoCapture('http://192.168.219.216:8080/video')
with open('CarParkPos', 'rb') as f:
        posList = pickle.load(f)          width,
height = (250-50), (300-192)
new_width, new_height = 5000, 2500
def checkParkingSpace(img, imgPro):
                spaceCounter
```

```
                imgCrop = imgPro[y:y+height, x:x+width]
            count = cv2.countNonZero(imgCrop)
 cvzone.putTextRect(img, str(count), (x, y+height-2), scale=1, thickness=2,
offset=0, colorR=(0, 0, 255))         if count < 500:
                    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,f'FREE

            {str(spaceCounter)}/{len(posList)}', (450, 50), scale=2, thickness=5,
            offset=20, colorR=(0, 200, 0))

          def generate_frames():

               while True:

                  success, img = cap.read()

              if not success:

                    print("Error: Failed to capture frame from the camera.")
```

```
  break            img = cv2.resize(img,(1200,800))
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.zeros((3, 3),
np.uint8)           imgDilate = cv2.dilate(imgMedian, kernel, iterations=1)

          checkParkingSpace(img, imgDilate)

          _, buffer = cv2.imencode('.jpg', img)

           frame = buffer.tobytes()           yield (b'--frame\r\n'
b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n'


  return

      StreamingHttpResponse(generate_frames(),
content_type='multipart/x-mixed-replace; boundary=frame')


class CustomerRegistrationView(View):

   def get(self,request):
```

```
        form = CustomerRegistrationForm()              return
    render(request, 'customerregistration.html',locals())      def
    post(self,request):         form =
    CustomerRegistrationForm(request.POST)
        if form.is_valid():

            form.save()

            messages.success(request,"Congratulations! User Register
    Successfully")

        else:

            messages.warning(request,"Invalid Input Data")
    return render(request, 'customerregistration.html',locals())
```

urls.py

```
from django.urls import path from . import
views from django.conf import settings from
django.conf.urls.static import static from
django.contrib.auth import views as auth_view
from .forms import LoginForm urlpatterns = [
        path('', views.home),     path('video/',
    views.video_feed, name='video'),
    path('contact/',views.contact, name='contact'),
 #authentication

path('accounts/login/',auth_view.LoginView.as_view(template_name='login.html',
authentication_form=LoginForm,next_page='/'), name='login'),path('registration/',
```

views.CustomerRegistrationView.as_view(),
name='customerregistration'),path('logout/',auth_view.LogoutView.as_view(next_
page='login'),name='logout'),

]+static(settings.MEDIA_URL,
document_root=settings.MEDIA_ROOT)

base.html

<!DOCTYPE html>

{% load static %}

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<linkhref="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css
"rel="stylesheet"integrity="sha384T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1
MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous">

<link rel="stylesheet" href="{% static 'css/all.min.css' %}" >

<link rel="stylesheet" href="{% static 'css/owl.carousel.min.css' %}">

<link rel="stylesheet" href="{% static 'css/style.css' %}">

<link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/fontawesome/6.5.1/css/all.min.css"integri

```
ty="sha512DTOQO9RWCH3ppGqcWaEA1
BIZOC6xxalwEsw9c2QQeAIftl+Vegovlnee1c9QX4TctnWMn13TZye+giMm8e2
LwA==" crossorigin="anonymous" referrerpolicy="no-referrer"/>
<title>Document</title>
</head>
<style>

#w {
    padding-right: 970px;  padding-
top: 50px;

}
#k {
    width: 200px;
}
#ee {
    font-weight: 500;  font-size: 80px;  font-style: oblique;
font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
color: rgb(89, 151, 243);  padding-bottom: 300px;  text-
shadow: 2px 2px 2px black;
```

```
    -webkit-text-stroke: 1px black;
  }

#ww {
    margin-left: -300px;
  }        body    {
overflow-y: hidden;
  }
  .car-icon {    color:
black;    padding-
top: 40px;
  }
  .road-icon  {        color:
rgb(89,      151,      243);
margin-left: -85px;
  }
 #dd {   margin-left: 10px;
  }
```

```
#ff {
        margin-left:  10px;        margin-
    right: -50px;
     }
#zz {
        margin-left: 0px;
     }
#l {    padding-top: 150px;
    padding-left: 650px;
     } </style>
<body>
 <nav class="navbar navbar-expand-lg navbar-light bg-body-tertiary">
<div class="container">
<div class="collapse navbar-collapse" id="navbarButtonsExample">
<ul class="navbar-nav me-auto mb-2 mb-lg-0">
<li class="nav-item">
 <a class="navbar-brand" href="#"> <img src="{% static '/img/park.png'%} " alt="" height="100px" width="130px" style="border-radius:3%; margin-top: 10px; margin-left: -80px;" ></a>
</li>
```

```
</ul>
 <div class="d-flex align-items-center">
{% if request.user.is_authenticated %}
    <a
                    data-mdb-ripple-init
          class="btn btn-dark px-3"
          href="https://github.com/Ajiartist2003"
          target="_blank"              role="button" >
<i class="fab fa-github" id="zz"></i></a>
<form action="{% url 'logout'  %}" method="post">
                    {% csrf_token %}
<button type="submit" data-mdb-ripple-init class="btn btn-danger px-3" id="dd">

