



**PES University, Bengaluru**

(Established under Karnataka Act 16 of 2013)

**Department of Computer Science & Engineering**  
**Session: Jan - May 2022**

**Object Oriented Analysis and Design with Java - Laboratory**  
**UE19CS353**

**Mini Project**

Report on

**Automated Parking System**

**By:**

**Abhishek Aditya BS – PES1UG19CS019**

**A Sai Chaithanya – PES1UG19CS002**

**Adithya MS – PES1UG19CS027**

**6<sup>th</sup> Semester 'A'**

## 1. Project Description

Automated Parking System allows users to park at the nearest parking location without human intervention. The application provides a default interface where a user can select if he is an admin of a parking lot or a user. If the user selects the option Parking lot admin then he is directed to a separate interface where they can add their parking locations by specifying the parking lot name, address, latitude, longitude, total capacity and currently filled slots. If the option selected is User, then he is directed to a user registration form if he is using the application for the first time. The registration form asks users to enter their details like name, userID, password, phone number, Driver's Licence number, Car Model, colour, plate number.

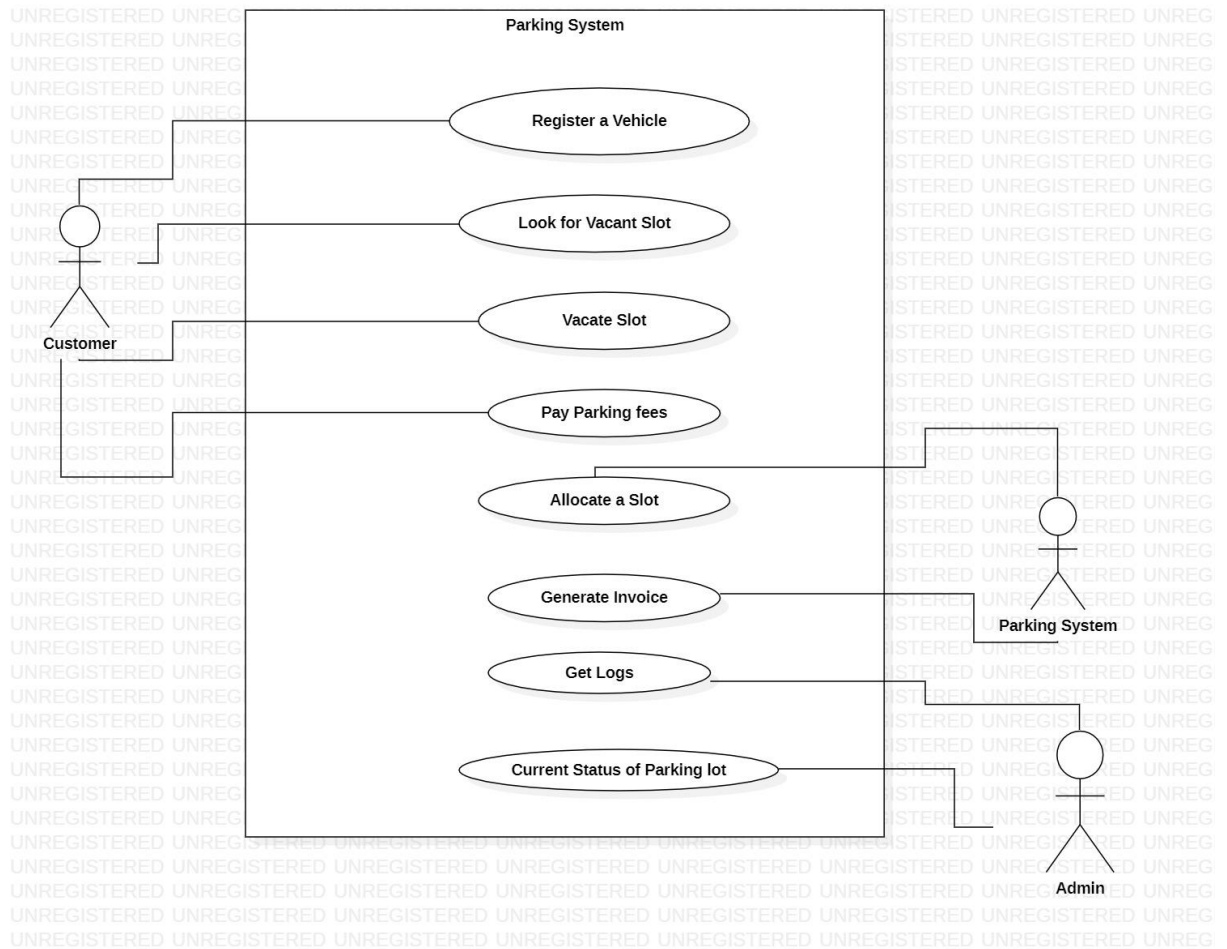
Once the user finishes the registration, he is asked to login to the system by entering the registered userID and password, and the system asks for his permission to fetch their geolocation. The geolocation is fetched by using a python API, which uses their IP address to determine the user's latitude, longitude and its corresponding address. Once the user's location is fetched, he is shown the top five nearest parking locations with total capacity, address and number of currently filled slots. He can choose any parking lot near him and he is shown the pricing for parking for different durations. After the user is shown the pricing, the system stores the user details with the slot number and the parking lot he has parked in, and logs out the user.

