Project Synopsis

On

FIRST PARK THEN PAY

Submitted by

Udhayarajan J

MCA I (Semester: I)

Guided by
Nidhi Damle

Assistant Professor, MCA Department

P.E. Society's
Modern Institute of Business Studies (Autonomous),
Nigdi, Pune - 411044

2025 - 2026

- **Student Name: Udhayarajan J**

- **Title Of the Project: First Park Then Pay**

- **Name of the Company/Client: Academic Project**

- **Abstract:**

The project **“First Park Then Pay”** is a smart parking management system designed to simplify and digitalize the vehicle parking process. It enables users to locate available parking spaces, reserve them, and make payments seamlessly after parking, rather than before. The system aims to minimize congestion, optimize parking space utilization, and improve user convenience. It integrates **Spring Boot RESTful APIs**, **MySQL database**, and **Keycloak** for secure user authentication and authorization. The entire system is containerized using **Docker** for deployment flexibility and scalability.

- **Introduction and Objectives of the Project:**

The **First Park Then Pay** application addresses the common challenges faced in urban parking management by providing a digital solution that automates space tracking, booking, and payment.

Objectives:

1. To provide real-time parking availability information to users.
2. To ensure secure and role-based access using **Keycloak authentication**.
3. To allow users to make payments after parking, ensuring transparency.
4. To maintain a centralized database of parking slots, users, and transactions.
5. To containerize the application using **Docker** for easy deployment and maintenance.

6. To develop a scalable **REST API** backend using **Spring Boot** that can integrate with web and mobile clients.

❖ **Project Category: Web Application**

❖ **Tools / Platform, Hardware and Software Requirement specifications:**

✓ **Operating System Platform**:
  - Ubuntu 22.04 LTS (Linux Environment)

✓ **Hardware Requirements:**
  - **Processor**: Intel Core i3 or equivalent
  - **Hard Disk**: Minimum 50 GB
  - **RAM**: 2 GB or above

✓ **Software Requirements:**
  - **Front-End (4GL Programming Language):** HTML5, CSS3, JavaScript (or Angular if applicable)
  - **Middle Layer**: Java 21 with Spring Boot Framework
  - **Back End**: MySQL 8.0
  - **Other Tool**:
    - **Keycloak** – for identity and access management
    - **Docker** – for containerization and deployment
    - **Postman** – for API testing
    - **Git & GitHub** – for version control
    - **Gradle** – for project build and dependency management

❖ Scope of the Project

The **“First Park Then Pay”** project aims to provide a complete digital solution for managing parking spaces efficiently in urban and semi-urban areas. The system covers the entire parking workflow — from locating available spaces to managing payments after the parking session — ensuring transparency and user convenience.

The project is designed as a **web-based application** with a modular and scalable architecture, allowing future integration with mobile apps or IoT-based sensors for real-time parking detection.

In Scope:

1. **User Management:** Secure registration, login, and authentication using **Keycloak**.
2. **Parking Slot Management:** Real-time tracking of available, booked, and occupied slots.
3. **Payment Management:** Post-parking payment feature with transaction history.
4. **Admin Dashboard:** Role-based access for administrators to manage users, parking slots, and transactions.
5. **RESTful APIs:** Backend APIs built using **Spring Boot** to handle all core operations.
6. **Database Management:** Storage and retrieval of data using **MySQL**.
7. **Deployment:** Containerized setup using **Docker** for consistent development and production environments.

Out of Scope (Future Enhancements):

1. Integration with **IoT sensors** for automated slot detection.
2. Support for **mobile applications** (Android/iOS).
3. Integration with **UPI or payment gateways** for live online transactions.
4. **Smart parking analytics** and predictive suggestions using AI/ML.

Overall, the system is developed to be **secure, maintainable, and scalable**, providing a strong foundation for future expansion into a complete smart parking ecosystem.

