A

PROJECT REPORT

ON

Go Park

Submitted in partial fulfillment for the award of

Post Graduate Diploma in Advance Computing

(PG-DAC) from

INSTITUTE OF EMERGING TECHNOLOGIES

Authorized Training Centre

Under the Guidance of

Mrs. Sampada Tarare

BY

Ms. Siddhi Dilip Londhe	**240345920039**
Mr. Nitin Sharad Patil	**240345920052**
Ms. Gauri Rajendra Kakade	**240345920029**
Mr. Saud Ahmed Khawjabhai Kadapa	**240345920033**

CERTIFICATE

This is to certify that the project report entitled (**GO PARK**) is a bonfire work carried out by (**Siddhi Dilip Londhe, Nitin Sharad Patil, Gauri Rajendra Kakade, Saud Ahmed Khawjabhai Kadapa**) and submitted in partial fulfilment of the requirement for the C-DAC ACTS, DAC course in Institute of Emerging Technology in the batch of Mar 2024.

Course Coordinator **External Examiner**

ACKNOWLEDGEMENT

This project **Go-Park** was a great learning experience for us and we are submitting this work to Advanced Computing Training School (CDAC).

We are very glad to mention the **Mrs. Sampada Tarare** for her valuable guidance to work on this project. Her guidance and support helped us to overcome various obstacles and intricacies during the course of project work.

Our most heart full thanks goes to ***Mr. Sangram Patil*** **(Director ,IET)**who gave all the required support and kind coordination to provide all the necessities like required hardware , internet facility and extra lab hours to complete the project and throughout the course up to the last day here in C-DAC ACTS, Pune.

Name of Students	**PRN Number**	**Signature**
Ms. Siddhi Dillip Londhe	**240345920039**	Londhe
Mr. Nitin Sharad Patil	**240345920052**	[illegible]
Ms. Gauri Rajendra Kakade	**240345920029**	[illegible]
Mr. Saud Ahmed Khawjabhai Kadapa	**240345920033**	[illegible]

Abstract

The "Go-Park Application" is an innovative web-based platform designed to streamline parking slot management and booking processes for both complex users and tenant users within commercial complexes. This application is engineered to simplify the parking experience by offering real-time availability, seamless booking, and an integrated payment gateway. Users can effortlessly book and manage parking slots, while complex administrators can efficiently monitor and allocate spaces. The system's robust backend, powered by Spring Boot, ensures secure, fast, and scalable operations, while the React.js-based frontend delivers a user-friendly interface. With features like user management, payment processing, and booking history tracking, the Go-Park Application not only enhances user convenience but also optimizes space utilization, reduces operational overhead, and fosters a more organized and efficient parking environment within commercial complexes

Index

1. INTRODUCTION

In today's rapidly urbanizing world, efficient management of parking spaces has become a critical challenge for commercial complexes. As more businesses and residents converge in these spaces, the demand for well-organized parking solutions has never been higher. This project aims to design and develop a comprehensive web-based application, Go-Park, that addresses this need by providing a robust parking slots management and booking system for complex users and daily visitors.

Leveraging cutting-edge technologies like React.js for the frontend, Spring Boot for the backend, and MySQL for the database, Go-Park ensures a seamless and user-friendly experience. The application is designed to optimize the use of parking spaces, reducing congestion and maximizing efficiency. Users, whether they are daily visitors or occasional guests, can easily book parking slots through an intuitive interface, with real-time updates on availability.
The backend architecture, powered by Spring Boot, facilitates robust RESTful APIs that handle data fetching, user authentication, and overall application processing. This ensures that the system remains scalable and secure, capable of handling a high volume of transactions and user interactions. Additionally, the integration of a payment gateway provides a secure and convenient method for users to complete their booking transactions, further enhancing the user experience.

Go-Park is more than just a parking management tool; it is a solution designed to bring order and efficiency to complex environments where space is at a premium. By implementing this system, commercial complexes can offer a value-added service to their visitors, ensuring smooth operations and increased satisfaction. This project not only addresses the immediate needs of parking management but also sets the foundation for future expansions and integrations, making it a forward-thinking solution in the realm of smart urban planning.

2. Problem Definition & Scope

Urban areas and large commercial spaces are increasingly struggling with managing vehicle parking efficiently. Traditional parking systems are often plagued by issues such as:

1. Lack of Real-time Information: Users face difficulties in knowing whether parking slots are available before they reach the parking area.
2. Manual Processes: The reliance on manual bookings, ticketing, and payments leads to delays and errors.
3. Inefficient Slot Management: Admins face challenges in tracking available, booked, or reserved slots, leading to suboptimal utilization of parking space.
4. User Experience Challenges: Users often deal with confusion, long queues, and a lack of a streamlined interface to easily book and manage parking.

Project Scope

The Go Park Parking Management System is designed to manage parking operations in a seamless and efficient manner, offering the following features:

1. User Module:

- **Real-time Slot Availability Check:** Users can check the availability of parking slots in real-time.
- **Slot Booking:** Users can book parking slots in advance through a user-friendly interface.
- **Booking History:** Users can view their past and current bookings for better tracking.
- **Slot Cancellation:** Users can cancel their booking if required.

2. Admin Module:

- **Slot Management:** Admins can add, update, or remove parking slots.
- **Update Availability:** Admins can manage slot statuses (available, reserved, booked).
- **Reporting:** Generate reports on slot usage, bookings, and parking trends for data-driven decisions.
- **Pricing Management:** Admins can set and update pricing for different parking slots.

3. System Automation:

- **Slot Status Updates:** Automatically update slot status based on user actions (booking, cancellation).
- **Notification Management:** Send notifications to users about booking confirmation, reminders, or updates.

2.1 Problem Definition

In today's urban environments, parking management in commercial complexes has become increasingly challenging due to the growing number of vehicles and limited parking space availability. Traditional parking systems are often inefficient, leading to frustration among users and tenants, increased congestion, and loss of valuable time. To address these issues, there is a need for an intelligent, web-based parking management and booking system specifically designed for commercial complexes. This system, named "Go-Park," aims to streamline the parking process by providing an easy-to-use platform for complex users and tenant users to reserve parking slots in advance.

2.2 Goals & Objectives

Goals:-

To design and develop a robust, user-friendly, and efficient web-based application, named **Go-Park**, for managing parking slots in a commercial complex. This application aims to streamline the parking slot booking process for both complex users and tenant users, reduce manual errors, and integrate a secure payment gateway for hassle-free transactions

Objectives:-

The primary objective of the **Go-Park** application is to provide a seamless and intuitive interface for users to book parking slots, whether they are regular complex users or tenant users. Utilizing React.js for the frontend, the application will offer a responsive and visually appealing user experience. The backend, powered by Spring Boot, will handle RESTful API calls for fetching data, user authentication, and processing booking requests. The integration with a MySQL database will ensure secure and efficient data management, while the payment gateway will enable users to complete transactions swiftly and securely. This project not only aims to enhance the user experience but also to optimize parking space utilization and reduce administrative overhead for the commercial complex management.

2.3 Major Constraints& Outcomes

- Major Constraints

1. **Scalability:**
- The application should handle a large number of users simultaneously, including complex users and tenant users, without performance degradation.
2. **Data Integrity**:
- Ensure that the booking system maintains accurate records, preventing double bookings and handling cancellations or modifications effectively.
3. **Security**:
- Implement strong user authentication and authorization mechanisms.
- Protect sensitive user data, especially payment information, using encryption and secure communication protocols (e.g., HTTPS).
4. **Responsive Design**:
- The frontend should be accessible and functional across different devices and screen sizes, ensuring a consistent user experience.
5. **Real-Time Updates**:
- Provide real-time availability of parking slots to prevent booking conflicts.
- The system should reflect changes instantly across all user interfaces.
6. **System Reliability and Availability**:
- The system must be highly reliable with minimal downtime, especially during peak hours.
- Implement backup and recovery mechanisms to ensure data is not lost.

- Expected Outcomes

- **Efficient Parking Slot Management**:

  - The application will streamline the management of parking slots, making it easier for both complex users to find and book available slots.

- **Improved User Satisfaction**:

  - Users will experience a seamless and hassle-free booking process with real-time updates on slot availability, leading to higher satisfaction.

- **Reduced Booking Conflicts**:
  - The real-time system will minimize conflicts and errors in slot booking, ensuring smooth operations and reducing the chances of double bookings.
- **Enhanced Security**:
  - With strong security measures in place, users will feel confident that their data and payment information are safe, fostering trust in the application.
- **Revenue Generation**:
  - The integration of a payment gateway will enable the commercial complex to efficiently collect fees, potentially increasing revenue through easy and accessible payment options.
- **Scalable System**:
  - The application will be designed to grow with the complex, handling an increasing number of users, parking slots, and transactions without performance issues.
- **Comprehensive Reporting**:
  - The system will provide detailed reports on parking usage, user behavior, and financial transactions, aiding in decision-making and strategic planning.
- **Compliance with Regulations**:
  - By adhering to legal and regulatory requirements, the application will avoid legal complications and penalties, ensuring smooth and lawful operation.
- **Modern, Responsive Interface**:
  - The application will offer a modern and visually appealing interface that works well on various devices, enhancing user engagement and accessibility.
  - **Seamless Integration**: If needed, the application will integrate smoothly with other systems within the complex, creating a unified experience for users and administrator.

3. Software Requirement Specification (SRS)

1. Introduction

1.1 Purpose

The purpose of this document is to define the software requirements for the Go-Park Application, a web-based system designed to manage and book parking slots within a commercial complex. This document will serve as a guide for developers, testers, and stakeholders to ensure that the system meets the specified requirements.

3.1 Proposed System

The proposed system is a web-based application named "Go-Park Application," designed to streamline the management and booking of parking slots within a commercial complex. The system will cater to the needs of various stakeholders, including the complex administrators (admins), companies, company users, and tenant users. The application will provide a comprehensive solution for parking slot allocation, real-time availability tracking, booking management, and payment processing, all through an intuitive and user-friendly interface.

A .Key Features of the Proposed System

1. **User Management:**

- **Role-based Access Control**: The system will support different user roles (Admin, Company User) with specific permissions and access levels.
- **Secure Authentication**: Users will log in through a secure authentication process, with the option for two-factor authentication for enhanced security.
- **Profile Management**: Users will be able to manage their personal information and preferences within their profiles.

2. **Parking Slot Management:**

Slot Allocation and Configuration: Admins will be able to create, update, or delete parking slots and assign them to companies or user The system will allow the configuration of parking slot

- **Slot Allocation and Configuration**: Admins will be able to create, update, or delete parking slots and assign them to companies or user The system will allow the configuration of parking slot attributes such as availability, pricing, and duration.
- **Real-time Availability**: The system will provide real-time updates on parking slot availability, allowing users to quickly identify and reserve available slots.