When the user comes back to the application to unpark his car, after he logs in, he is shown the details about his car and his parking slot and location. He is shown the bill for the amount of duration parked, and prompts him to choose the payment method and asks for his feedback on the user experience. After the user completes all the actions, his entry from the loggedInUsers is removed and the parking slot is freed and all other details are updated accordingly.

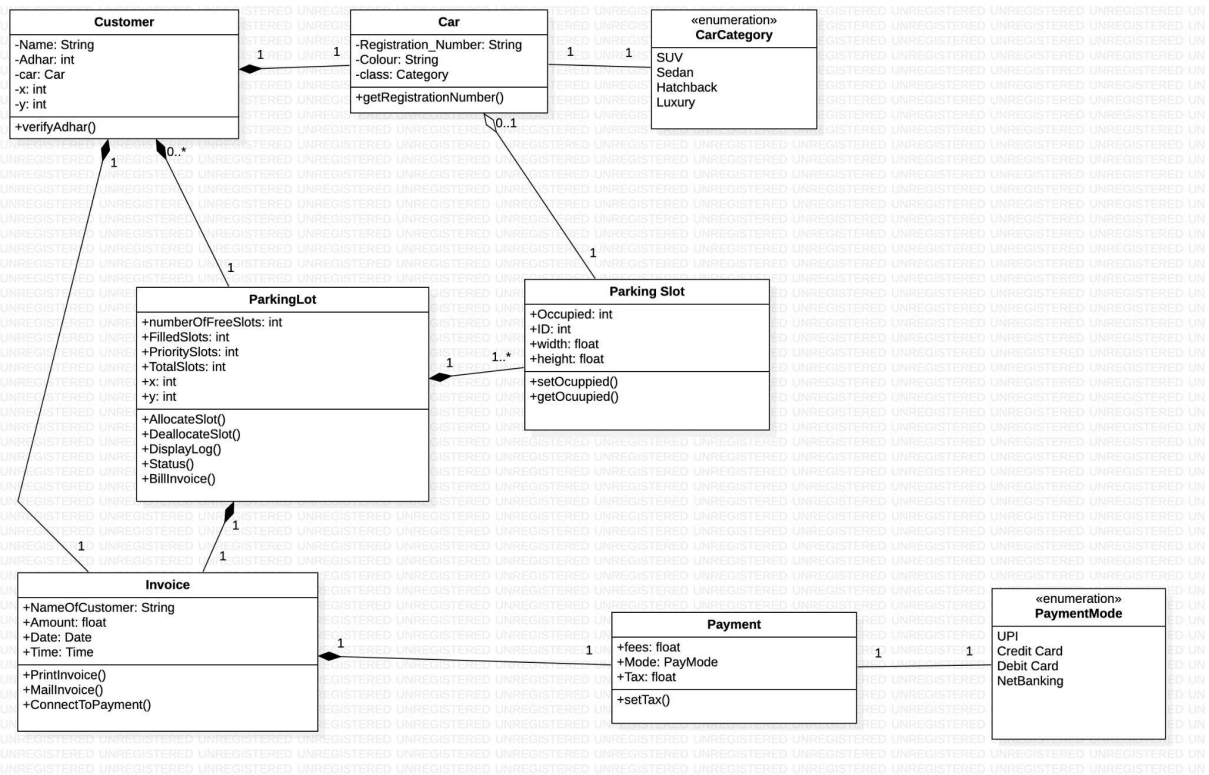
Github link to our project: [Automated Parking System](#)