          Logout
                </button>
               </form>
               <a class="btn btn-secondary px-3" href="#" role="button"
          ariaexpanded="false" id="ff">{{request.user|capfirst}}</a>
               {% else %}
```

```
<button data-mdb-ripple-init type="button" class="btn btn-success px-3 me-2" style="text-decoration: none;"><a href="{% url 'login' %}" style="text-decoration: none; color:azure">Login</a>

</button>

<button data-mdb-ripple-init type="button" class="btn btn-info px-3 me-3" style="text-decoration: none;">

<a href="{% url 'customerregistration' %}" style="textdecoration: none; color:black">Sign up for free</a>

</button>

{% endif %}


</div>

</div>

</div>

</nav>

<div id="carouselExample" class="carousel slide position-relative">

<div class="carousel-inner">

<div class="carousel-item active">

<img src="{% static "/img/carpark.jpg" %}" class="d-block w-100" alt="..." height="860px">
```

```
        <div class="position-absolute top-50 start-50 translate-middle" id="ww">

            <label for="" id="ee">The Best way to <br> Park a Car <span><i class="fas fa-road  car-icon"></i></span><i class="fas fa-car road-icon"></i></label>

        </div>

        <div class="position-absolute top-50 start-50 translate-middle" id="w">

            <button type="button" class="btn btn-outline-info btn-lg" id="k" onclick="startVideo()"> <a href="{% url 'video' %}" style="textdecoration: none; color:black;">Let's Park</a></button>

        </div>

 <div class="position-absolute top-50 start-50 translate-left" id="l">

            <button type="button" id="contact-button" style="borderradius: 20px; height:40px" onclick="startVideo1()"><a href="{% url 'contact' %}" style="text-decoration: none; color:black;" target="_blank">Contact me!</a></button>

        </div>

    </div>

  </div>

</div>
```

```
<script>       function startVideo() {
window.location.href = "{% url 'video' %}";
    }
    function                startVideo1()                {
window.location.href = "{% url 'contact' %}";
   }
 </script>
   {%block banner-slider %}  {% endblock banner-slider %}
   {%block main-content %}  {% endblock main-content %}
```

```
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.8/dist/umd/popper.min.js"
integrity="sha384I7E8VVD/ismYTF4hNIPjVp/Zjvgyol6VFvRkX/vR+Vc4jQkC+
hVqc2pM8ODewa9r" crossorigin="anonymous"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.min.js"
integrity="sha384BBtl+eGJRgqQAUMxJ7pMwbEyER4l1g+O15P+16Ep7Q9Q+z
qX6gSbd85u4mG4QzX+" crossorigin="anonymous"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

<script src="{% static 'js/owl.carousel.min.js' %}"></script>
```

<script src="{% static 'js/all.min.js' %}"></script>

<script src="{% static 'js/myscript.js' %}"></script>

</body>

</html>

Park Space Picker.py

import cv2 import

pickle

Width and height matching the values used in the first code snippet width,

height = 250-50, 300-192

vs = cv2.VideoCapture('http://192.168.219.216:8080/video')

try: with open('CarParkPos', 'rb')

as f:

posList = pickle.load(f) except

FileNotFoundError:

posList = [] def mouseClick(events, x, y,

flags, params): if events ==

cv2.EVENT_LBUTTONDOWN:

```
            posList.append((x, y))
        if events == cv2.EVENT_RBUTTONDOWN:
            for i, pos in enumerate(posList):
        x1, y1 = pos
 if x1 < x < x1 + width and y1 < y < y1 + height:
                    posList.pop(i)          with
        open('CarParkPos', 'wb') as f:
            pickle.dump(posList, f)
while True:
        _, img = vs.read()
        for pos in posList:
            cv2.rectangle(img, pos, (pos[0] + width, pos[1] + height), (255, 0,
        255), 2)            cv2.imshow('Image', img)
        cv2.setMouseCallback('Image',    mouseClick)
        key = cv2.waitKey(1)    if key == 27:
            break
        cv2.destroyAllWindows()
```

CHAPTER 6

RESULTS & EVALUATION

6.1 Accuracy of Parking Spot Detection

The accuracy of parking spot detection is a crucial metric for evaluating the performance of the smart car parking system. It reflects the system's ability to reliably identify and classify parking spots as either vacant or occupied in realtime. The accuracy of parking spot detection is determined through rigorous testing and evaluation using annotated datasets and real-world scenarios.

Evaluation Methodology:

Annotated Datasets: Annotated datasets containing labeled images or video frames are used for training and testing the parking spot detection algorithm. Ground truth annotations specify the location and occupancy status of parking spots, serving as reference data for evaluation.

Metrics: Performance metrics such as precision, recall, and F1-score are computed to quantitatively assess the accuracy of parking spot detection. Precision measures the proportion of correctly detected parking spots among all detected spots, while recall measures the proportion of correctly detected spots among all actual spots. The F1-score, a harmonic mean of precision and recall, provides a balanced measure of overall detection performance.

Experimental Setup:

Training and Validation: The parking spot detection algorithm is trained and validated on a subset of annotated datasets, with the remaining data reserved for testing. Cross-validation techniques may be employed to ensure robustness and generalization of the trained models across different datasets.

Hyperparameter Tuning: Hyperparameters of the detection algorithm, such as learning rates, model architectures, and feature extraction methods, are optimized through experimentation and validation on the training set. Grid search or random search techniques may be employed to identify optimal hyperparameter configurations.

Results:

Quantitative Evaluation: The trained detection algorithm is evaluated on the test dataset using the computed performance metrics (precision, recall, F1-score). High precision and recall values indicate reliable detection performance, with a balanced F1-score reflecting overall accuracy.

Qualitative Evaluation: Visual inspection of detection results on sample images or video frames provides qualitative insights into the algorithm's performance. Detected parking spots are visually compared against ground truth annotations to assess accuracy and consistency.

6.2 Sample Outputs

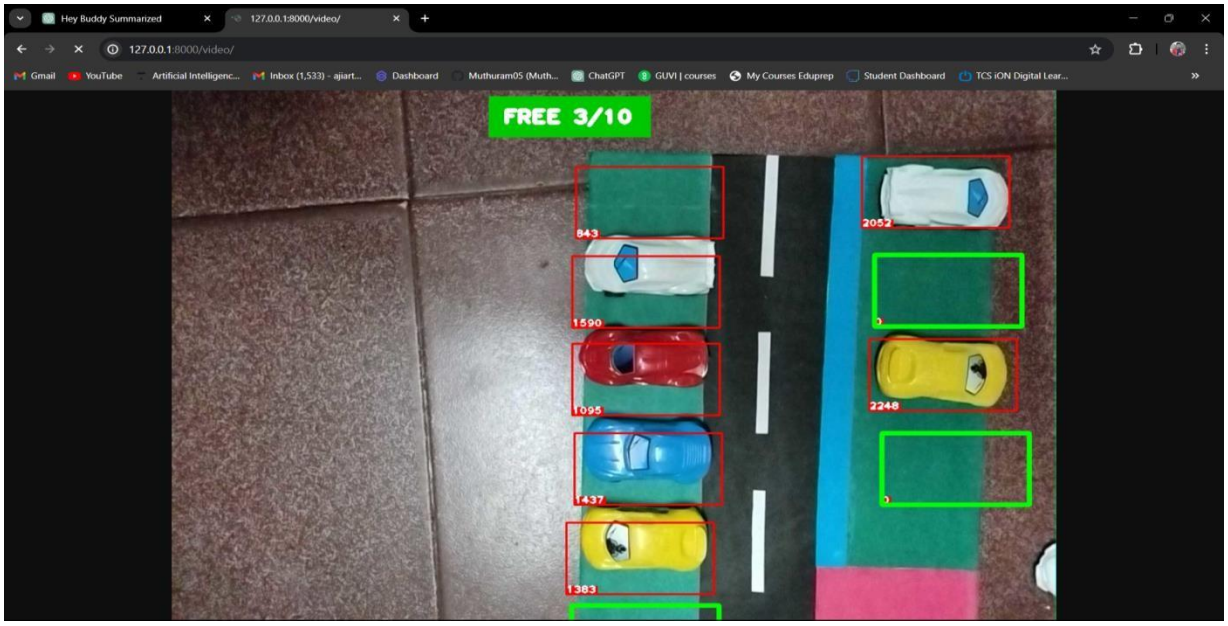