3. **Booking Management:**

- **Slot Reservation**: Company users will be able to search for and reserve parking slots based on their specific needs (e.g., date, time, location).
- **Booking Modification and Cancellation**: Users can modify or cancel their bookings within the allowed time frame, based on the booking policies set by the admin.
- **Booking History**: The system will maintain a comprehensive history of all bookings made by users, which can be reviewed at any time.

4. **Admin Dashboard:**

- **Comprehensive Monitoring**: The dashboard will offer admins a real-time view of system performance, including parking slot occupancy, booking trends, and financial transactions.
- **Reporting Tools**: Admins will be able to generate detailed reports on various aspects of the system, such as booking statistics, payment summaries, and user activity.
- **System Notifications**: The dashboard will include notification capabilities to alert admins of important events, such as booking cancellations or system errors.

3.2 Scope

A. Functional Scope

The functional scope defines the core functionalities that the Go-Park Application will provide. These functionalities address the specific needs of managing and booking parking slots within a commercial complex.

1. **User Management**

- **Admin Management**: Admins can create, edit, and delete user accounts (Admin, Company User).
- **User Authentication**: Secure login and registration for users with role-based access control.
- **Profile Management**: Users can update their profile information, including contact details.

2. Parking Slot Management

- **Slot Allocation**: Admins can create, update, or delete parking slots. Slots can be allocated to specific companies or users.
- **Real-time Availability**: The system displays real-time availability of parking slots to users.
- **Slot Configuration**: Admins can configure the pricing and availability for different types of parking slots.

3. Booking Management

- **Slot Booking**: Company users and tenant users can search for available parking slots and make reservations.
- **Booking Modification**: Users can cancel or modify their bookings based on availability and booking policies.
- **Booking History**: The system maintains a history of bookings for each user.

4. Admin Dashboard

- **System Monitoring**: Admins can view real-time data on parking slot usage, booking trends.
- **Reporting**: The system generates reports on bookings, and system performance.
- **Notifications**: The system can send notifications to users regarding bookings, payments, and other important updates.

Non-Functional Scope

The non-functional scope defines the quality attributes, performance standards, and operational constraints of the Go-Park Application. These aspects ensure that the system is reliable, secure, and user-friendly.

1. Performance Requirements

- **Scalability**: The system should support concurrent access by up to 10,000 users without performance degradation.
- **Response Time**: The system should provide a response time of less than 2 seconds for most user interactions.
- **Throughput**: The system should be capable of processing up to 1,000 booking transactions per minute.

2. Security Requirements

- **Data Encryption**: Sensitive data, such as user passwords and payment information, should be encrypted both in transit and at rest.
- **Authentication & Authorization**: Implement robust authentication mechanisms, including two-factor authentication (2FA) for sensitive operations.
- **Data Privacy**: The system must comply with data protection regulations such as GDPR, ensuring user data privacy.

3. Usability Requirements

- **User Interface**: The application should have an intuitive and consistent user interface that is easy to navigate for all user types.
- **Accessibility**: The system should be accessible to users with disabilities, adhering to WCAG 2.1 standards.
- **Mobile Responsiveness**: The user interface should be responsive and functional on mobile devices, tablets, and desktops.

4. Reliability Requirements

- **Uptime**: The system should have a minimum uptime of 99.9% to ensure continuous availability.
- **Backup and Recovery**: Automated daily backups should be performed, with a disaster recovery plan in place to restore data within 4 hours in case of a failure.

5. Maintainability Requirements

- **Code Modularity**: The system should be built using modular code to allow for easy maintenance and updates.
- **Documentation**: Comprehensive documentation should be provided for the system, including user guides, API documentation, and technical manuals.

4. System Modules

1. User Management Module

- **Features:** Handles user registration, login, profile management, and authentication (e.g., admin, tenant, guest users).
- **Responsibilities:**
- User roles and permissions
- Profile updates and password management
- Authentication (e.g., via OAuth or JWT)

2. Parking Slot Management Module

- **Features:** Manages the available parking slots, including adding new slots, updating status, and monitoring availability.
- **Responsibilities:**
- Slot creation and categorization (e.g., by levels, sections)
- Availability status (booked, available, blocked)
- Real-time updates of slot status

3. Booking Management Module

- **Features:** Handles the booking process, including selecting parking slots, scheduling, and payment integration.
- **Responsibilities:**
- Slot reservation (with time-based hold)
- Booking confirmation
- Slot marking and real-time updates after booking

4. Payment and Billing Module

- **Features:** Manages payment transactions and billing details.
- **Responsibilities:**
- Payment gateway integration
- Invoice generation and billing history
- Refunds and cancellations

5. Dashboard and Reporting Module

- **Features:** Provides an overview of system metrics for admins and detailed reports for user.

- **Responsibilities:**
- Real-time dashboard showing system usage)
- Customized reports for user segments (e.g., admin, User)

4. Performance-Requirements

Hardware Type	Minimum Requirement
Processor	2.66 GHz x86
Primary Memory	4 GB RAM
SecondaryMemory	20 GB HDD Space
Internet Connection	4 Mbps Stable Connection

Software Type	Minimum Requirement
Platform (OS)	Windows 10
Front End	HTML, CSS, Bootstrap , React , JavaScript
Back End	Spring , Springbot , MySQL
Design Tool	RSE , Drawio
Development Tool (IDE)	Eclipse , Visual Studio
Testing Tool	JUnit
Documentation Tool	Microsoft Office

5.1 H/W Requirements & S/W Requirements

1. Hardware Requirements (H/W)

Hardware requirements are depending on the scope of work in the Go-Park Project, how many numbers of users are covered, and what is the expected traffic. Some of the general hardware requirements are given as below:

A. Server-Side Hardware Requirements

These requirements apply to the server infrastructure needed to host the Go-Park Application.

1. Web Server:

- **Processor**: 4-core CPU (minimum), such as Intel Xeon or AMD EPYC
- **RAM**: 8 GB (minimum), 16 GB recommended for larger deployments
- **Storage**: 256 GB SSD (minimum), scalable based on the size of the database
- **Network**: 1 Gbps network interface card (NIC) for handling multiple requests
- **Operating System**: Linux (e.g., Ubuntu Server, CentOS) or Windows Server

2. Database Server:

- **Processor**: 8-core CPU (minimum), such as Intel Xeon or AMD EPYC
- **RAM**: 16 GB (minimum), 32 GB recommended for high-volume transactions
- **Storage**: 1 TB SSD (minimum), with RAID configuration for redundancy
- **Network**: 1 Gbps network interface card (NIC)
- **Operating System**: Linux (e.g., Ubuntu Server, CentOS) or Windows Server
- **Backup Storage**: External storage solution or cloud storage for regular backups

B. Client-Side Hardware Requirements

These specifications are for end-users accessing the Go-Park Application via a web browser.

- **Processor**: Dual-Core (e.g., Intel i3, AMD Ryzen 3) or higher
- **RAM**: 4 GB or higher
- **Storage**: 20 GB free space (for browser cache, temporary files)
- **Display**: 1280x720 resolution or higher
- **Network**: Stable internet connection with at least 5 Mbps download speed
- **Input Devices**: Keyboard and mouse (or touchscreen for mobile devices)

2. Software Requirements(S/W)

The software requirements cover both the server-side and client-side environments necessary to run and access the Go-Park Application.

Server-Side Software Requirements :

- **Operating System**:

- **Linux**: Ubuntu 20.04 LTS, CentOS 7 or higher
- **Windows Server**: Windows Server 2019 or higher

- **Web Server**: Apache HTTP Server 2.4 or Nginx 1.18

- **Application Server**:

- **Java**: JDK 11 or higher
- **Spring Boot**: Spring Boot 2.5 or higher

- **Database**: MySQL 8.0 or higher
- **Runtime Environment**: Java Runtime Environment (JRE) 11 or higher
- **API Management**: Spring Boot REST APIs
- **Security**: SSL/TLS certificates for HTTPS, configured firewalls (e.g., iptables)
- **Monitoring Tools**: Prometheus, Grafana for monitoring server health and application performance
- **Backup Software**: Rsync, Bacula, or any cloud-based backup service for automated backups

Client-Side Software Requirements :

- **Operating System**:

- **Windows**: Windows 10 or higher
- **macOS**: macOS Mojave or higher
- **Linux**: Ubuntu 18.04 or higher

- **Mobile OS**: Android 9.0 or higher, iOS 12 or higher

➤ **Web Browser**:

- **Google Chrome**: Version 88 or higher
- **Mozilla Firefox**: Version 85 or higher
- **Microsoft Edge**: Version 88 or higher
- **Safari**: Version 14 or higher

➤ **Additional Software**:

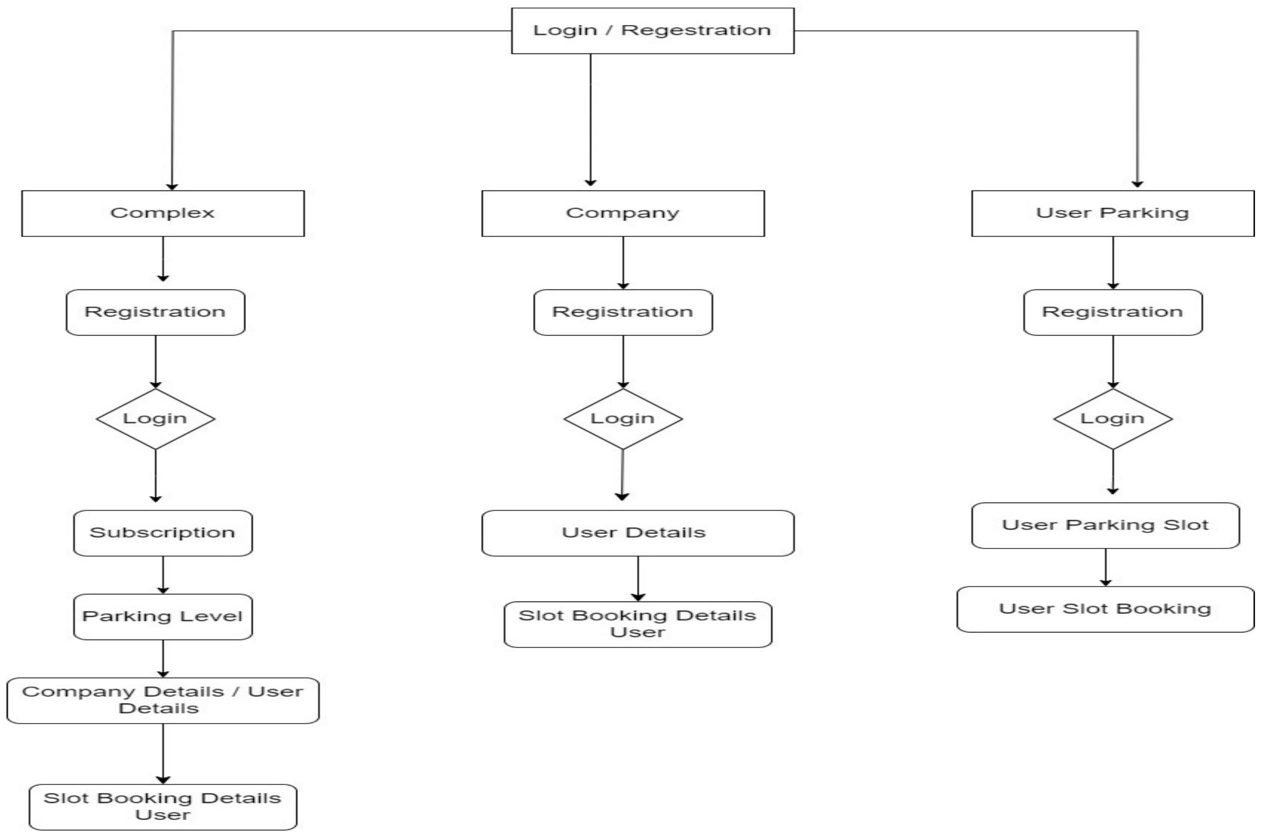
- **PDF Reader**: Adobe Acrobat Reader or any compatible PDF viewer for accessing reports and invoices.
- **Email Client**: Any standard email client (e.g., Outlook, Gmail) for receiving notifications and confirmat.