❖ Proposed system

The **proposed system**, *First Park Then Pay*, is a modern, automated parking management solution that addresses the inefficiencies of traditional parking systems. Unlike manual methods or prepaid parking solutions, this system allows users to **park first and pay later**, ensuring transparency, convenience, and efficient resource utilization.

The system is designed as a **web-based application** powered by **Java 21** and **Spring Boot RESTful APIs**, providing secure communication between users and administrators. User authentication and role management are handled using **Keycloak**, ensuring a robust and centralized identity management mechanism.

The backend uses **MySQL** for data storage, maintaining details of users, vehicles, parking slots, and transactions. To ensure easy deployment and scalability, the entire application is containerized using **Docker**, enabling consistent performance across different environments.

Key Features of the Proposed System:

1. **User Authentication and Authorization:**
    - Secure login and role-based access using **Keycloak**.
2. **Parking Slot Management:**
    - Real-time tracking of available and occupied parking slots.
3. **Smart Payment System:**
    - Users are billed after their parking duration, enhancing fairness and flexibility.
4. **Admin Dashboard:**
    - Allows administrators to manage users, slots, and payments efficiently.
5. **REST API Integration:**
    - Backend services developed using **Spring Boot** can be easily integrated with web or mobile clients.
6. **Containerized Deployment:**

- Application and database run as **Docker containers** for consistent and portable environments.

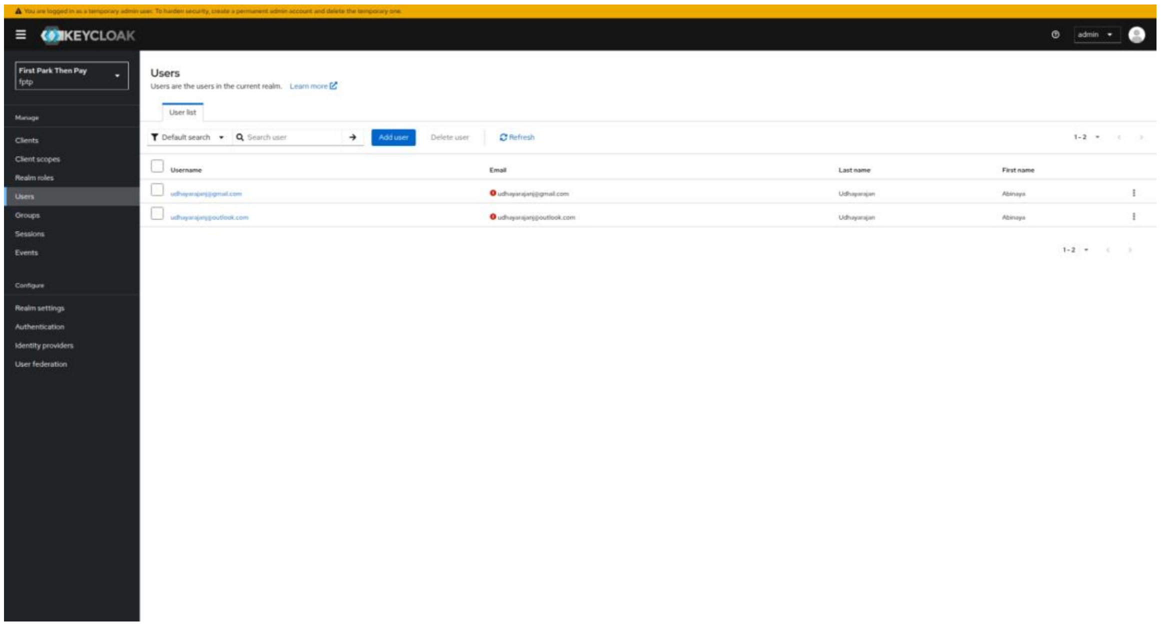
Advantages of the Proposed System:

- Eliminates manual errors and delays in slot allocation.
- Ensures security through token-based authentication and role management.
- Reduces congestion by providing real-time slot availability.
- Improves user satisfaction with a post-parking payment model.
- Scalable and adaptable for future IoT or mobile integrations.