## 2. Analysis and Design Models

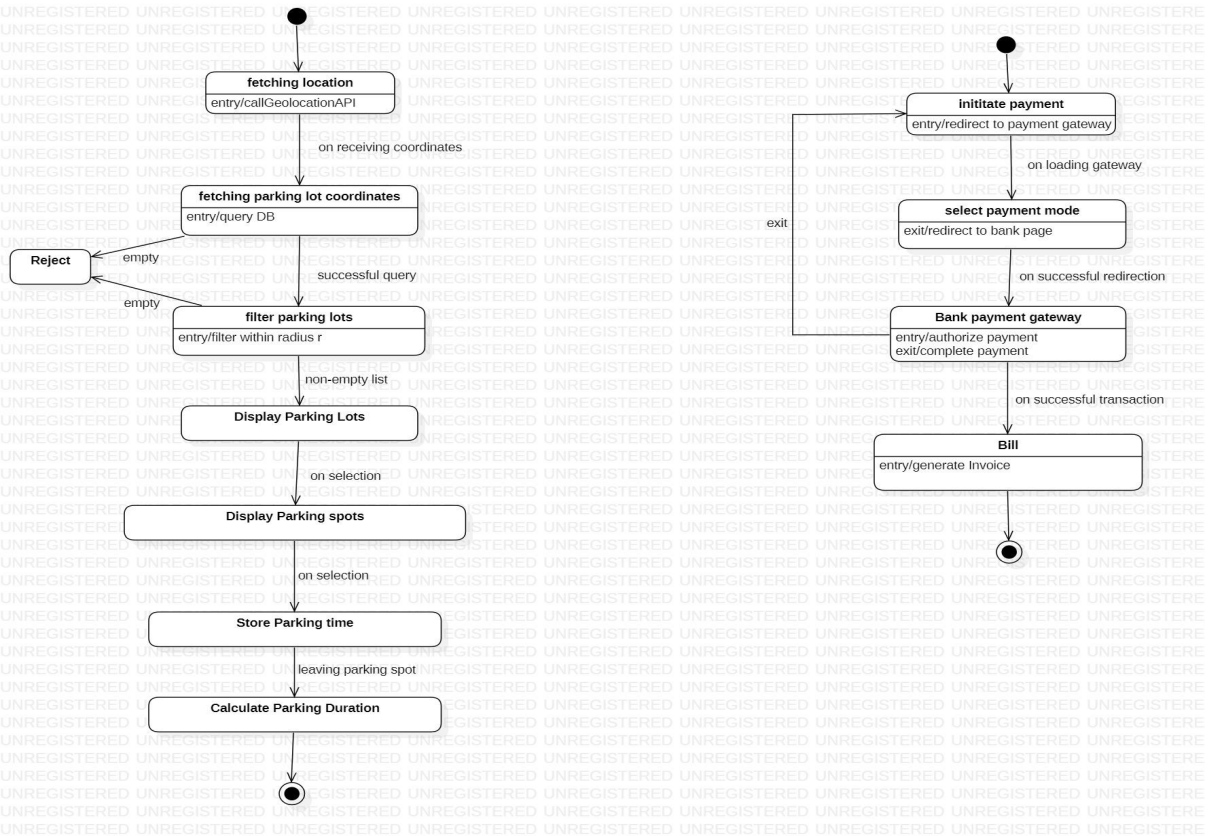
### - Use Case Diagram



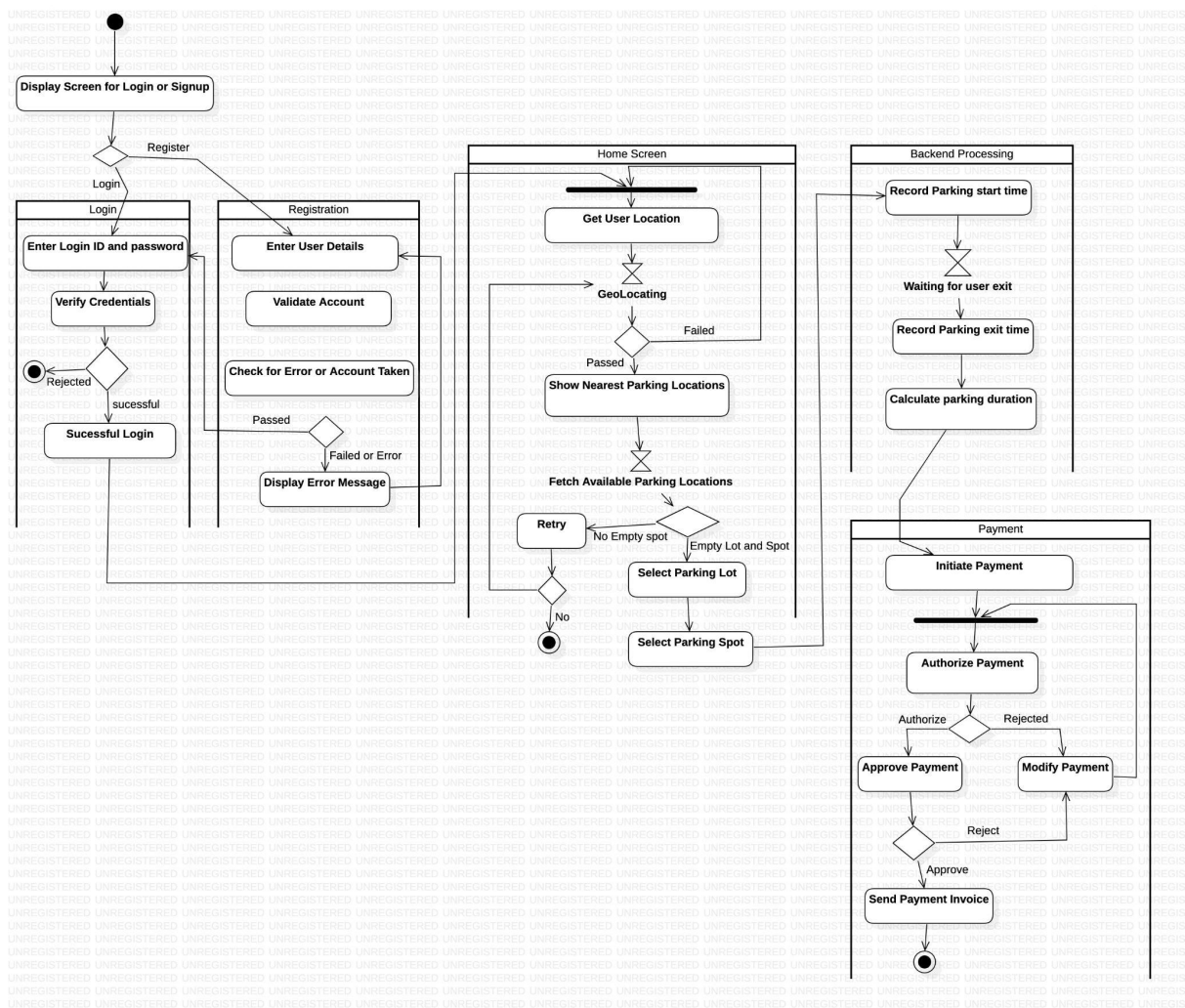
## - Class Diagram



## - State Diagram



## - Activity Diagram



## 3. Tools and Frameworks Used:

- **Maven:** Maven is a build automation tool used primarily for Java projects. Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information.
- **Python Geolocator:** It is possible using geopy to extract the coordinates meaning its latitude and longitude. Therefore, it can be used to express the location in terms of coordinates.