6. Unified Modeling Language (UML) Diagram

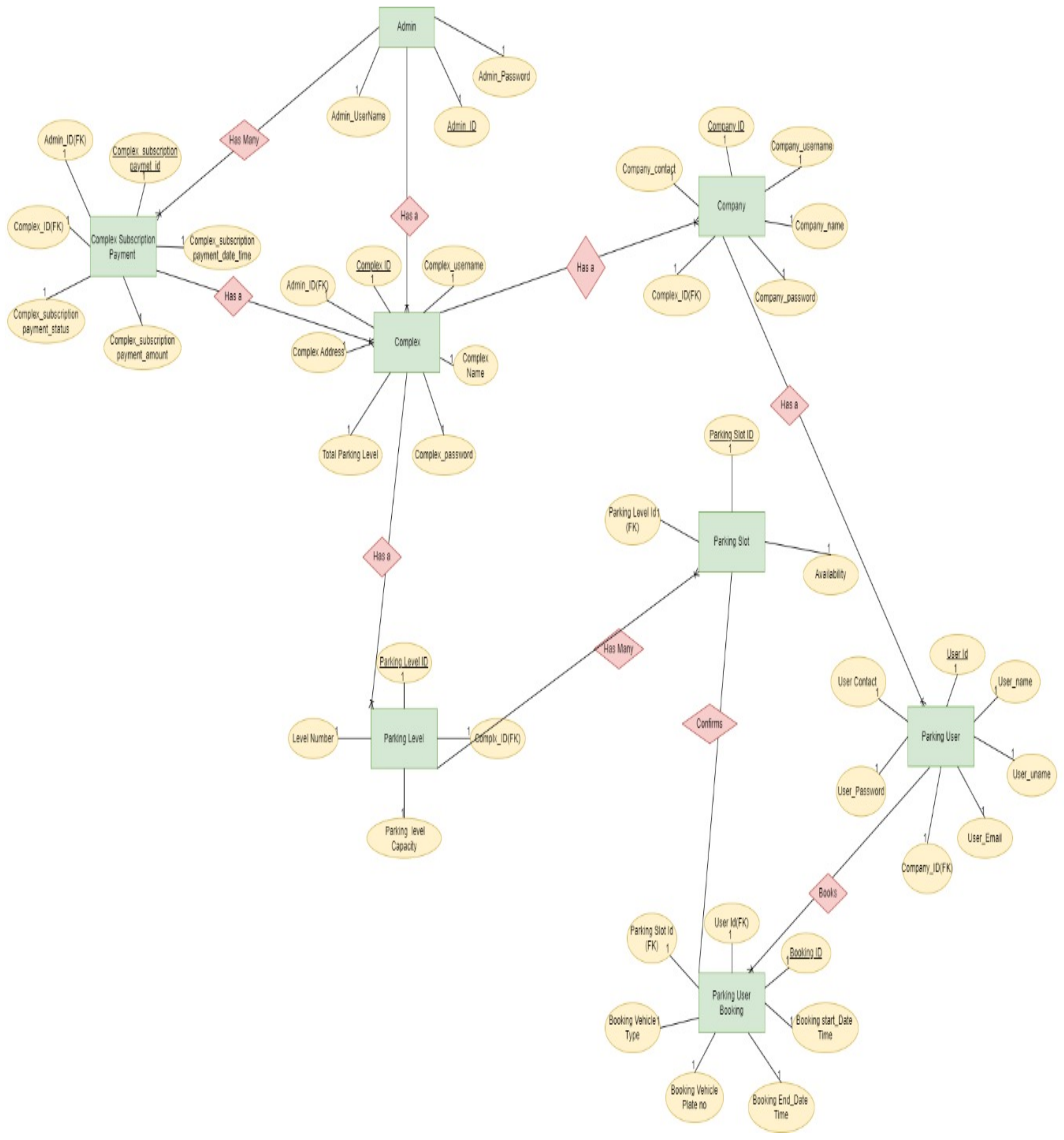
The unified modeling language (UML) is a Graphical Language for visualization, Specifying, construction and documenting the artifacts of a software intensive system. The UML gives a standard was to write system's blueprints, covering conceptual thing, such as Business Processes & system functions, as well as concrete things, such as classes written in a specific programming language, database schemas, and reusable software components.

6.1 Data Flow Diagram (DFD)

Data flow diagram (DFD), also referred to as 'Bubble chart' is a graphical technique, which is used to represent information flow, and transformers are applied when data moves from input to output.

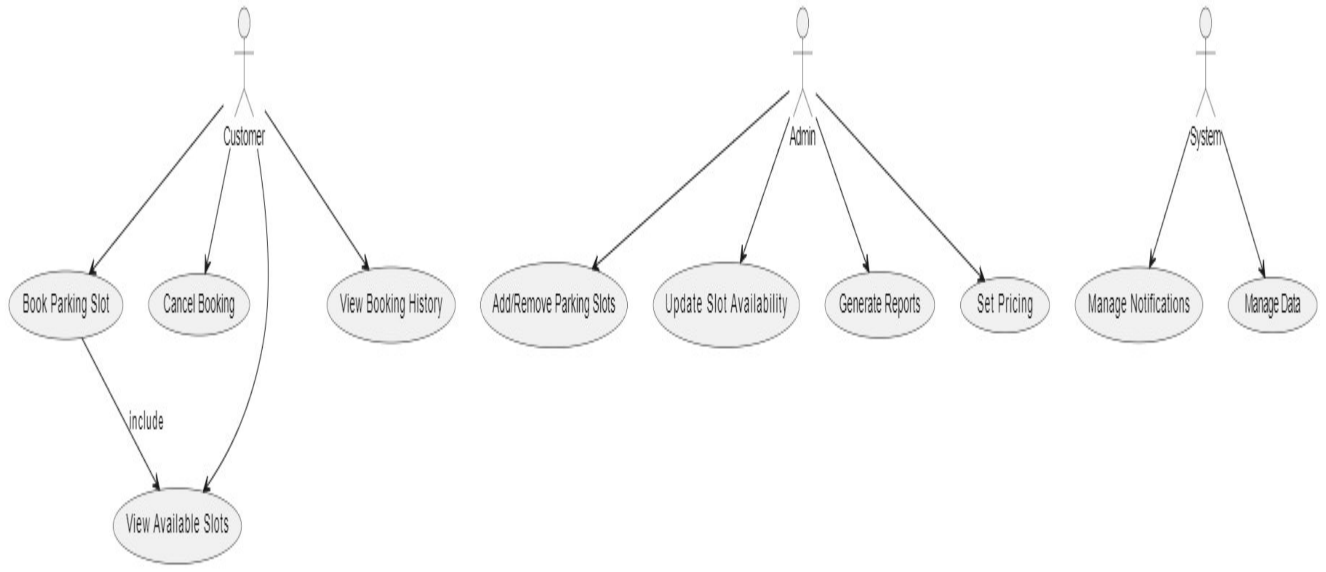


6.2 Entity-Relationship Diagram (ERD)



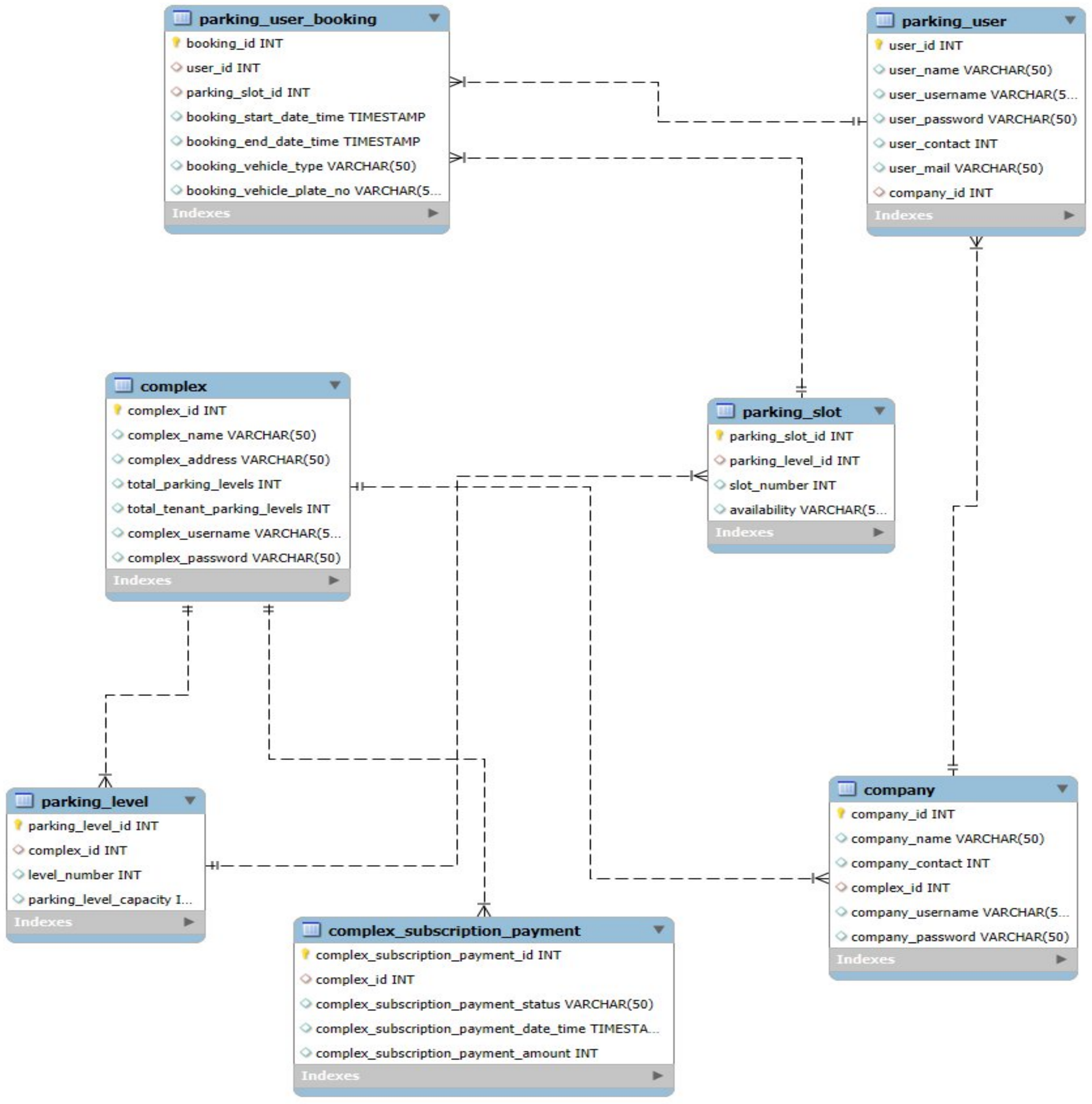
6.3 Use case Diagram

A use case defines behavioral features of a system. Each use case is named using a verb phase expresses a goal of the system. A use case diagram shows a set of use cases and actors &their relationships. Use case diagrams address the static use case view of a system. These diagrams are especially important in organizing and modeling the behaviors of a system. It shows the graphical overview of functionality provided by the system intents actor.