❖ **Module Specification:** [No. of Modules & their Specification]

The **“First Park Then Pay”** application is organized into **four main modules**, each focusing on a specific functionality within the system. It follows a **SaaS (Software as a Service)** model where each customer (parking provider) can create an account and manage their own parking setup, charges, and billing through a secure web application.

1. User Management Module

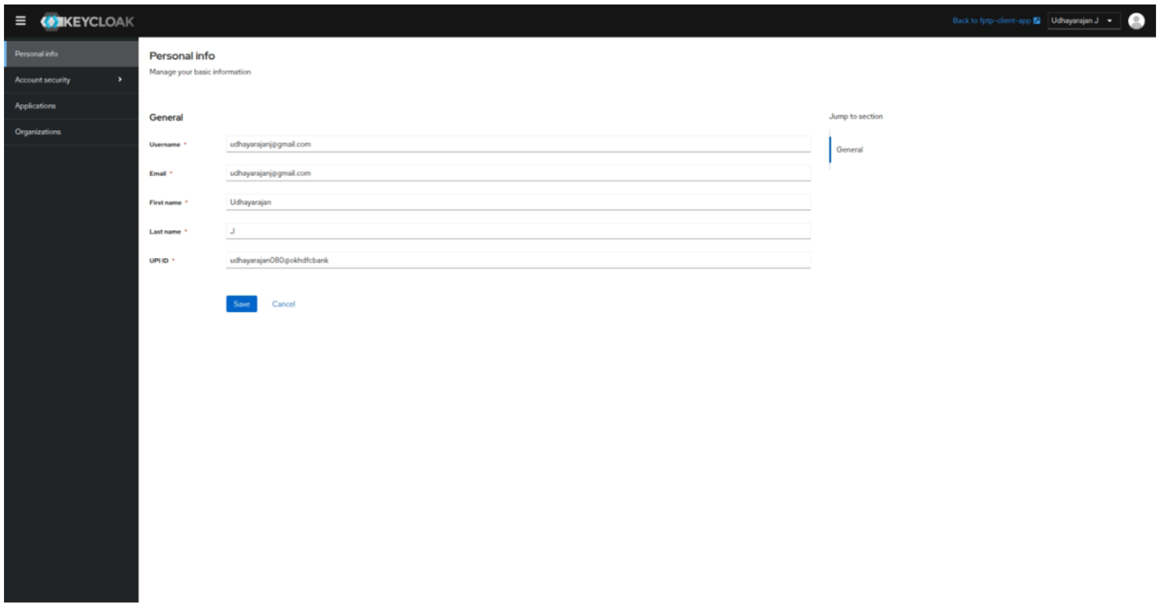


You are logged in as a temporary admin user. To harden security, create a permanent admin account and delete the temporary one.

Users

Users are the users in the current realm. Learn more

Username	Email	Last name	First name
udhayarajanj@gmail.com	udhayarajanj@gmail.com	Udhayarajan	Abinaya
udhayarajanj@outlook.com	udhayarajanj@outlook.com	Udhayarajan	Abinaya