- **Java Swing:** Java Swing tutorial is a part of Java Foundation Classes (JFC) that is used to create window-based applications. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java. Unlike AWT, Java Swing provides platform-independent and lightweight components. The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.
- **MongoDB:** MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server Side Public Licence.

#### 4. Design Principles and Design Patterns Applied

- **MVC** is an architectural pattern which means it rules the whole architecture of the applications. Even though often it is known as a design pattern, we may be wrong if we refer to it only as a design pattern because design patterns are used to solve a specific technical problem, whereas architecture pattern is used for solving architectural problems, so it affects the entire architecture of our application. It has three main components Model, View, Controller and each of them has specific responsibilities. Modules where MVC pattern was used in the project:
  1. User Registration
  2. User Login
  3. Admin
  4. Dashboard
  5. Billing
  6. WelcomeBack
  7. GeoLocation

- **Keep It Simple and Stupid (KISS) Principle**

The Keep it Simple and Stupid (KISS) principle is a reminder to keep your code simple and readable for humans. If your method handles multiple use-cases, split them into smaller functions. Unreadable and long methods are very hard to maintain for human programmers, bugs are harder to find if it performs multiple functionalities, programmers need to make multiple methods instead. Since the whole project is completely modular with different functionalities organised into different modules, the system has high cohesion and low coupling, therefore simple to read and maintain.

- **Open Closed Design Principle**

Classes, methods or functions should be Open for extension (new functionality) and Closed for modification. All the classes in the project can be extended by inheriting and overriding the existing functions, but trying to modify the classes and its methods will break the system and cause other unforeseen issues.

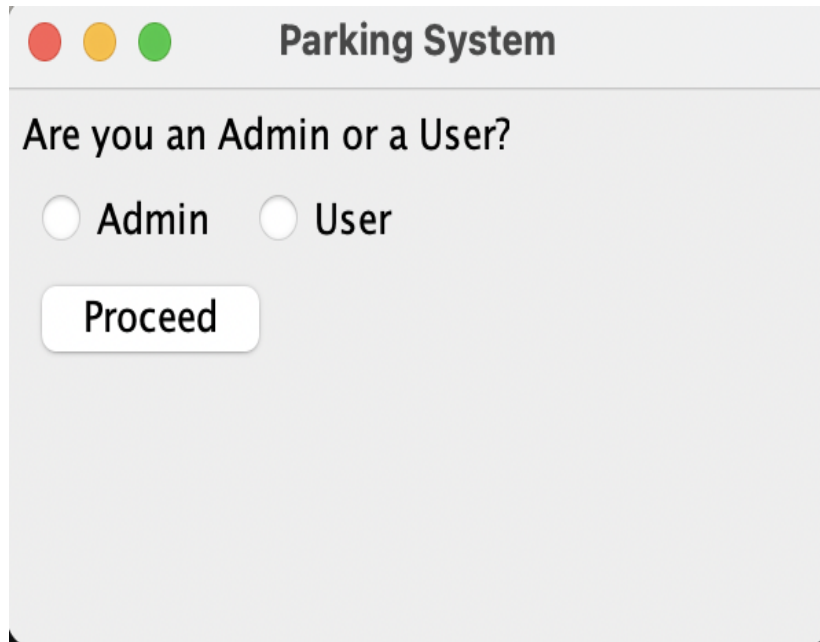
- **Single Responsibility Principle (SRP)**

As per SRP, there should not be more than one reason for a class to change, or a class should always handle single functionality. An example where the above principle was used in the database API's where each API performs only a single user functionality like fetching userID, inserting parkingLotSelected, etc.

- **The Composition Over Inheritance Principle**

One should often prefer composition over inheritance when designing their systems. In Java, this means that we should more often define interfaces and implement them, rather than defining classes and extending them. This principle was used throughout the project where objects of the database were created inside the classes using it, instead of inheriting the functionalities of the database class. Another example where this principle was used was using the objects of the view and controller of various modules during the current frame dispose function and calling the next frame by using the view and controller of the required module.