6.4 Class Diagram

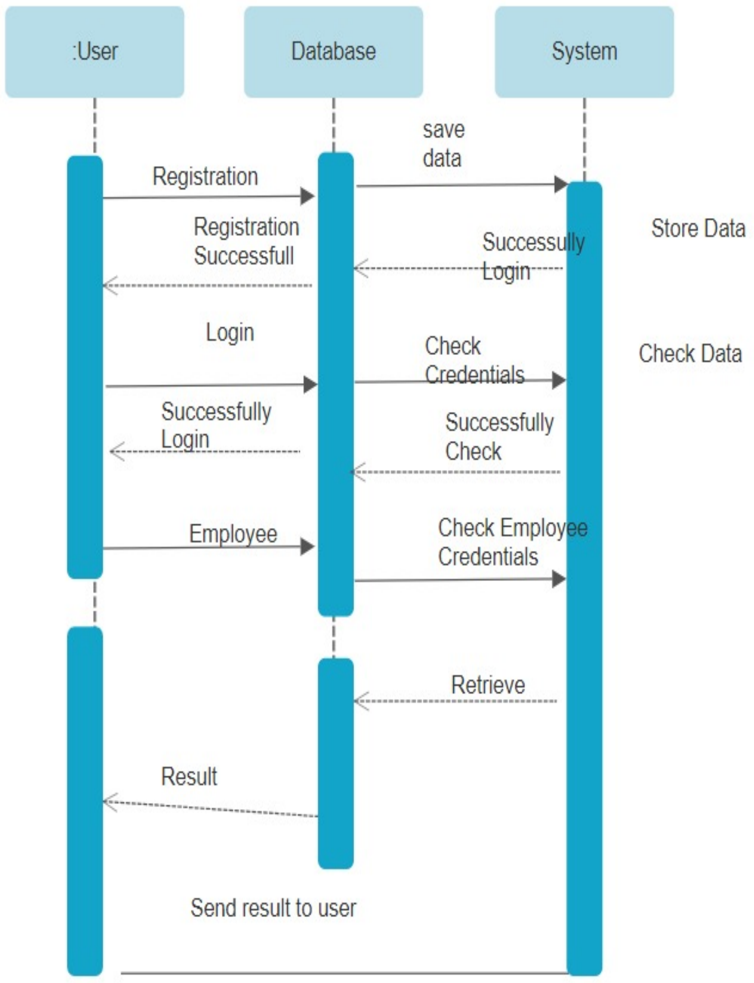
A class diagram shows a set of classes, interfaces and collaborations and their relationship. These diagrams are the most common diagram found in modeling object-oriented systems. Class diagram addressed the static design view of a system.



parking_user_booking
booking_id INT
user_id INT
parking_slot_id INT
booking_start_date_time TIMESTAMP
booking_end_date_time TIMESTAMP
booking_vehicle_type VARCHAR(50)
booking_vehicle_plate_no VARCHAR(5...
Indexes
parking_user
user_id INT
user_name VARCHAR(50)
user_username VARCHAR(5...
user_password VARCHAR(50)
user_contact INT
user_mail VARCHAR(50)
company_id INT
Indexes
complex
complex_id INT
complex_name VARCHAR(50)
complex_address VARCHAR(50)
total_parking_levels INT
total_tenant_parking_levels INT
complex_username VARCHAR(5...
complex_password VARCHAR(50)
Indexes
parking_slot
parking_slot_id INT
parking_level_id INT
slot_number INT
availability VARCHAR(5...
Indexes
parking_level
parking_level_id INT
complex_id INT
level_number INT
parking_level_capacity I...
Indexes
complex_subscription_payment
complex_subscription_payment_id INT
complex_id INT
complex_subscription_payment_status VARCHAR(50)
complex_subscription_payment_date_time TIMESTA...
complex_subscription_payment_amount INT
Indexes
company
company_id INT
company_name VARCHAR(50)
company_contact INT
complex_id INT
company_username VARCHAR(5...
company_password VARCHAR(50)
Indexes

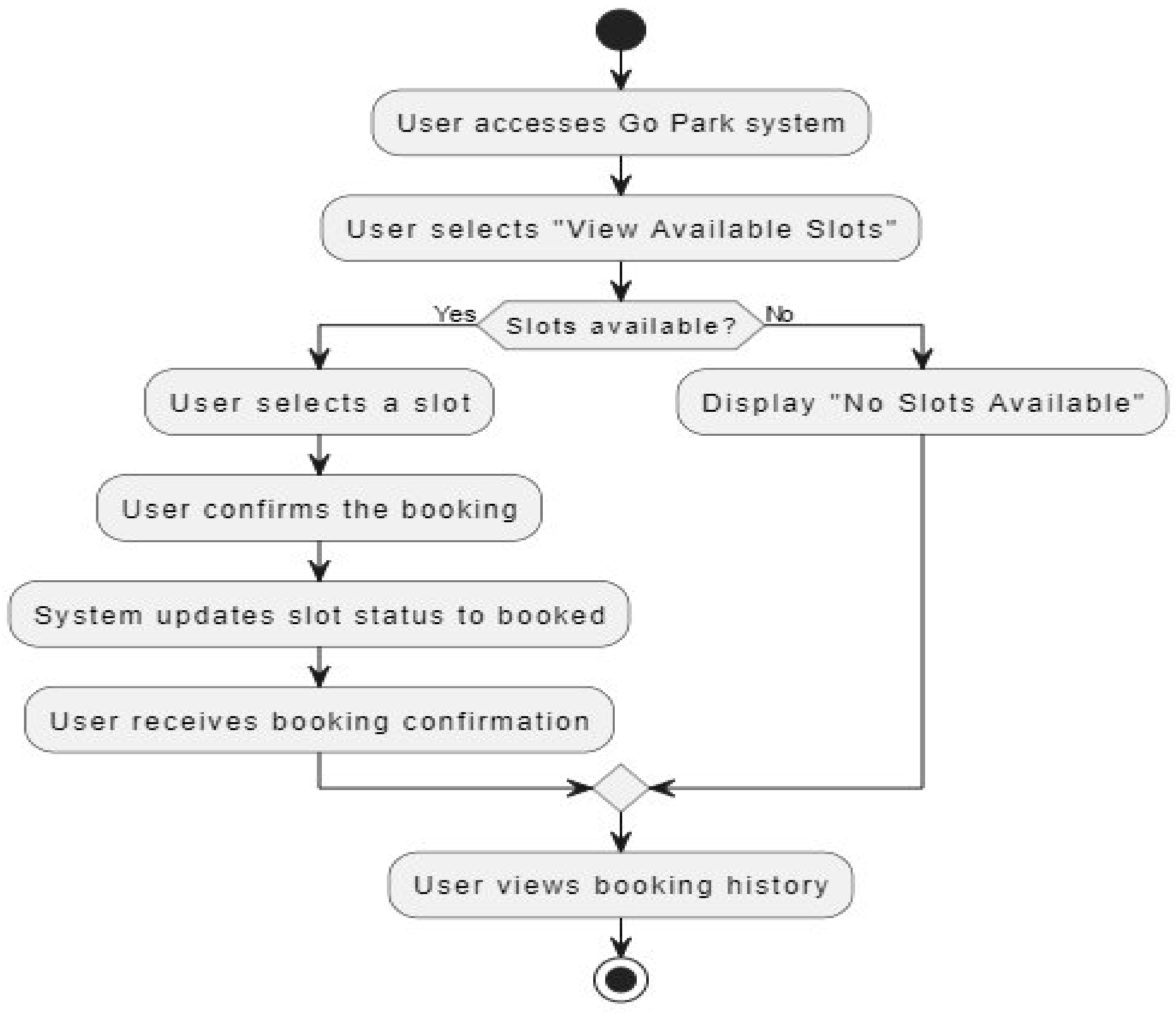
6.5 Sequence Diagram

Sequence diagram is a kind of interaction diagram. It shows an interaction, consisting of a set of objects and their relationships, including the message that may be dispatched among them. A sequence diagram emphasizes the time ordering of messages. As shown in the figure we can form a sequence diagram by first placing the objects that participate in the interaction at the top of our diagram. The object that initiates the interaction at the left and increasingly more subordinate objects to the right. The messages that these objects send and receive along the Y-axis, in order of increasing time from top to bottom. This gives the reader a clear visual cue to the flow of control over time.