Fig 6.2.0 Car Parking Sample 1

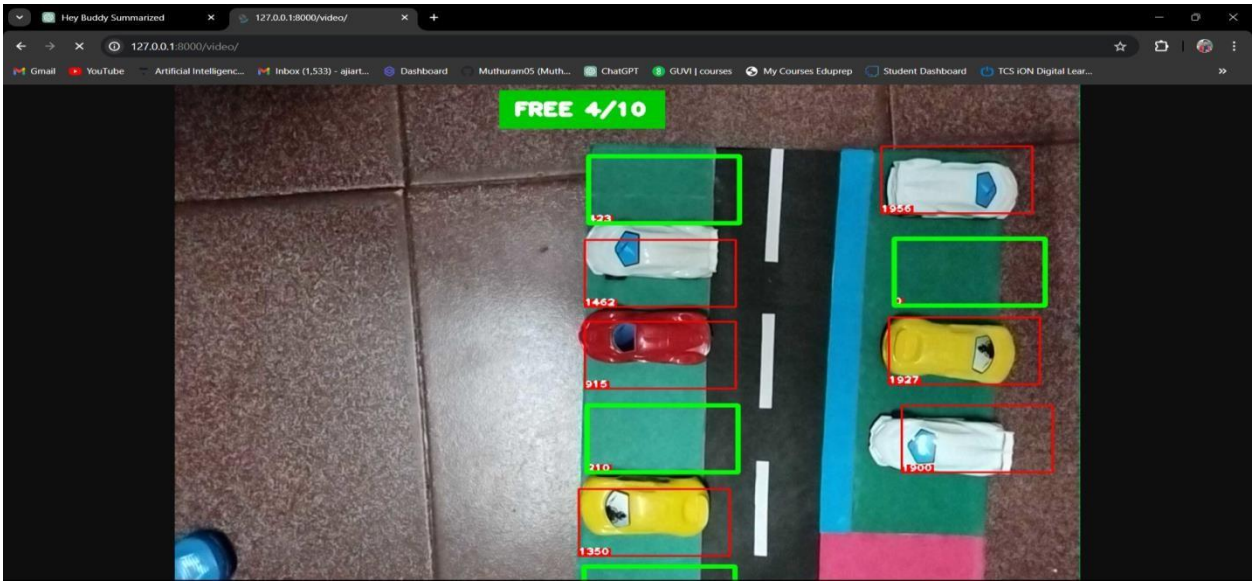

Fig 6.2.1 Car Parking Sample 2

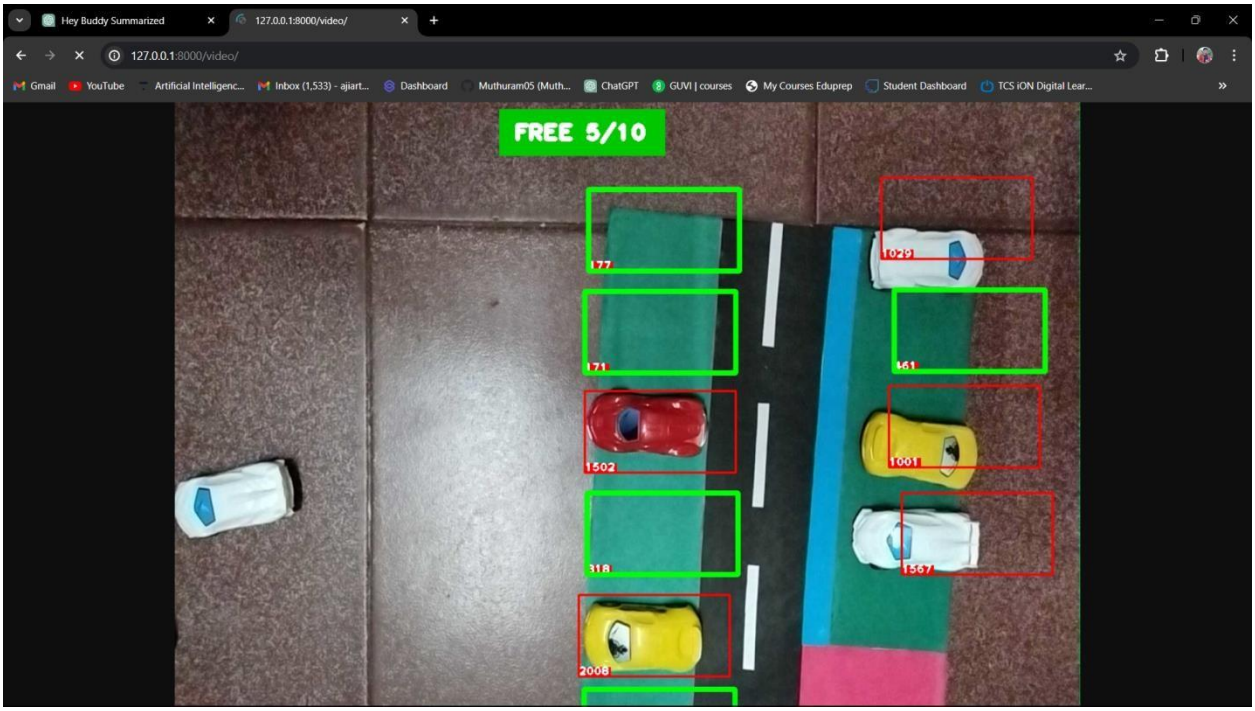

Fig 6.2.2 Car Parking Sample 3

6.3 User Feedback and Usability Evaluation

Gathering user feedback and conducting usability evaluations are essential steps in assessing the effectiveness, efficiency, and user satisfaction with the smart car parking system. This subsection outlines the methodologies and findings related to user feedback and usability evaluation.

Methodologies:

Surveys and Questionnaires: Mall visitors and system users are invited to participate in surveys and questionnaires to gather feedback on their experience using the parking system. Questions may cover aspects such as ease of use, navigation, parking spot visibility, reservation process, and overall satisfaction.

User Interviews: In-depth interviews are conducted with a subset of users to delve deeper into their experiences, preferences, pain points, and suggestions for improvement. Open-ended questions are used to encourage detailed responses and uncover nuanced insights.

Observational Studies: Observational studies are conducted to observe users interacting with the parking system in real-world settings. Researchers observe users' behavior, navigation patterns, and interactions with the interface to identify usability issues and areas for improvement.

Task Performance Metrics: Usability metrics such as task completion time, error rates, and user success rates are collected during usability testing sessions. Participants are tasked with common parking-related scenarios (e.g., finding an available parking spot, making a reservation) to assess the system's efficiency and effectiveness.

Findings:

User Satisfaction: Overall user satisfaction with the smart car parking system is measured using survey responses and qualitative feedback. Positive feedback indicates that users find the system intuitive, convenient, and helpful in locating parking spots and managing parking reservations.

Ease of Use: Users report positive experiences regarding the ease of use and navigation within the parking system interface. Intuitive design elements, clear navigation paths, and informative visuals contribute to a user-friendly experience.

Reservation Process: Feedback on the reservation process highlights user preferences, pain points, and suggestions for improvement. Users appreciate the convenience of reserving parking spots in advance but may provide suggestions for streamlining the process or adding additional features (e.g., payment options, reminder notifications).

Visibility and Accessibility: Users express satisfaction with the visibility and accessibility of parking spot information, including real-time availability updates and interactive maps. Clear visual indicators and informative tooltips enhance users' ability to locate available parking spots efficiently.

Suggestions for Improvement: User feedback includes suggestions for improving specific features, enhancing system performance, and addressing usability issues. Common suggestions may include optimizing loading times, enhancing search functionality, improving mobile responsiveness, and adding additional language support.