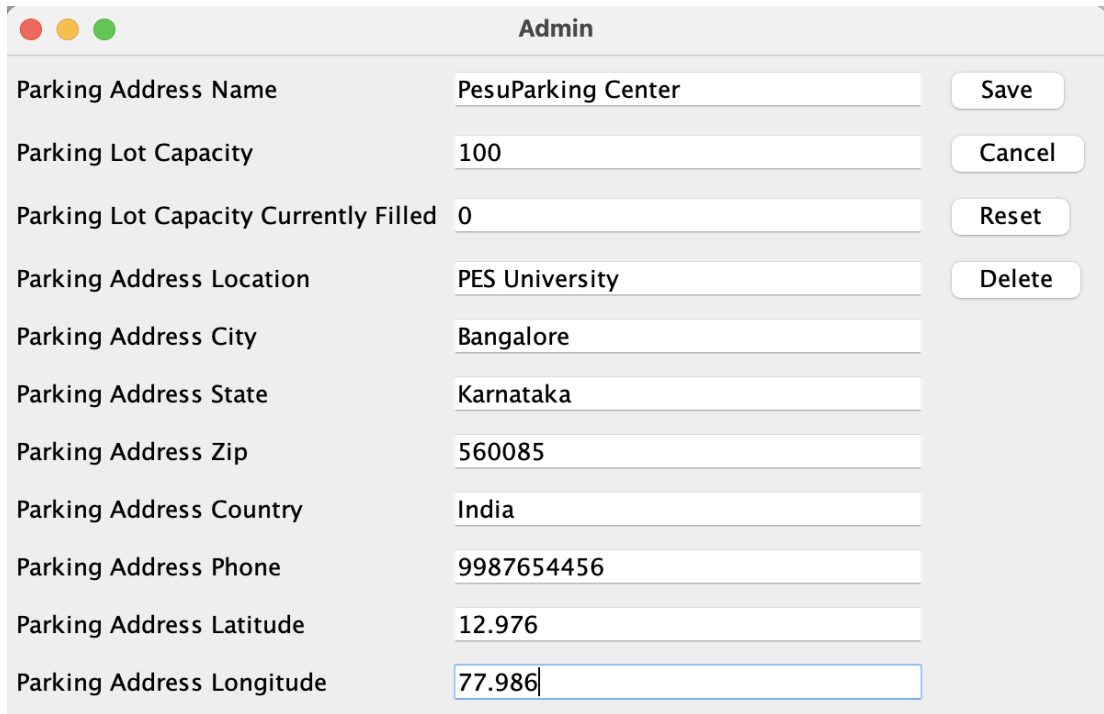
## 5. Application Screenshots



A screenshot of a macOS-style window titled "Parking System". The window has a light gray background and rounded corners. At the top, there are three colored window control buttons (red, yellow, green). Below the title bar, the text "Are you an Admin or a User?" is displayed. Underneath, there are two radio buttons: "Admin" and "User". Below the radio buttons is a white button with the text "Proceed".

**Figure 1: Default Display shown to the users**

A very first onboarding page which asks if the user is a admin of parking lot or a user who wants his/her car to be parked