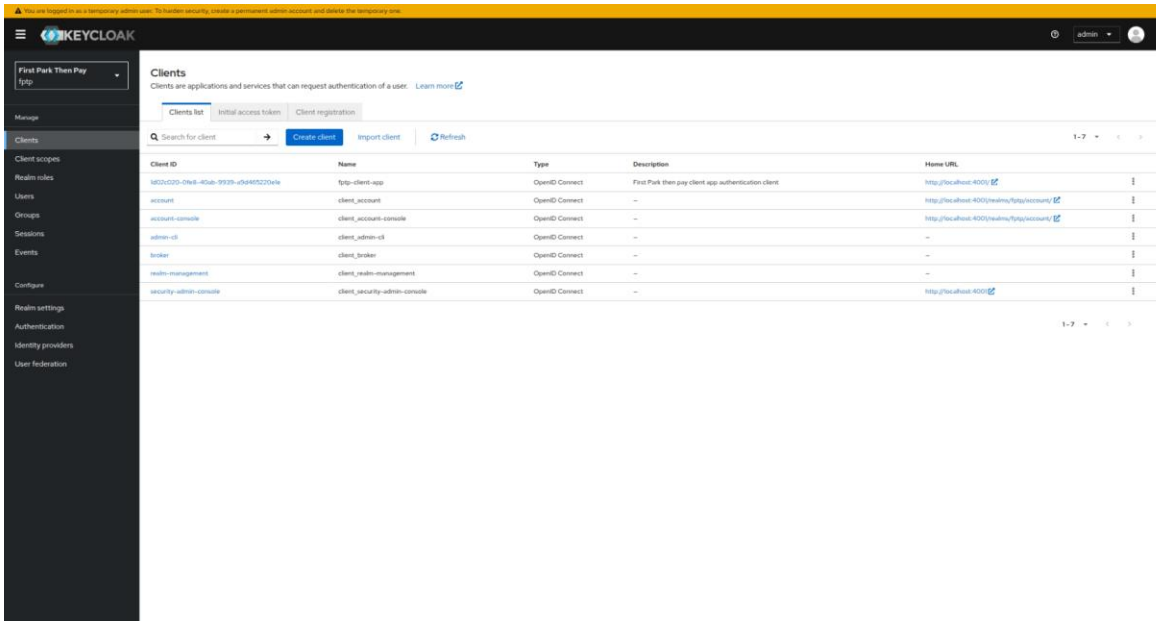


Personal info

Manage your basic information

General

Username *	udhayarajanj@gmail.com
Email *	udhayarajanj@gmail.com
First name *	Udhayarajan
Last name *	J
UPI ID *	udhayarajan080@okhdfcbank



Clients

Clients are applications and services that can request authentication of a user. Learn more

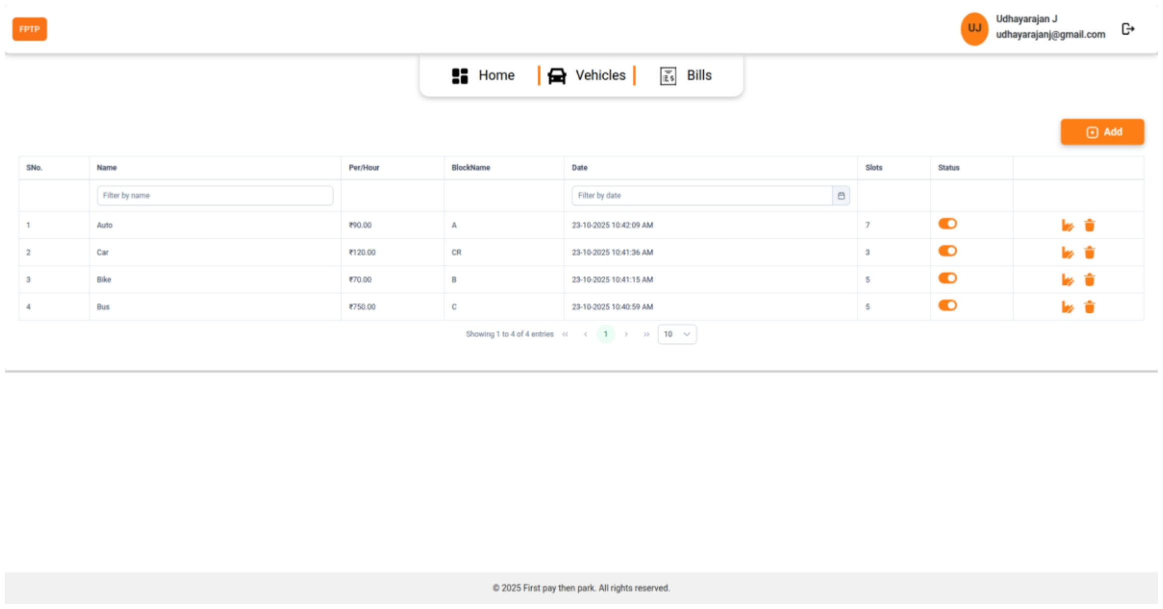
Client ID	Name	Type	Description	Home URL
1d02c020-0fe8-40ab-9939-a9d465220e1e	fptp-client-app	OpenID Connect	First Park then pay client app authentication client	http://localhost:4001/
account	client_account	OpenID Connect	–	http://localhost:4001/realms/fptp/account/
account-console	client_account-console	OpenID Connect	–	http://localhost:4001/realms/fptp/account/
admin-cli	client_admin-cli	OpenID Connect	–	–
broker	client_broker	OpenID Connect	–	–
realm-management	client_realm-management	OpenID Connect	–	–
security-admin-console	client_security-admin-console	OpenID Connect	–	http://localhost:4001