CHAPTER 7

CONCLUSION

7. Conclusion

The smart car parking system developed for malls represents a significant advancement in parking management technology, offering innovative features and functionalities to enhance the parking experience for mall visitors. This section highlights the strengths of the system and acknowledges its limitations and challenges.

7.1 Strengths of the System

The smart car parking system boasts several strengths that contribute to its effectiveness, usability, and value proposition:

Real-Time Parking Spot Detection: The integration of computer vision algorithms enables real-time detection of parking spot occupancy, providing users with up-todate information on available parking spots.

User-Friendly Interface: The intuitive frontend interface, coupled with interactive maps and personalized recommendations, offers a user-friendly experience that simplifies parking navigation and reservation processes.

Scalability and Reliability: Leveraging cloud-based infrastructure ensures scalability and reliability, allowing the system to accommodate varying user loads and maintain high availability.

Efficient Reservation System: The reservation feature enables users to book parking spots in advance, promoting convenience and reducing the hassle of finding parking during peak hours.

Continuous Improvement: A commitment to iterative design and user feedbackdriven enhancements ensures that the system evolves to meet the changing needs and preferences of mall visitors over time.

7.2 Limitations and Challenges

Despite its strengths, the smart car parking system faces several limitations and challenges that warrant consideration:

Accuracy of Detection: Achieving consistently high accuracy in parking spot detection, especially under challenging environmental conditions such as low lighting or occlusions, remains a ongoing challenge.

User Adoption: Encouraging widespread adoption of the system among mall visitors may require targeted marketing efforts, user education initiatives, and incentives to overcome inertia and habituation to traditional parking methods.

Integration Complexity: Integrating the system with existing infrastructure, including mall management systems, parking gates, and payment systems, may pose technical and logistical challenges that require careful planning and coordination.

Data Privacy and Security: Safeguarding user data and ensuring compliance with privacy regulations are paramount concerns, requiring robust security measures and adherence to best practices in data handling and storage.

Maintenance and Support: Providing timely maintenance, technical support, and system updates to address issues and ensure optimal performance over the system's lifecycle requires dedicated resources and ongoing investment.

CHAPTER 8

FUTURE ENHANCEMENTS

8. Future Enhancements

The smart car parking system developed for malls represents a significant step forward in parking management technology. However, there are several areas where future enhancements and refinements could further improve the system's functionality, usability, and overall value proposition. This section outlines potential avenues for future development and improvement:

1. Enhanced Detection Accuracy: Continuously refining and optimizing the parking spot detection algorithm to achieve higher accuracy and robustness in various environmental conditions. Exploring advanced computer vision techniques, such as deep learning architectures and multi-sensor fusion, could lead to significant improvements in detection performance.

2. Predictive Analytics: Integrating predictive analytics capabilities to anticipate parking demand patterns and optimize resource allocation. By analyzing historical parking data, traffic flow patterns, and event schedules, the system could proactively recommend optimal parking spots and anticipate congestion areas, enhancing overall efficiency and user satisfaction.

3. Intelligent Pricing Models: Implementing dynamic pricing models based on realtime demand and supply dynamics to optimize parking revenue and utilization. Adaptive pricing strategies, including surge pricing during peak hours and discounts for off-peak periods, could incentivize efficient use of parking resources while maximizing revenue generation.

4. Seamless Integration with Mobility Services: Establishing seamless integration with mobility services such as ride-sharing platforms, public transit systems, and micro-mobility solutions. Providing users with integrated trip planning capabilities and last-mile connectivity options could enhance the overall mobility experience and reduce reliance on private vehicle ownership.

5. Accessibility Features: Incorporating accessibility features and accommodations to cater to the needs of users with disabilities or mobility impairments. Implementing features such as designated accessible parking spots, wheelchairfriendly navigation paths, and voice-activated interfaces can promote inclusivity and ensure equal access to parking facilities.

6. Sustainability Initiatives: Introducing sustainability initiatives and eco-friendly practices to promote environmental stewardship and reduce carbon emissions associated with parking. Implementing features such as electric vehicle (EV) charging stations, priority parking for low-emission vehicles, and incentives for carpooling and alternative transportation modes can contribute to a greener and more sustainable parking ecosystem.

7. Enhanced User Engagement: Developing innovative user engagement strategies, such as gamification elements, loyalty programs, and community-driven features, to foster long-term user engagement and loyalty. Encouraging user-generated content, reviews, and recommendations can create a sense of community and enhance the overall user experience.

8. Augmented Reality (AR) Integration: Exploring the integration of augmented reality (AR) technology to provide immersive and interactive parking navigation experiences. AR overlays could superimpose real-time parking spot information, navigation cues, and directions onto users' smartphone screens, enhancing situational awareness and navigation precision.