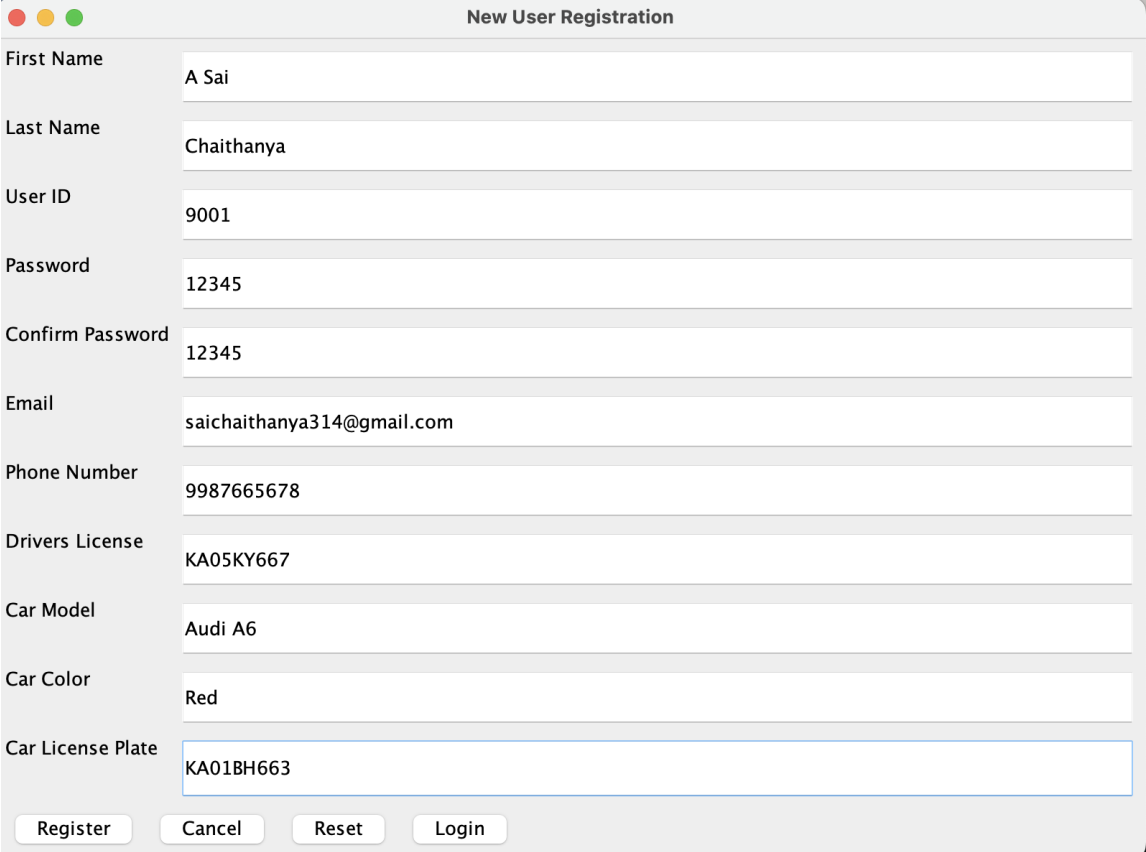


A screenshot of a macOS-style window titled "Admin". The window has a light gray background and rounded corners. At the top, there are three colored window control buttons (red, yellow, green). Below the title bar, there is a form for "Parking Lot Registration". The form consists of several text input fields, each with a label to its left. The labels are: "Parking Address Name", "Parking Lot Capacity", "Parking Lot Capacity Currently Filled", "Parking Address Location", "Parking Address City", "Parking Address State", "Parking Address Zip", "Parking Address Country", "Parking Address Phone", "Parking Address Latitude", and "Parking Address Longitude". The input fields contain the following values: "PesuParking Center", "100", "0", "PES University", "Bangalore", "Karnataka", "560085", "India", "9987654456", "12.976", and "77.986". To the right of the input fields, there are four buttons: "Save", "Cancel", "Reset", and "Delete".

**Figure 2: Admin Interface**

Parking Lot Registration form for the admin of the parking lot





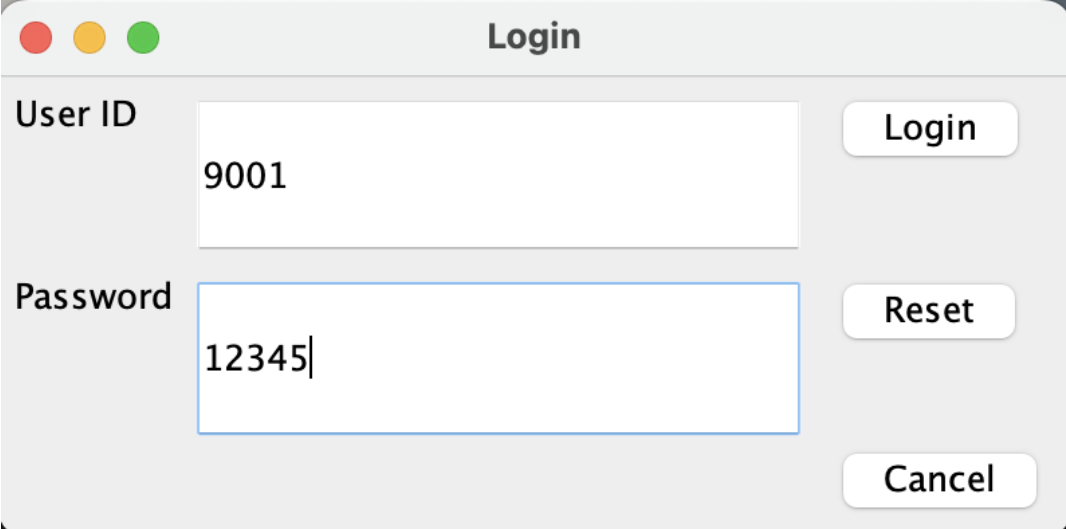
A screenshot of a 'New User Registration' window. It features a series of input fields for user details: First Name (A Sai), Last Name (Chaithanya), User ID (9001), Password (12345), Confirm Password (12345), Email (saichaithanya314@gmail.com), Phone Number (9987665678), Drivers License (KA05KY667), Car Model (Audi A6), Car Color (Red), and Car License Plate (KA01BH663). At the bottom, there are four buttons: Register, Cancel, Reset, and Login.

First Name	A Sai
Last Name	Chaithanya
User ID	9001
Password	12345
Confirm Password	12345
Email	saichaithanya314@gmail.com
Phone Number	9987665678
Drivers License	KA05KY667
Car Model	Audi A6
Car Color	Red
Car License Plate	KA01BH663

Buttons: Register, Cancel, Reset, Login

**Figure 3: New User Registration**

Registration form for the user which takes in all the information related to user and his/her car



A screenshot of a 'Login' window. It contains two input fields: User ID (9001) and Password (12345). To the right of these fields are three buttons: Login, Reset, and Cancel.