Purpose:
Manages all user accounts, authentication, and authorization through **Keycloak**.

Specifications:

- User authentication and authorization handled by **Keycloak**.
- Role-based access for **Customer (Parking Owner)** and **Admin**.
- Each customer can register and maintain their own parking business account.
- Secure login, token-based access, and session handling.
- Profile details and credentials synced from Keycloak.

2. Vehicle Management Module



SNo.	Name	Per/Hour	BlockName	Date	Slots	Status	
	Filter by name			Filter by date			
1	Auto	₹90.00	A	23-10-2025 10:42:09 AM	7		
2	Car	₹120.00	CR	23-10-2025 10:41:36 AM	3		
3	Bike	₹70.00	B	23-10-2025 10:41:15 AM	5		
4	Bus	₹750.00	C	23-10-2025 10:40:59 AM	5		

Showing 1 to 4 of 4 entries 1 10

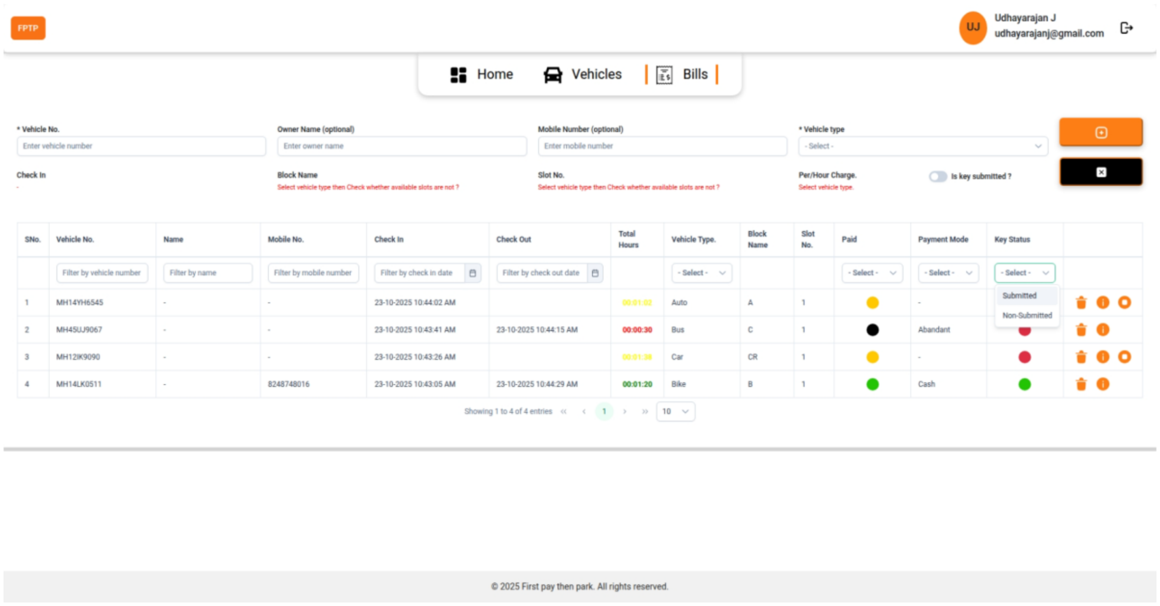
© 2025 First pay then park. All rights reserved.

