

# SMART PARKING SYSTEM



## A PROJECT REPORT

*Submitted by*

**ARCHANA J (2303811710422009)**

*in partial fulfillment of requirements for the award of the course*

**CGB1201 - JAVA PROGRAMMING**

*In*

**COMPUTER SCIENCE AND ENGINEERING**

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY**

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

**SAMAYAPURAM – 621 112**

**NOVEMBER- 2024**

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY  
(AUTONOMOUS)**

**SAMAYAPURAM – 621 112**

**BONAFIDE CERTIFICATE**

Certified that this project report on “**SMART PARKING SYSTEM**”  
is the bonafide work of **ARCHANA J (2303811710422009)** who carried out the  
project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING  
Dr.A.DELPHIN CAROLINA RANI, M.E., Ph.D.,  
HEAD OF THE DEPARTMENT  
PROFESSOR

**SIGNATURE**

Dr.A.Delphin Carolina Rani, M.E., Ph.D.,

**HEAD OF THE DEPARTMENT**

PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology  
(Autonomous)

Samayapuram-621112.

CGB1201-JAVA PROGRAMMING  
Mr. M. SARAVANAN, M.E.,  
SUPERVISOR  
ASSISTANT PROFESSOR

**SIGNATURE**

Mr. M. Saravanan, M.E.,

**SUPERVISOR**

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology  
(Autonomous)

Samayapuram-621112.

Submitted for the viva-voce examination held on 02.12.2024

CGB1201-JAVA PROGRAMMING  
Mr. MAHARAJAN A, M.E.,  
INTERNAL EXAMINER  
ASSISTANT PROFESSOR

**INTERNAL EXAMINER**

CGB1201-JAVA PROGRAMMING  
Dr. K. SETHAMILSELVI, M.E., Ph.D.,  
EXTERNAL EXAMINER  
PROFESSOR  
8138-SCE, TRICHY.

**EXTERNAL EXAMINER**

## DECLARATION

I declare that the project report on “**SMART PARKING SYSTEM**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201- JAVA PROGRAMMING**.

**Signature**



---

ARCHANA J

Place: Samayapuram

Date: 02.12.2024

## ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr A. DELPHIN CAROLINA RANI, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project guide **Mr. M. SARAVANAN, M.E.**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

## **VISION OF THE INSTITUTION**

To serve the society by offering top-notch technical education on par with global standards

## **MISSION OF THE INSTITUTION**

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

## **VISION OF DEPARTMENT**

To be a center of eminence in creating competent software professionals with research and innovative skills.

## **MISSION OF DEPARTMENT**

**M1: Industry Specific:** To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

**M2: Research:** To prepare students for research-oriented activities.

**M3: Society:** To empower students with the required skills to solve complex technological problems of society.

## **PROGRAM EDUCATIONAL OBJECTIVES**

### **1. PEO1: Domain Knowledge**

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

### **2. PEO2: Employability Skills and Research**

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

### **3. PEO3: Ethics and Values**

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

### **PROGRAM SPECIFIC OUTCOMES (PSOs)**

#### **PSO 1: Domain Knowledge**

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

#### **PSO 2: Quality Software**

To apply software engineering principles and practices for developing quality software for scientific and business applications.

#### **PSO 3: Innovation Ideas**

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

### **PROGRAM OUTCOMES (POs)**

Engineering students will be able to:

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

- 5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
- 6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
- 7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
- 8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

## **ABSTRACT**

Efficient management of parking spaces is essential in modern urban environments to enhance user convenience and optimize utilization. This project introduces a Java application designed to monitor real-time parking space occupancy and dynamically adjust pricing based on demand. Additionally, it features a reservation module that allows users to book parking spaces for specific time slots, ensuring guaranteed availability. Designed to be scalable, the application employs object-oriented design, data structures, and algorithms for robustness and efficiency. By integrating real-time monitoring, dynamic pricing, and reservation management, this Java application offers a comprehensive and innovative solution for modern parking management challenges. Furthermore, the application can be adapted to various parking lot sizes and configurations, making it a versatile tool for different environments. The use of dynamic pricing strategies helps manage peak times effectively, encouraging more balanced usage of parking spaces throughout the day. Through its reservation system, users experience increased convenience, reducing the time spent searching for available parking. This project demonstrates the potential of leveraging advanced programming techniques to solve real-world problems, providing a practical approach to efficient parking management.