6.6Activity Diagram

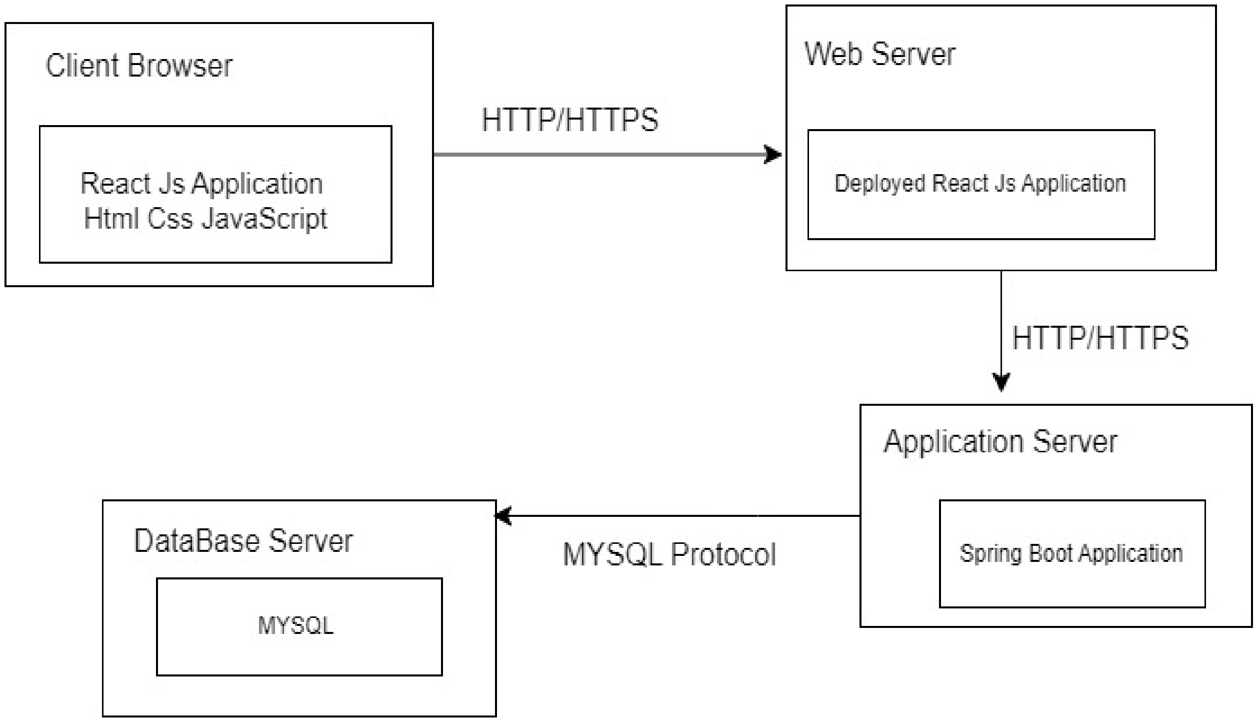
An activity diagram of a special kind of state chart diagram that shows the flow from activity within a system. An activity addresses the dynamic view of a system. The activity diagram is often seen as part of the functional view of a system because it describes logical processes, or functions. Each process describes a sequence of tasks and the decisions that govern when and when they are performed. The flow in an activity diagram is driven by the completion of an action.



User accesses Go Park system
User selects "View Available Slots"
Yes
Slots available?
No
User selects a slot
Display "No Slots Available"
User confirms the booking
System updates slot status to booked
User receives booking confirmation
User views booking history

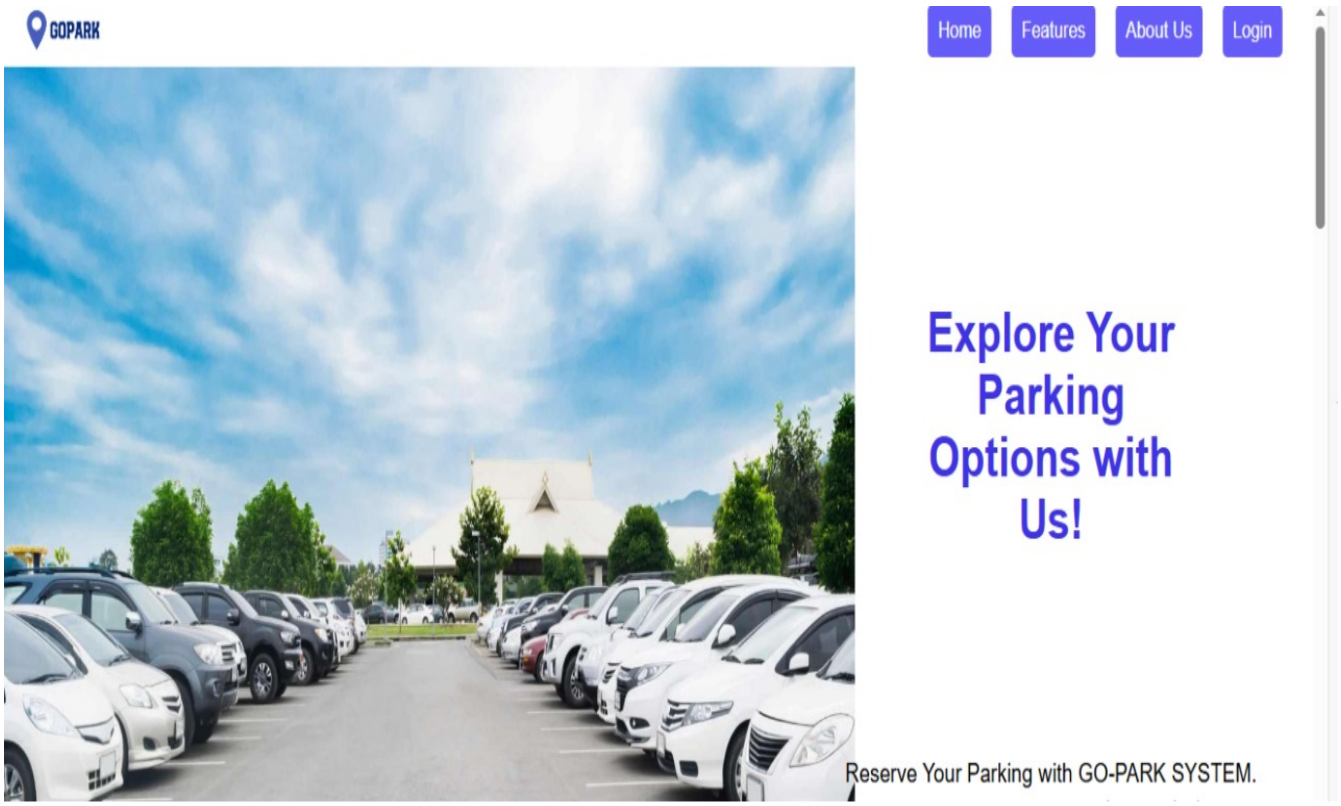
6.7 Deployment Diagram

Deployment diagram shows the configuration of run time processing nodes and components that live on them. Deployment diagram addresses the static deployment view of architecture. A deployment diagram shows the configuration of run-time processing nodes and the components that live on them. Deployment diagrams address the static view of architecture. They are related to components diagram in that a node typically encloses one or more components

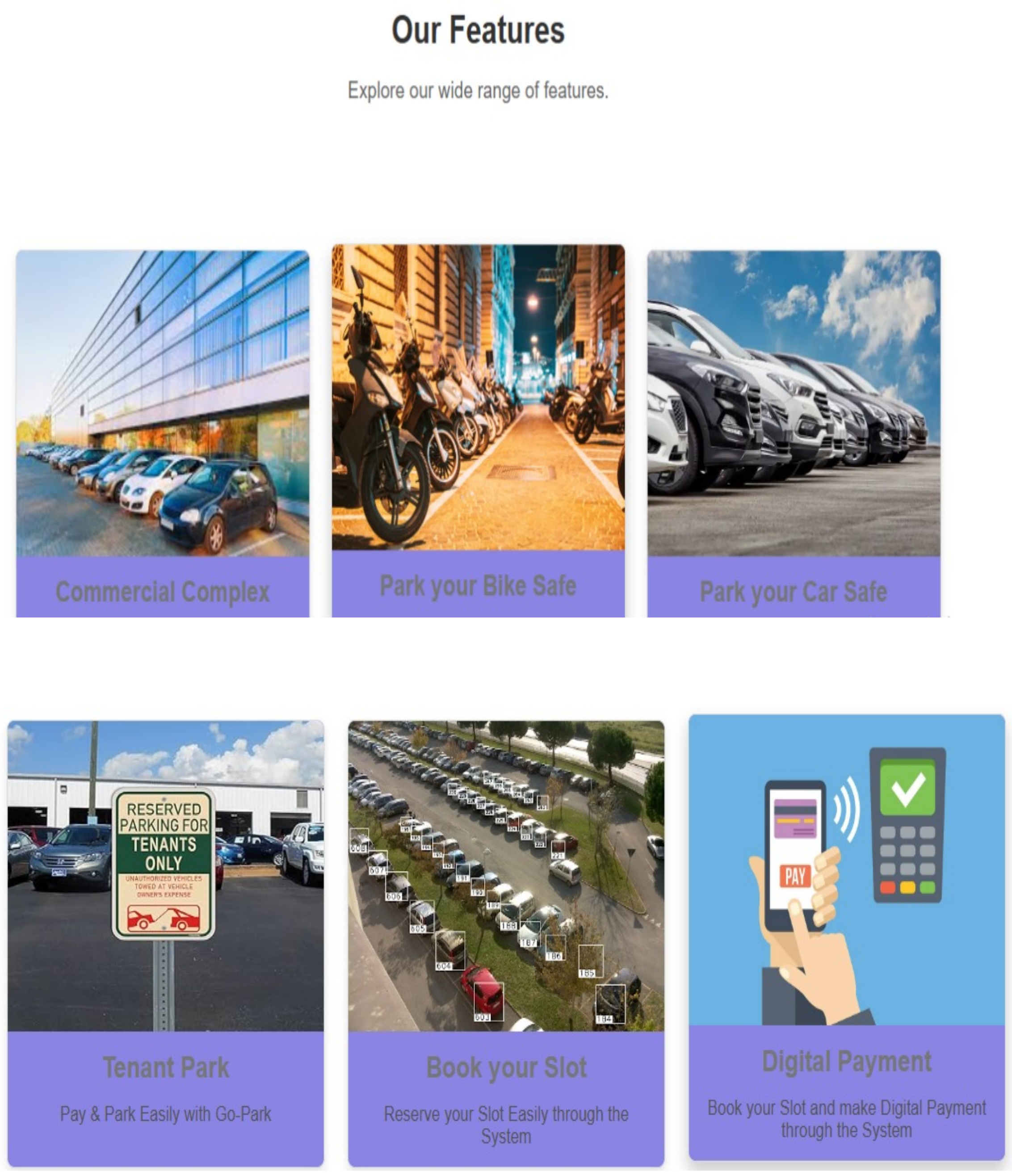


7. Screenshots

➢ Landing Page


GOPARK
Home
Features
About Us
Login
Explore Your Parking Options with Us!
Reserve Your Parking with GO-PARK SYSTEM.

➢ Features



Our Features

Explore our wide range of features.