Purpose:
Maintains vehicle type configurations and associates parking slots with those types.

Specifications:

- Customers can create and manage vehicle types (e.g., Car, Bike, Van, EV, etc.).
- Each vehicle type includes configuration details such as:
    - **Block Name**
    - **Slot Number**
    - **Per-Hour Parking Charges**
- Vehicle type data is used for billing and parking allocation.
- Dropdown-based vehicle type selection ensures consistency.
- Allows expansion for new vehicle types as needed.

3. Billing and Payment Module


FPTP

Udhayarajan J
udhayarajanj@gmail.com

* Vehicle No.	Owner Name (optional)	Mobile Number (optional)	* Vehicle type
Enter vehicle number	Enter owner name	Enter mobile number	- Select -
Check In	Block Name	Slot No.	Per/Hour Charge.
-	Select vehicle type then Check whether available slots are not ?	Select vehicle type then Check whether available slots are not ?	Select vehicle type.

Is key submitted ?

SNo.	Vehicle No.	Name	Mobile No.	Check In	Check Out	Total Hours	Vehicle Type.	Block Name	Slot No.	Paid	Payment Mode	Key Status	
	Filter by vehicle number	Filter by name	Filter by mobile number	Filter by check in date	Filter by check out date		- Select -			- Select -	- Select -	- Select -	
1	MH14YH6545	-	-	23-10-2025 10:44:02 AM		00:01:02	Auto	A	1	🟡	-	Submitted	
2	MH45UJ9067	-	-	23-10-2025 10:43:41 AM	23-10-2025 10:44:15 AM	00:00:30	Bus	C	1	⚫	Abandant	Non-Submitted 🔴	
3	MH12IK9090	-	-	23-10-2025 10:43:26 AM		00:01:38	Car	CR	1	🟡	-	🔴	
4	MH14LK0511	-	8248748016	23-10-2025 10:43:05 AM	23-10-2025 10:44:29 AM	00:01:20	Bike	B	1	🟢	Cash	🟢	