User ID	9001
Password	12345

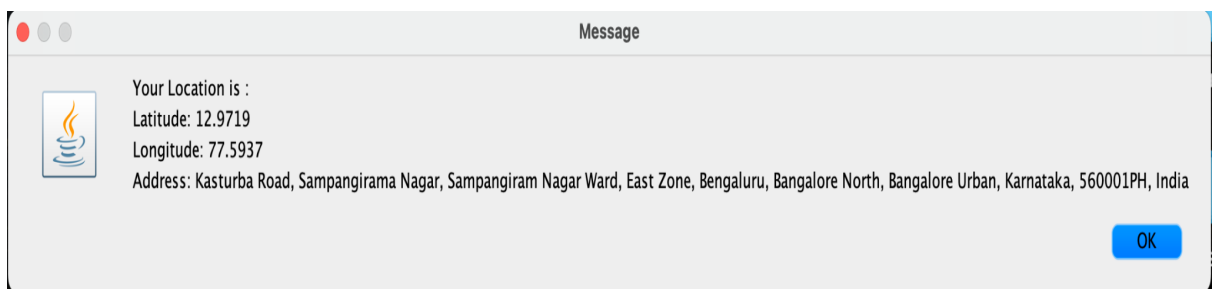
Buttons: Login, Reset, Cancel

**Figure 4: User Login**

Login Page to sign in to the application with correct credentials



**Figure 5: Geolocation Permission**  
Permission asked for the user to fetch his geolocation



**Figure 6: GeoLocation Details**  
Geolocation details of the user fetched by the PythonAPI

Parking System

Showing top 5 nearest parking lots

Select a parking lot to proceed

Parking Lot 1

Jayanagar Parking Center Jayanagar 4th Block, Bangalore 560045, Karnataka, India | Total Capacity :100 | Currently Filled: 2

Parking Lot 2

Malleshwaram Parking Center Mantri Mall, Malleshwaram 7th cross, Bangalore 560085, Karnataka, India | Total Capacity :100 | Currently Filled: 0

Parking Lot 3

Kengeri Parking Center Kengeri Main road, Bangalore 560043, Karnataka, India | Total Capacity :50 | Currently Filled: 2

Parking Lot 4

Rajaji Nagar Parking Center Rajaji Nagar, Orion Mall, Bangalore 560038, Karnataka, India | Total Capacity :465 | Currently Filled: 2

Parking Lot 5

Indiranagar Parking Center Indiranagar 100ft Road, Bangalore 560096, Karnataka, India | Total Capacity :500 | Currently Filled: 1

☒ Parking Lot 1

☐ Parking Lot 2

☐ Parking Lot 3

☐ Parking Lot 4

☐ Parking Lot 5

Proceed

Cancel

**Figure 7: Available Parking lots**  
Shows top 5 nearest parking locations to the user

User Logout

Pricing for Parking

The price for parking at Rajaji Nagar Parking Center is as follows:

First 2 hours: 20\$

First 3 hours: 25\$

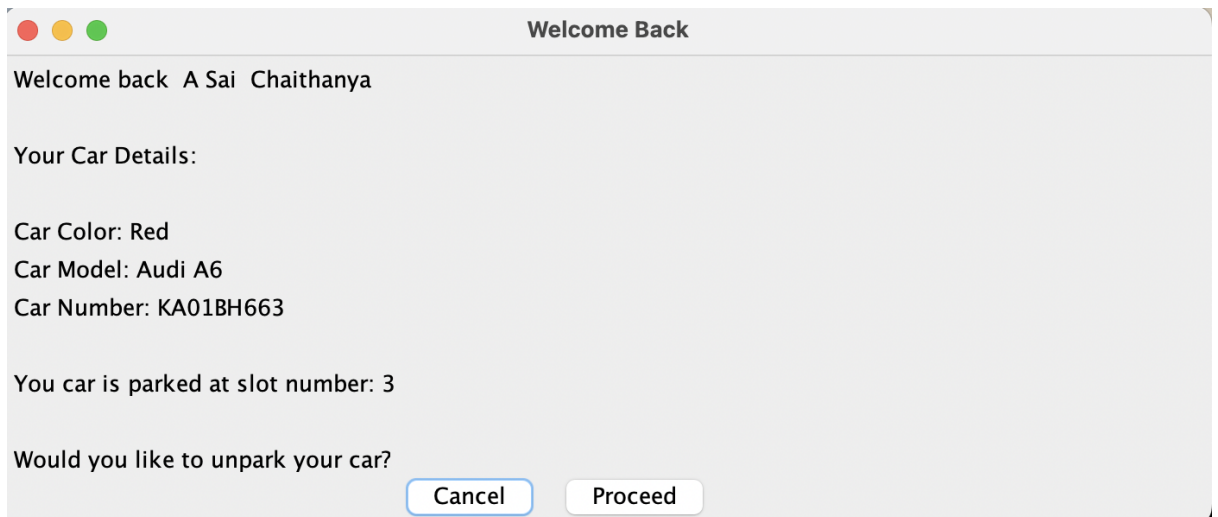
Every Additional hour above 3 hours: 10\$