9. Continuous Feedback Mechanisms: Establishing continuous feedback mechanisms, such as user surveys, feedback forms, and sentiment analysis, to gather insights into user preferences, pain points, and suggestions for improvement. Leveraging user feedback to inform iterative design iterations ensures that the system evolves in alignment with user needs and expectations.

10. Expansion to Other Verticals: Considering the expansion of the smart car parking system beyond malls to other verticals such as airports, stadiums, commercial complexes, and urban centers. Tailoring the system's features and functionalities to suit the unique requirements of different environments and user demographics can unlock new opportunities for growth and adoption.

CHAPTER 9

APPENDICES

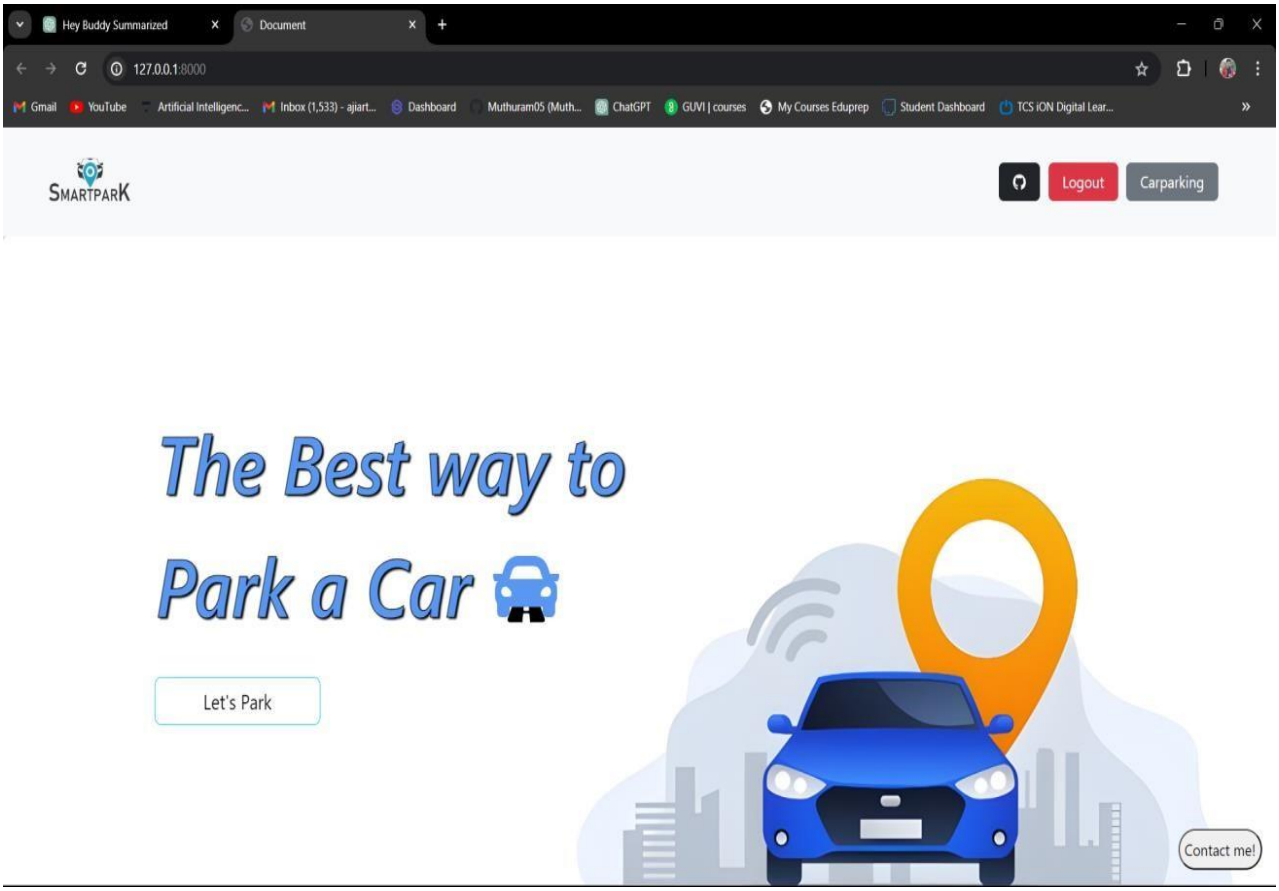

Fig 9.0 Home page

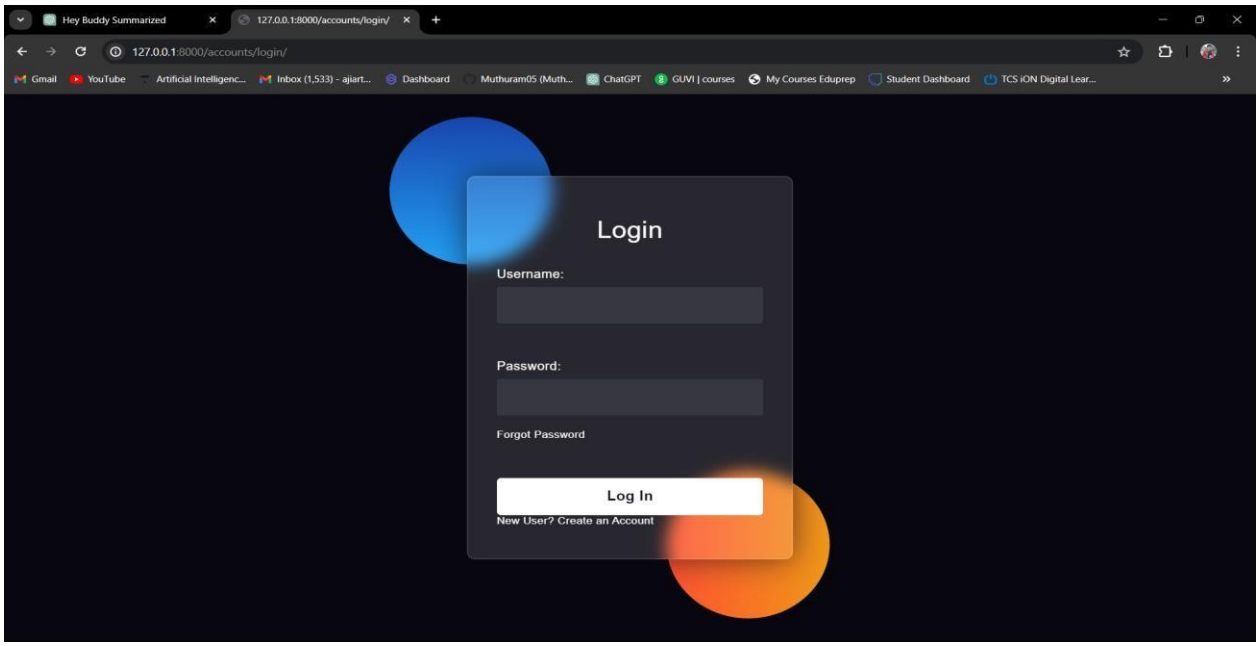

Fig 9.1 Login Page

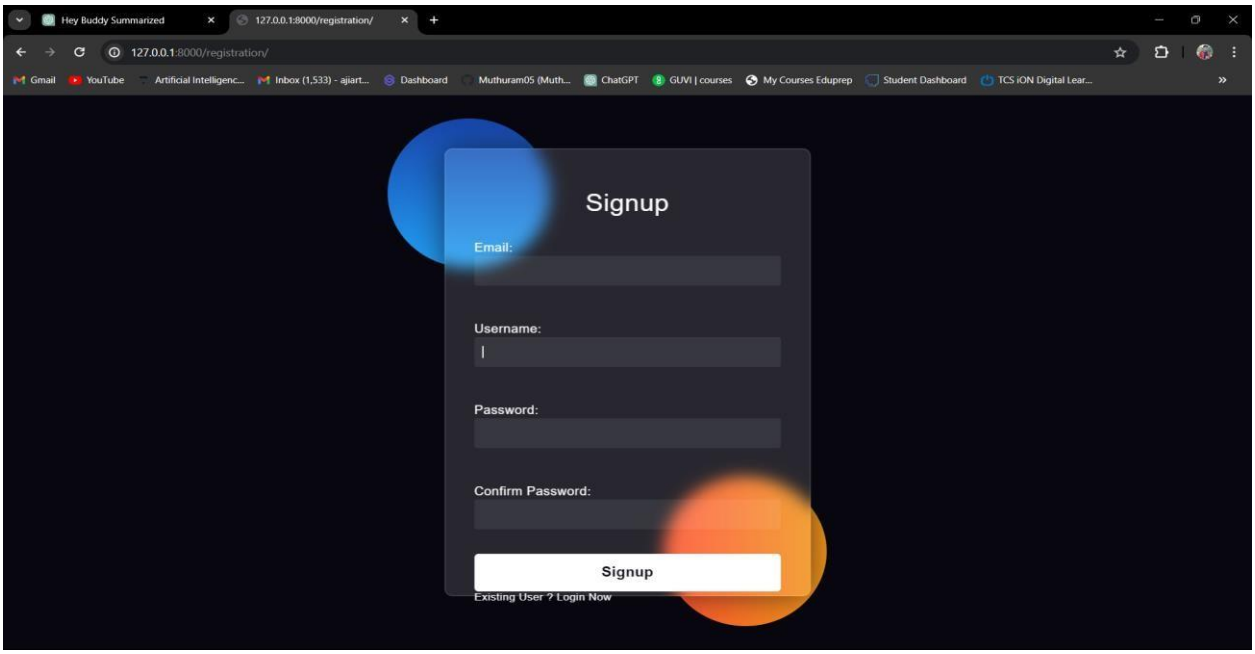

Fig 9.2 Signup Page

CHAPTER 10

REFERENCES

REFERENCES

[1] Ghaffarian, S., Karimi, H. A., & Su, H. N. (2019). A survey of smart parking solutions. ACM Computing Surveys (CSUR), 52(5), 1-35.

[2] Chien, S., & Ouyang, Y. (2018). Deep learning based real-time parking occupancy detection using spatialtemporal contextual information. Transportation Research Part C: Emerging Technologies, 86, 77-91.

[3] Luo, Y., & Liu, Z. (2019). An Internet of Things-based smart parking system. IEEE Transactions on Industrial Informatics, 16(6), 4376-4383.

[4] Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. arXiv preprint arXiv:1804.02767.

[5] Hossain, M. A., & Rahman, M. M. (2019). Parking lot management and vehicle detection system using IoT. International Journal of Scientific & Technology Research, 8(5), 412-417.

[6] Du, Y., & Wang, Y. (2020). Real-time parking space detection based on deep learning. In 2020 7th International Conference on Systems and Informatics (ICSAI) (pp. 522- 526). IEEE.

[7] Chen, C. P., & Hsu, Y. W. (2017). Development of intelligent parking management system with image processing. In 2017 IEEE International Conference on Applied System Invention (ICASI) (pp. 2323-2326). IEEE. Cortes, J., & Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273297.

[8] OpenCV Library. (n.d.). OpenCV: Open Source Computer Vision Library. Retrieved from https://opencv.org/

[9] Django Software Foundation. (n.d.). Django: The Web framework for perfectionists with deadlines. Retrieved from https://www.djangoproject.com/

[10] Zhang, S., Xu, C., & Hu, L. (2017). A real-time parking space detection system based on deep learning. In 2017 IEEE 7th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER) (pp. 340-345). IEEE.

[11] Wang, T., Zhang, S., & Fu, Y. (2018). A real-time parking occupancy detection system using deep learning. In 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON) (pp. 1-6). IEEE.

[12] Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., & Torralba, A. (2016). Learning deep features for discriminative localization. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2921-2929).

[13] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

[14] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Berg, A. C. (2015). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, 115(3), 211-252.