Commercial Complex

Park your Bike Safe

Park your Car Safe

Tenant Park

Pay & Park Easily with Go-Park

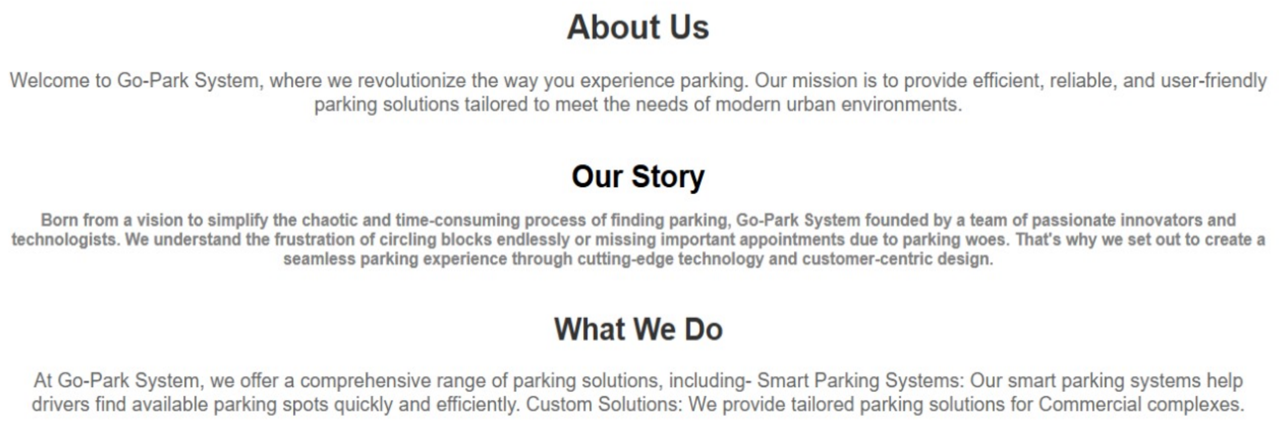
Book your Slot

Reserve your Slot Easily through the System

Digital Payment

Book your Slot and make Digital Payment through the System

➢ About


About Us

Welcome to Go-Park System, where we revolutionize the way you experience parking. Our mission is to provide efficient, reliable, and user-friendly parking solutions tailored to meet the needs of modern urban environments.

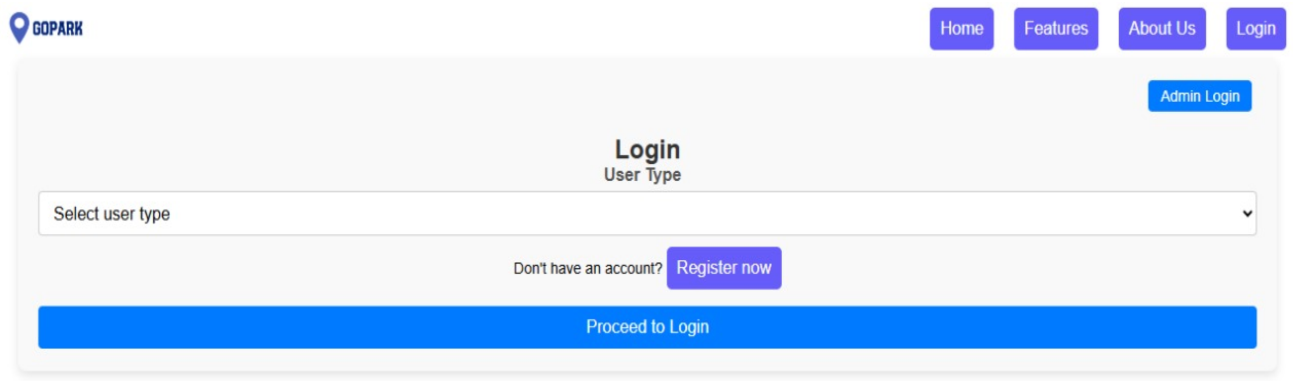
Our Story

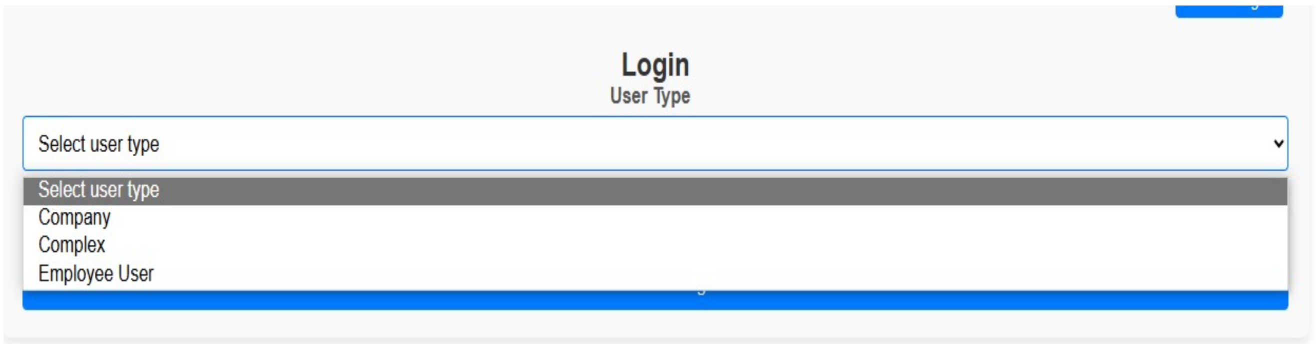
Born from a vision to simplify the chaotic and time-consuming process of finding parking, Go-Park System founded by a team of passionate innovators and technologists. We understand the frustration of circling blocks endlessly or missing important appointments due to parking woes. That's why we set out to create a seamless parking experience through cutting-edge technology and customer-centric design.

What We Do

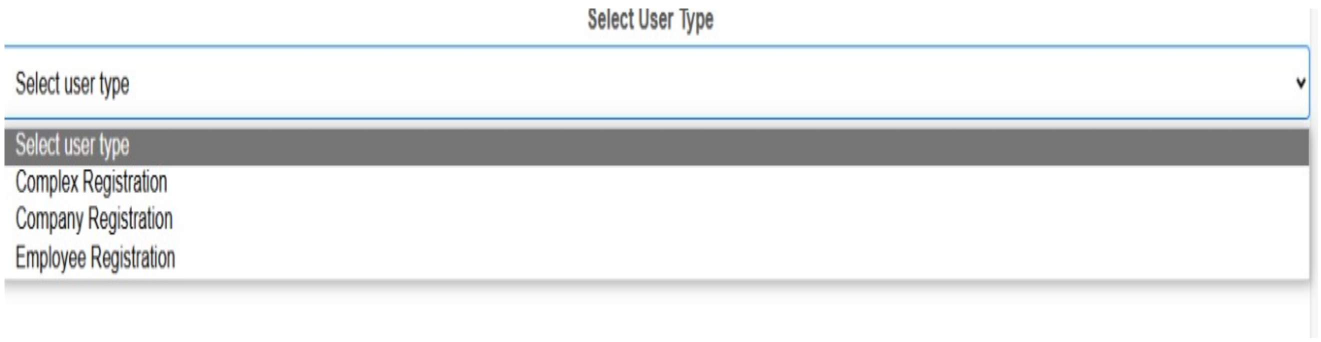
At Go-Park System, we offer a comprehensive range of parking solutions, including- Smart Parking Systems: Our smart parking systems help drivers find available parking spots quickly and efficiently. Custom Solutions: We provide tailored parking solutions for Commercial complexes.