First 24 hours: 100\$

Every Additional hour above 24 hours: 20\$

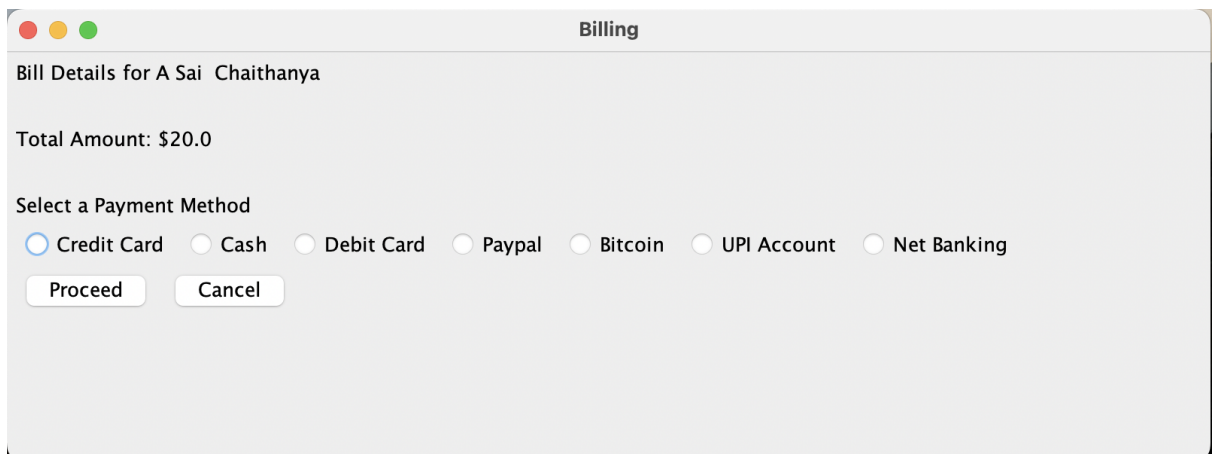
Logout

**Figure 8: Parking Lot Price Details**  
Shows the pricing details of parking lot



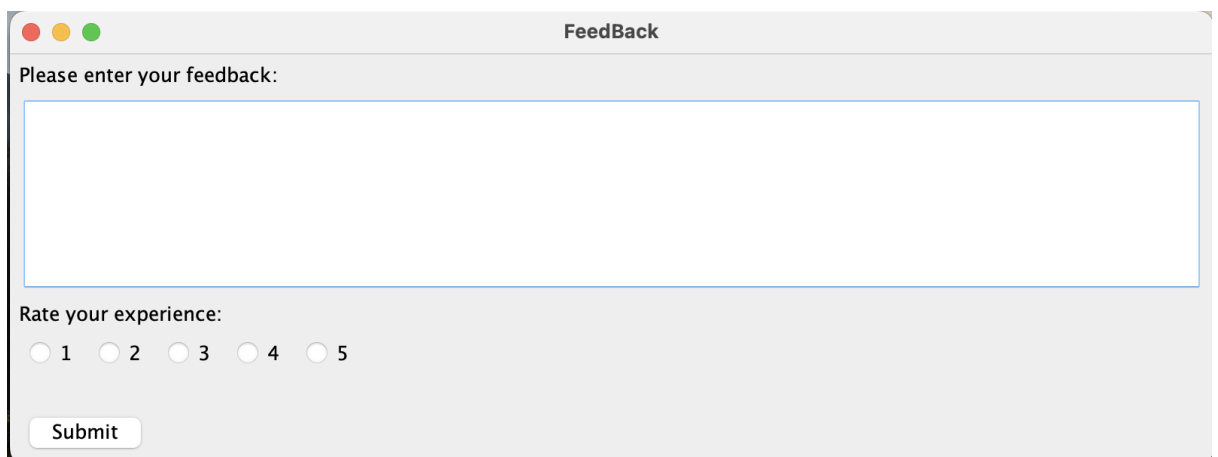
A screenshot of a web application window titled "Welcome Back". The window has a light gray background and a title bar with three colored buttons (red, yellow, green). The content area displays the following text: "Welcome back A Sai Chaithanya", "Your Car Details:", "Car Color: Red", "Car Model: Audi A6", "Car Number: KA01BH663", "You car is parked at slot number: 3", and "Would you like to unpark your car?". At the bottom right, there are two buttons: "Cancel" and "Proceed".

**Figure 9: Welcome back screen**  
Shows user's details when the user logs back in



A screenshot of a web application window titled "Billing". The window has a light gray background and a title bar with three colored buttons (red, yellow, green). The content area displays the following text: "Bill Details for A Sai Chaithanya", "Total Amount: \$20.0", "Select a Payment Method", and a list of payment methods: "Credit Card", "Cash", "Debit Card", "Paypal", "Bitcoin", "UPI Account", and "Net Banking". Each method is preceded by a radio button. At the bottom, there are two buttons: "Proceed" and "Cancel".

**Figure 10: Billing screen**  
Shows the parking charges and asks user for payment method



A screenshot of a web application window titled "FeedBack". The window has a light gray background and a title bar with three colored buttons (red, yellow, green). The content area displays the following text: "Please enter your feedback:", a large text input field, "Rate your experience:", and a list of ratings: "1", "2", "3", "4", and "5". Each rating is preceded by a radio button. At the bottom left, there is a "Submit" button.

**Figure 11: User Feedback screen**  
User can provide feedback and rate the Parking lot parking experience

## 6. Team member contributions

Team Member Name	Contribution
A Sai Chaithanya	<ol style="list-style-type: none"><li>1. High Level System Design and Workflow.</li><li>2. Use Case Diagram.</li><li>3. Parking Lot Feedback Page.</li><li>4. Payment Page.</li><li>5. WelcomeBack page</li></ol>
Abhishek Aditya BS	<ol style="list-style-type: none"><li>1. Activity Diagrams.</li><li>2. Class Diagram.</li><li>3. Parking Lot Registration.</li><li>4. Dashboard</li><li>5. Fetch Top 5 nearest parking lots.</li></ol>
Adithya M S	<ol style="list-style-type: none"><li>1. State Diagrams.</li><li>2. Class Diagram.</li><li>3. User Registration and Login Pages.</li><li>4. Fetch Geolocation</li><li>5. User Logout functionality.</li></ol>