[15] Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, Inception-ResNet and the impact of residual connections on learning. In ThirtyFirst AAAI Conference on Artificial Intelligence.

[16] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[17] Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards realtime object detection with region proposal networks. IEEE transactions on pattern analysis and machine intelligence, 39(6), 1137- 1149.

[18] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[19] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In European conference on computer vision (pp. 740- 755). Springer, Cham.

[20] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., & Reed, S. (2016). SSD: Single shot multibox detector. In European conference on computer vision (pp. 21-37). Springer, Cham.

[21] Yu, F., Wang, D., Shelhamer, E., & Darrell, T. (2016). Deep layer aggregation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2403-2412).

[22] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

[23] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In Proceedings of the IEEE international conference on computer vision (pp. 2980-2988).

[24] Lin, T. Y., Dollar, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125).

[25] Papandreou, G., Zhu, T., Chen, L. C., Gidaris, S., Tompson, J., & Murphy, K. (2018). PersonLab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. arXiv preprint arXiv:1803.08225.

[26] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. arXiv preprint arXiv:1612.08242.

[27] Redmon, J., & Farhadi, A. (2016). YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271).

[28] Redmon, J., & Farhadi, A. (2015). YOLO: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

[29] Redmon, J., & Farhadi, A. (2016). YOLO9000: better, faster, stronger. arXiv preprint arXiv:1612.08242.

[30] Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2018). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs.IEEE transactions on pattern analysis and machine intelligence, 40(4), 834-848.

[31] Dai, J., Li, Y., He, K., & Sun, J. (2016). R-FCN: Object detection via regionbased fully convolutional networks. In Advances in neural information processing systems (pp. 379-387).

[32] Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable convolutional networks. In Proceedings of the IEEE international conference on computer vision (pp. 764-773).

[33] Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).

[34] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

[35] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In Proceedings of the IEEE international conference on computer vision (pp. 2961- 2969).

[36] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE transactions on pattern analysis and machine intelligence, 37(9), 1904-1916.

[37] Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., ... & Murphy, K. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3296-3297).

[38] He, K., Zhang, X., Ren, S., & Sun, J. (2017). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[39] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2117-2125).