➢ Login





➢ Registration

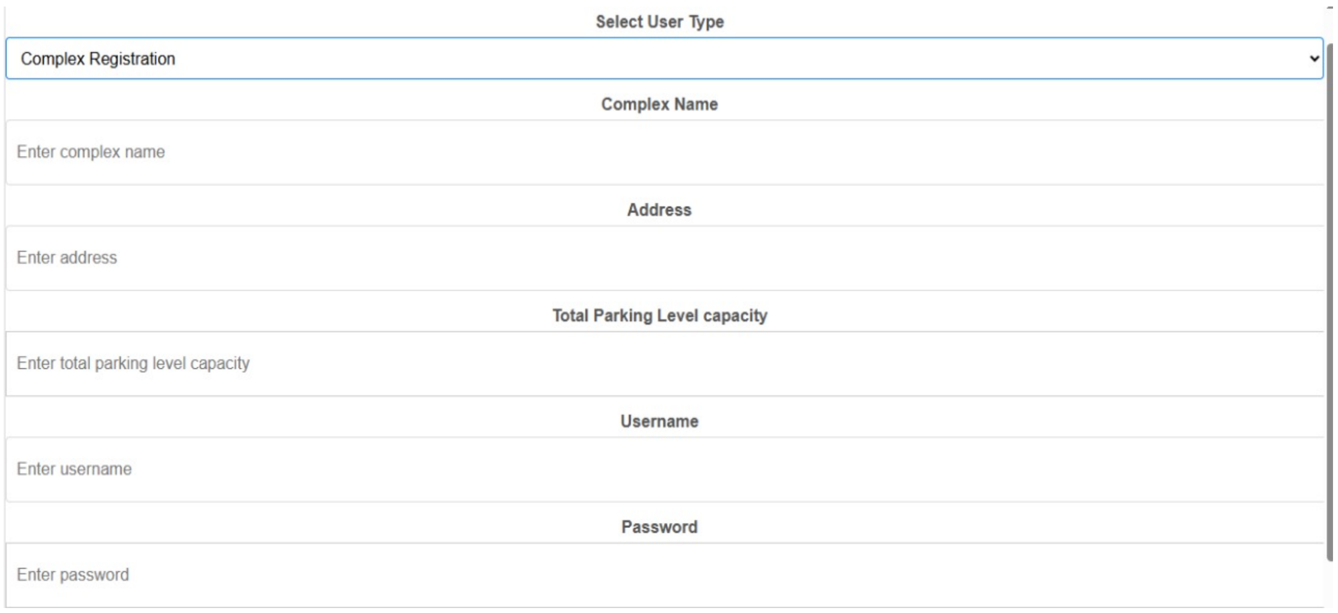


Select User Type

Select user type

Select user type
Complex Registration
Company Registration
Employee Registration

1 . Complex Registration



Select User Type

Complex Registration

Complex Name

Enter complex name

Address

Enter address

Total Parking Level capacity

Enter total parking level capacity

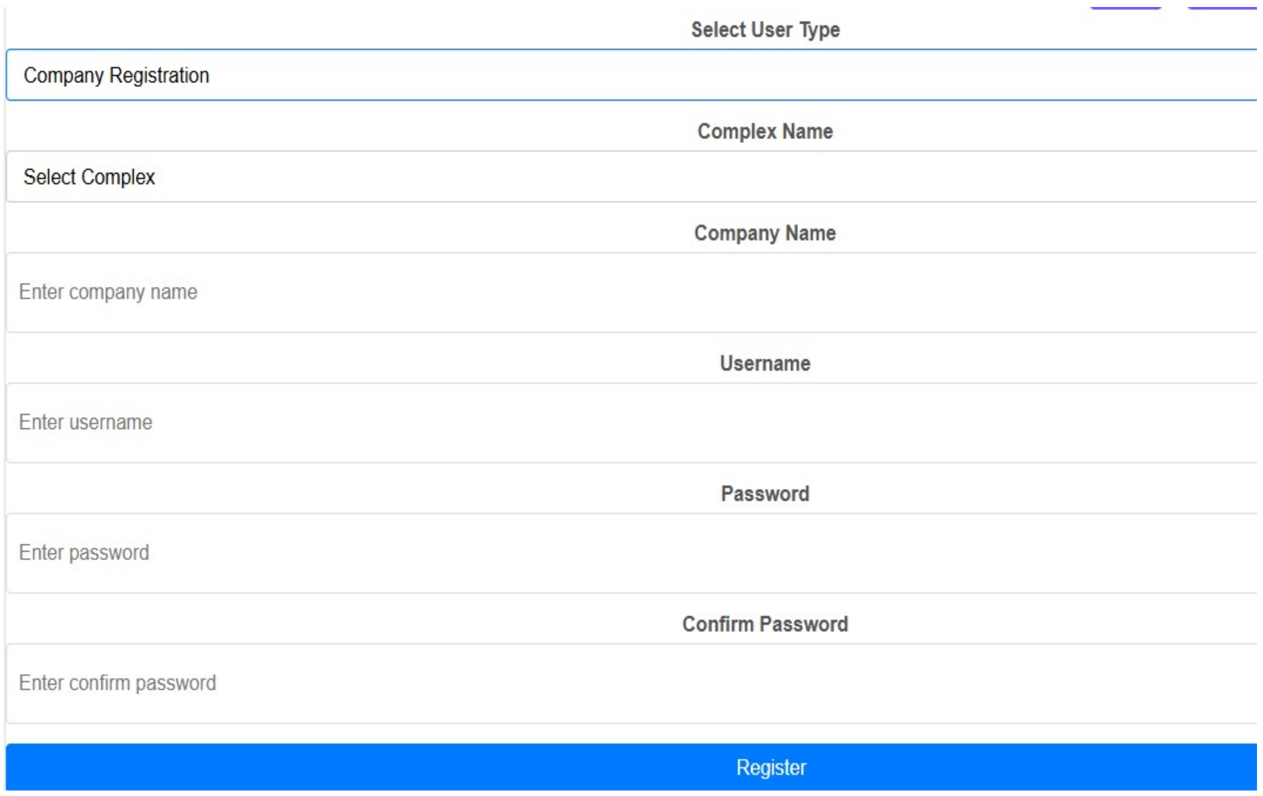
Username

Enter username

Password

Enter password

2. Company Registration



Select User Type

Company Registration

Complex Name

Select Complex

Company Name

Enter company name

Username

Enter username

Password

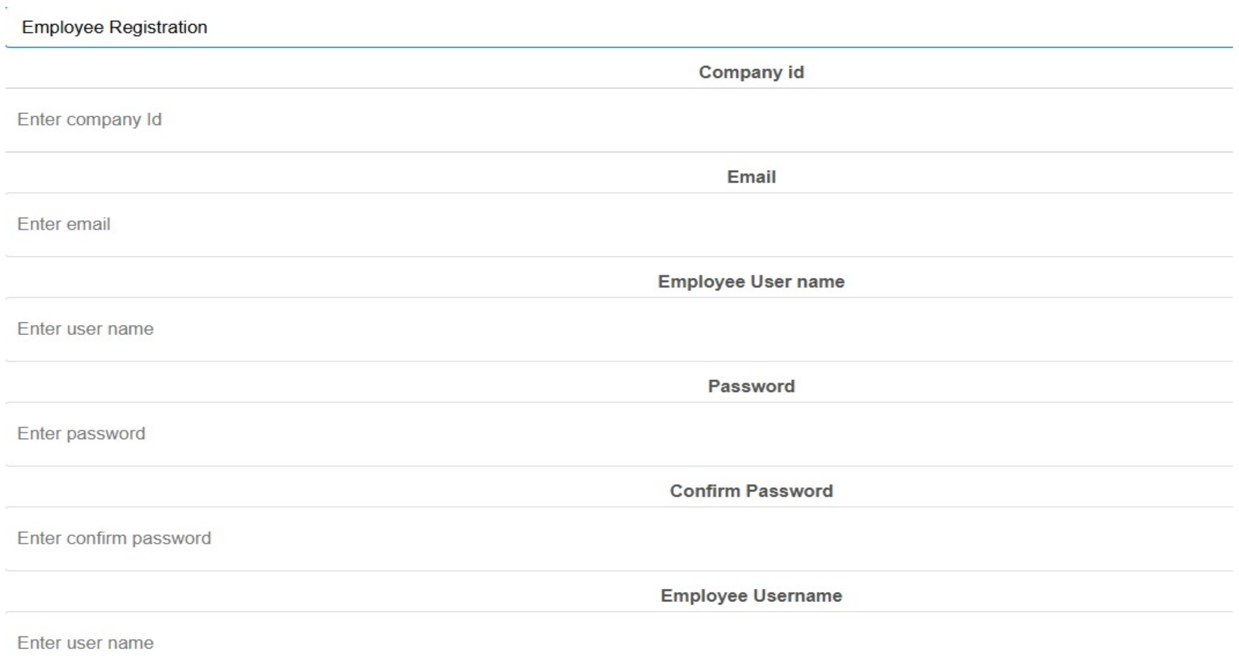
Enter password

Confirm Password

Enter confirm password

Register

3. Employee Registration



Employee Registration

Company id

Enter company Id

Email

Enter email

Employee User name

Enter user name

Password

Enter password

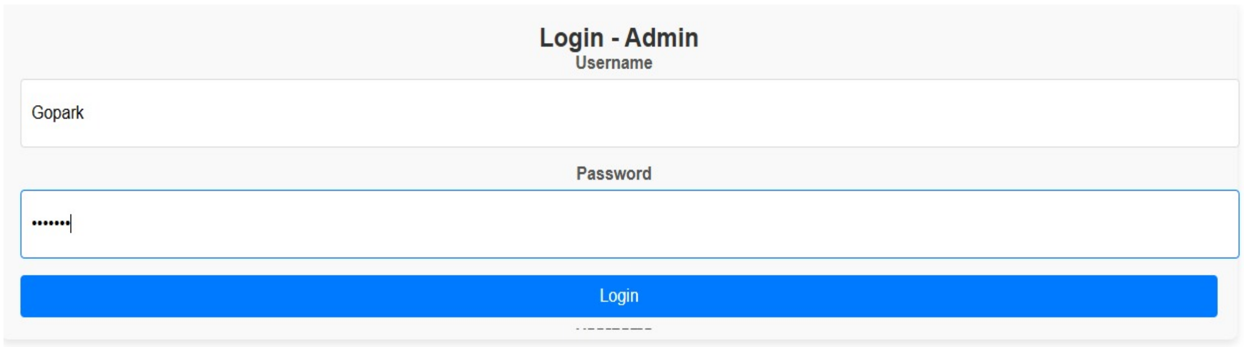
Confirm Password

Enter confirm password

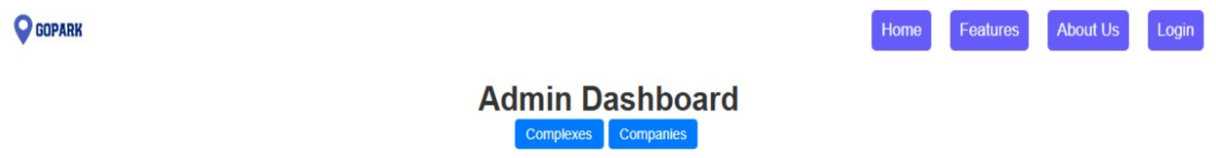
Employee Username

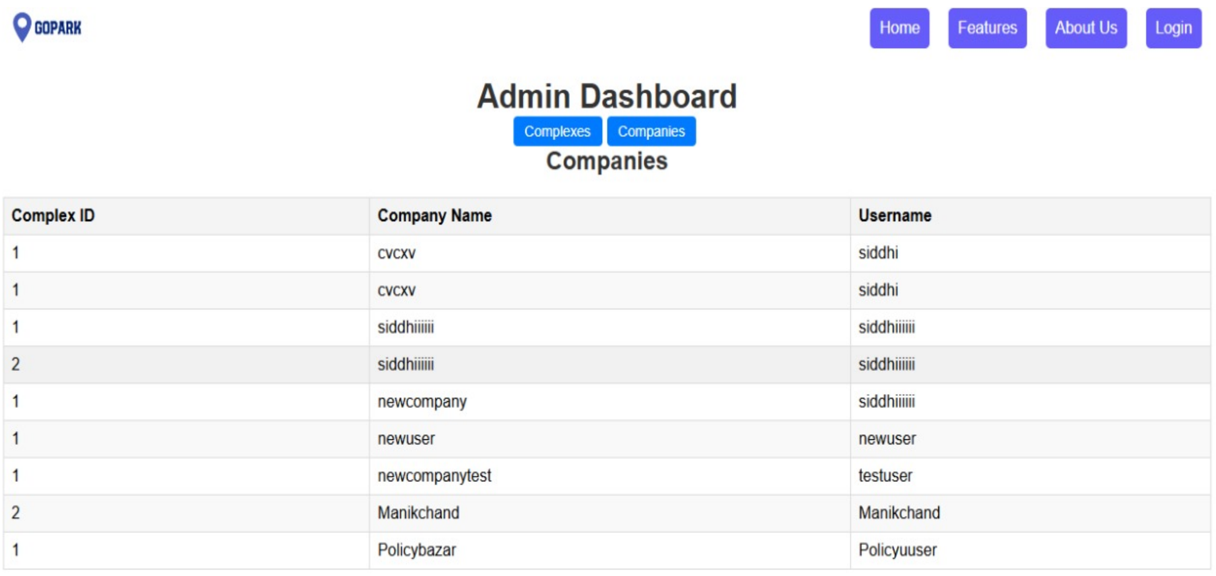
Enter user name

➢ Admin Login



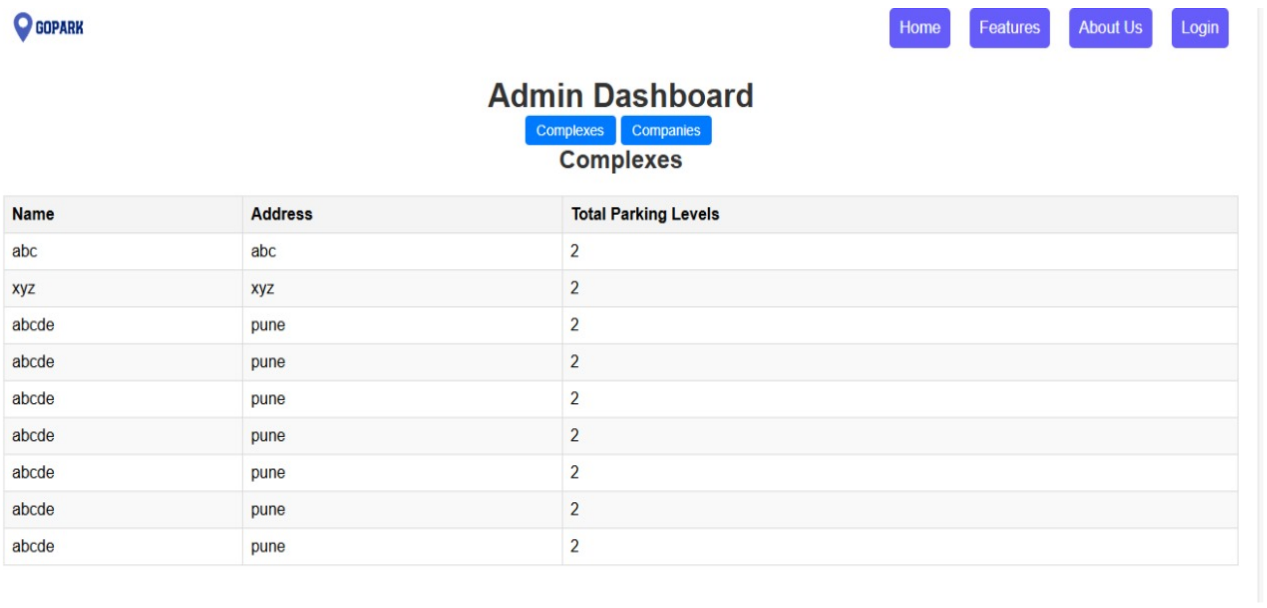
➢ Admin Dashboard





Companies

Complex ID	Company Name	Username
1	cvcxv	siddhi
1	cvcxv	siddhi
1	siddhiiiiii	siddhiiiiii
2	siddhiiiiii	siddhiiiiii
1	newcompany	siddhiiiiii
1	newuser	newuser
1	newcompanytest	testuser
2	Manikchand	Manikchand
1	Policybazar	Policyuuser