Showing 1 to 4 of 4 entries « ‹ 1 › » 10

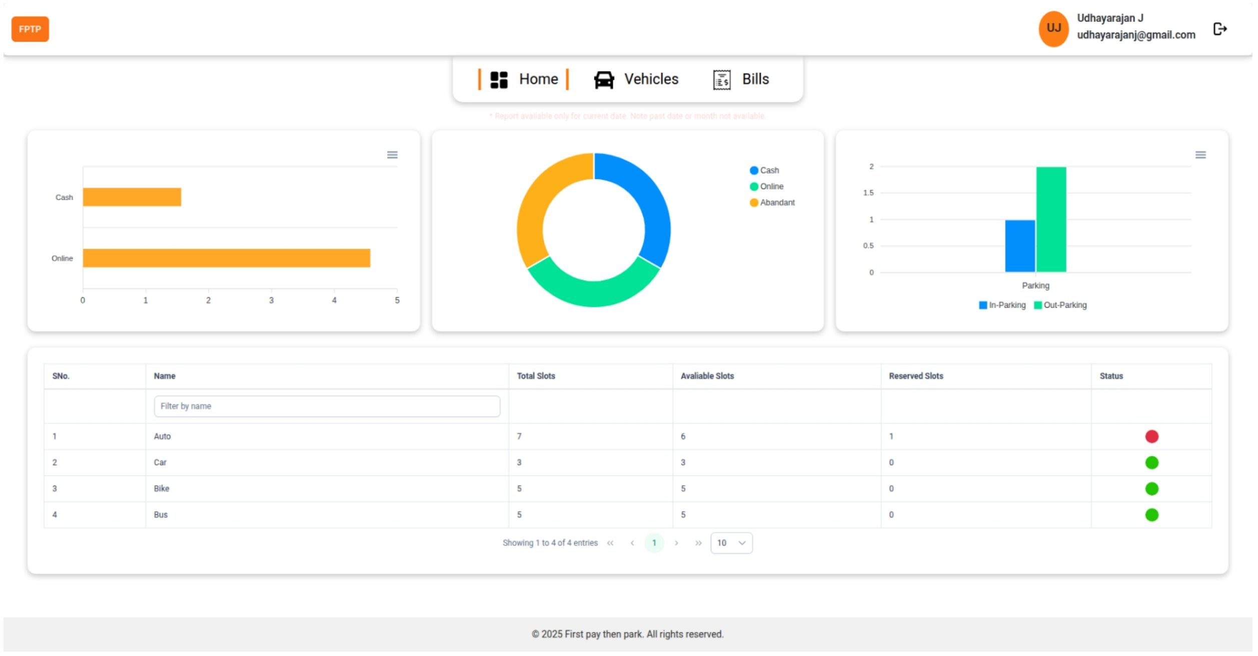
© 2025 First pay then park. All rights reserved.

Purpose:
Handles billing, payment tracking, and transaction verification for parked vehicles.

Specifications:

- Automatically calculates charges based on **parking duration** and **vehicle type**.
- Payment mode options: **Cash** or **Online**.
- Validates whether the bill is paid or pending before releasing the slot.
- Maintains detailed payment history for both admin and users.
- Integrated with vehicle data for accurate charge mapping.

4. Dashboard Module



* Report available only for current date. Note past date or month not available.

SNo.	Name	Total Slots	Avaliable Slots	Reserved Slots	Status
	Filter by name				
1	Auto	7	6	1	🔴
2	Car	3	3	0	🟢
3	Bike	5	5	0	🟢
4	Bus	5	5	0	🟢

Showing 1 to 4 of 4 entries « ‹ 1 › » 10

© 2025 First pay then park. All rights reserved.

Purpose:
Acts as the central interface for both users and admins to monitor system activities and performance.

Specifications:

- **User Dashboard:** Displays registered vehicle types, created blocks/slots, and transaction history.
- **Admin Dashboard:** Summarizes customer accounts, total payments, and parking activities.
- Real-time statistics such as total vehicles parked, earnings, and slot usage.
- Data visualizations through charts or tables for better analysis.

❖ Bibliography:

- **Angular Framework Documentation**
  https://angular.io/docs
- **Node.js Official Documentation**
  https://nodejs.org/en/docs
- **Java 21 Documentation (Oracle)**
  https://docs.oracle.com/en/java/
- **Spring Boot Framework Documentation**
  https://docs.spring.io/spring-boot/docs/current/reference/html/
- **Keycloak Documentation**
  https://www.keycloak.org/documentation
- **MySQL Database Documentation**
  https://dev.mysql.com/doc/
- **Docker Documentation**
  https://docs.docker.com/
- **Visual Studio Code Documentation**
  https://code.visualstudio.com/docs
- **IntelliJ IDEA Community Edition Documentation**
  https://www.jetbrains.com/idea/documentation/
- **Project Source Code (GitHub Repository)**
  https://github.com/UdhayarajanJ/2202561-MCA-2025-2027.git

Student Email: udhayarajanj@gmail.com

Student Contact No: 8248748016

Date: _____________________

Place: Pune **Signature of Student**