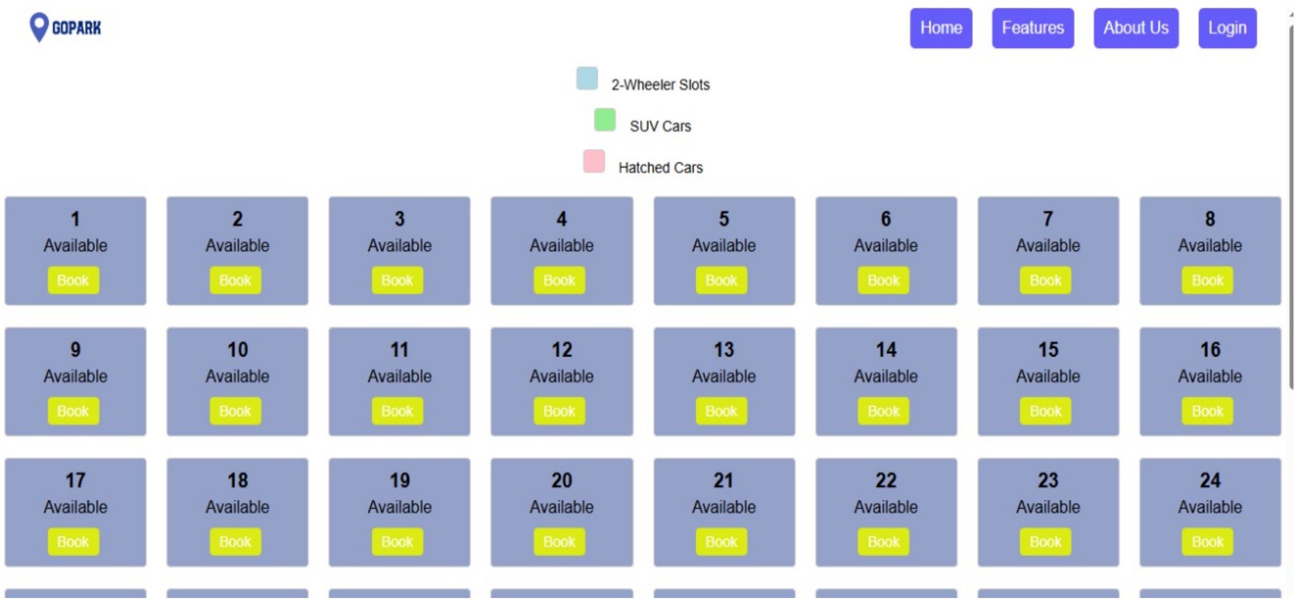
Admin Dashboard

Complexes Companies

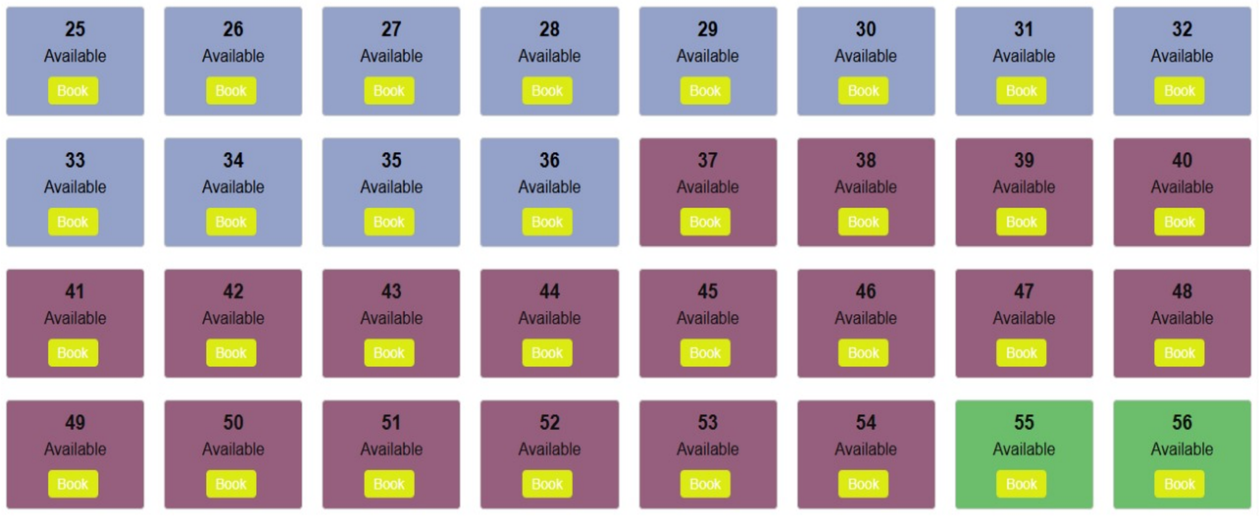
Complexes

Name	Address	Total Parking Levels
abc	abc	2
xyz	xyz	2
abcde	pune	2
abcde	pune	2
abcde	pune	2
abcde	pune	2
abcde	pune	2
abcde	pune	2
abcde	pune	2

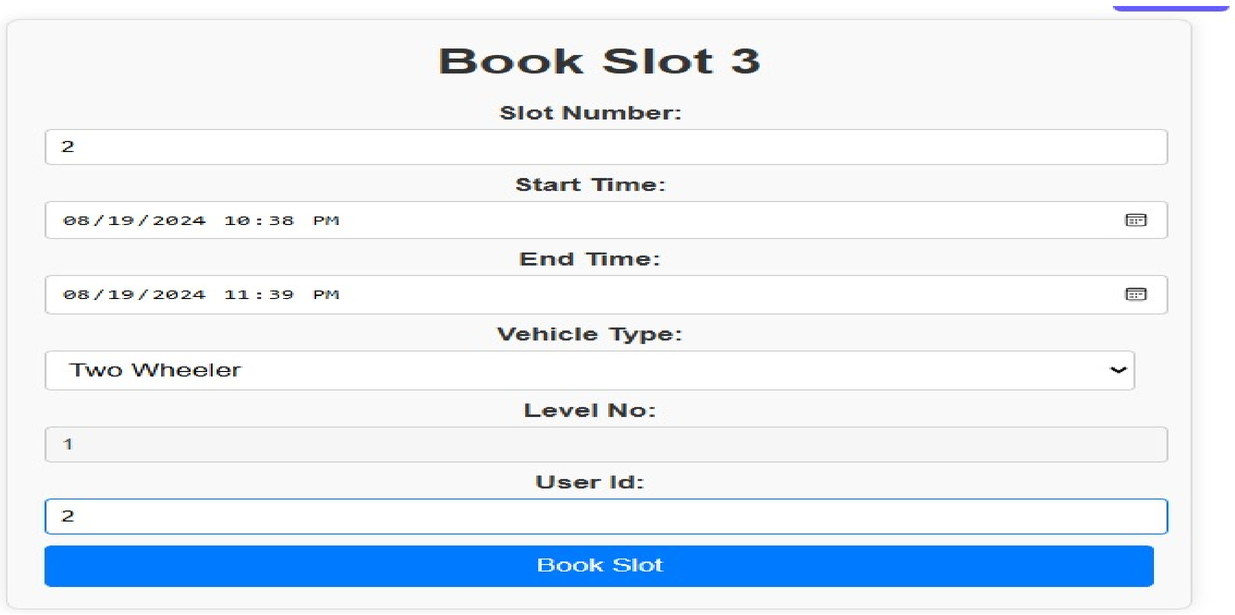
➢ User Booking Grid



1 Available Book	2 Available Book	3 Available Book	4 Available Book	5 Available Book	6 Available Book	7 Available Book	8 Available Book
9 Available Book	10 Available Book	11 Available Book	12 Available Book	13 Available Book	14 Available Book	15 Available Book	16 Available Book
17 Available Book	18 Available Book	19 Available Book	20 Available Book	21 Available Book	22 Available Book	23 Available Book	24 Available Book



➤ **Booking Form**



Book Slot 3

Slot Number: 2

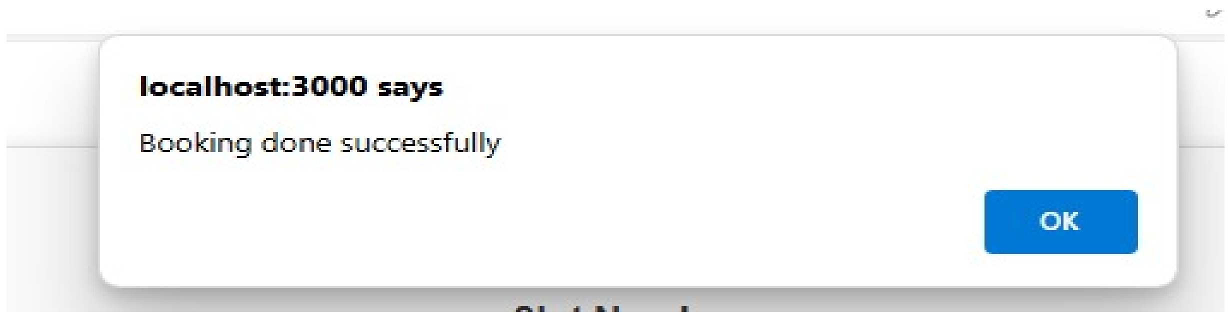
Start Time: 08/19/2024 10:38 PM

End Time: 08/19/2024 11:39 PM

Vehicle Type: Two Wheeler

Level No: 1

User Id: 2



localhost:3000 says

Booking done successfully

8. References

- **React : https://react.dev/learn**
- **Javdocs : https://docs.oracle.com/javase/8/docs/api/**
- **MySql : https://dev.mysql.com/doc/**
- **SpringBoot : https://docs.spring.io/spring-boot/index.html**
- **Spring Boot REST API : https://spring.io/guides/gs/rest-service**