## ABSTRACT WITH POs AND PSOs MAPPING

### CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
Efficient management of parking spaces is essential in modern urban environments to enhance user convenience and optimize utilization. This project introduces a Java application designed to monitor real-time parking space occupancy and dynamically adjust pricing based on demand. Additionally, it features a reservation module that allows users to book parking spaces for specific time slots, ensuring guaranteed availability. The application includes three core components: real-time occupancy monitoring, which tracks the occupancy status of each parking space, providing accurate and up-to-date availability information; dynamic pricing adjustment, which automatically modifies prices based on occupancy levels to optimize revenue and utilization; and a reservation module, which handles booking requests, confirms reservations, and updates availability, ensuring a reserved spot for users upon arrival. Designed to be scalable, the application employs object-oriented design, data structures, and algorithms for robustness and efficiency. By integrating real-time monitoring, dynamic pricing, and reservation management, this Java application offers a comprehensive and innovative solution for modern parking management challenges..	<b>PO1 -3</b> <b>PO2 -3</b> <b>PO3 -3</b> <b>PO4 -3</b> <b>PO5 -3</b> <b>PO6 -3</b> <b>PO7 -3</b> <b>PO8 -3</b> <b>PO9 -3</b> <b>PO10 -3</b> <b>PO11-3</b> <b>PO12 -3</b>	<b>PSO1 -3</b> <b>PSO2 -3</b> <b>PSO3 -3</b>

Note: 1- Low, 2-Medium, 3- High

## **TABLE OF CONTENTS**

<b>CHAPTER NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
	<b>ABSTRACT</b>	<b>viii</b>
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming Concepts	2
		<b>3</b>
<b>2</b>	<b>PROJECT METHODOLOGY</b>	
	2.1 Proposed Work	3
	2.2 Block Diagram	3
		<b>4</b>
<b>3</b>	<b>MODULE DESCRIPTION</b>	
	3.1 Calculate Net Salary Module	4
	3.2 Print Salary Report Module	4
	3.3 Add Employee Module	4
	3.4 Display Menu Module	4
		<b>5</b>
<b>4</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	
	4.1 Conclusion	5
	4.2 Future Scope	5
		<b>6</b>
	<b>REFERENCES</b>	
		<b>7</b>
	<b>APPENDIX A(SOURCE CODE)</b>	
		<b>11</b>
	<b>APPENDIX B(SCREENSHOTS)</b>	

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 OBJECTIVE**

The objective of this Java project is to create a comprehensive application for efficient parking space management in urban environments. This application aims to monitor real-time occupancy of parking spaces and dynamically adjust pricing based on demand to optimize usage and revenue. Additionally, it includes a reservation module that allows users to book parking spaces for specific time slots, ensuring availability upon arrival. By implementing features such as real-time monitoring, dynamic pricing, and a user-friendly reservation system, the project seeks to enhance convenience for users and operational efficiency for parking lot managers. The design will be scalable and adaptable to various parking lot sizes and configurations, making it a versatile tool for different environments. Advanced programming techniques, including object-oriented design, data structures, and algorithms, will be utilized to ensure the robustness, efficiency, and reliability of the application. The ultimate goal is to provide a practical, innovative solution to modern parking management challenges, leveraging Java to address real-world problems effectively and improve the overall user experience.

### **1.2 OVERVIEW**

This project involves developing a Java-based application designed to efficiently manage parking spaces in urban environments. The application monitors real-time occupancy of parking spaces, dynamically adjusting pricing based on demand to optimize usage and revenue. It includes a reservation module, allowing users to book parking spaces for specific time slots, ensuring availability upon arrival. Key components of the application include real-time occupancy monitoring, which tracks and updates the status of parking spaces as vehicles enter and exit, providing accurate availability information. The dynamic pricing adjustment mechanism automatically modifies parking fees based on occupancy levels, encouraging balanced usage and maximizing revenue. The reservation module handles booking requests, confirms reservations, and updates the availability status, guaranteeing reserved spots for users. The application is designed to be scalable and adaptable to various parking lot sizes and configurations. It employs advanced programming techniques such as object-oriented design, data structures, and algorithms to ensure robustness, efficiency, and reliability. By

integrating these features, the Java application offers a comprehensive and innovative solution to modern parking management challenges, enhancing convenience for users and operational efficiency for parking lot managers.

## 1.3 JAVA PROGRAMMING CONCEPTS

**The basic concepts of Object-Oriented Programming (OOP) are:**

- ✓ **Class and Object:** A class is a blueprint, and an object is an instance of the class.
- ✓ **Encapsulation:** Fields in the ParkingSpace class are encapsulated within the class, with access managed through the class constructor and methods.
- ✓ **Inheritance:** Enables a class (child) to inherit properties and methods from another class (parent), promoting code reuse.
- ✓ **Polymorphism:** Allows methods to perform differently based on the object context (e.g., method overloading and overriding).
- ✓ **Abstraction:** Hides implementation details and exposes only essential features, simplifying system design.

### Project related concepts

#### 1. Swing for GUI Components:

- **JFrame:** Used to create the main window.
- **JTextArea:** Used to display the parking status and messages.
- **TextField:** Used to get user input for date, time slot, and space to vacate.
- **JComboBox:** Used for selecting vehicle type and parking space.
- **JButton:** Used for actions like reserving and vacating parking spaces.
- **JLabel:** Used to label text fields and combo boxes.

#### 2. Method Calls:

- The code defines and calls several methods to perform specific tasks, such as `reserveParkingSpace()`, `vacateParkingSpace()`, and `updateParkingStatus()`.

#### 3. Collections Framework:

- **HashMap:** Used to store the parking spaces, where the key is a string (space identifier) and the value is a `ParkingSpace` object.

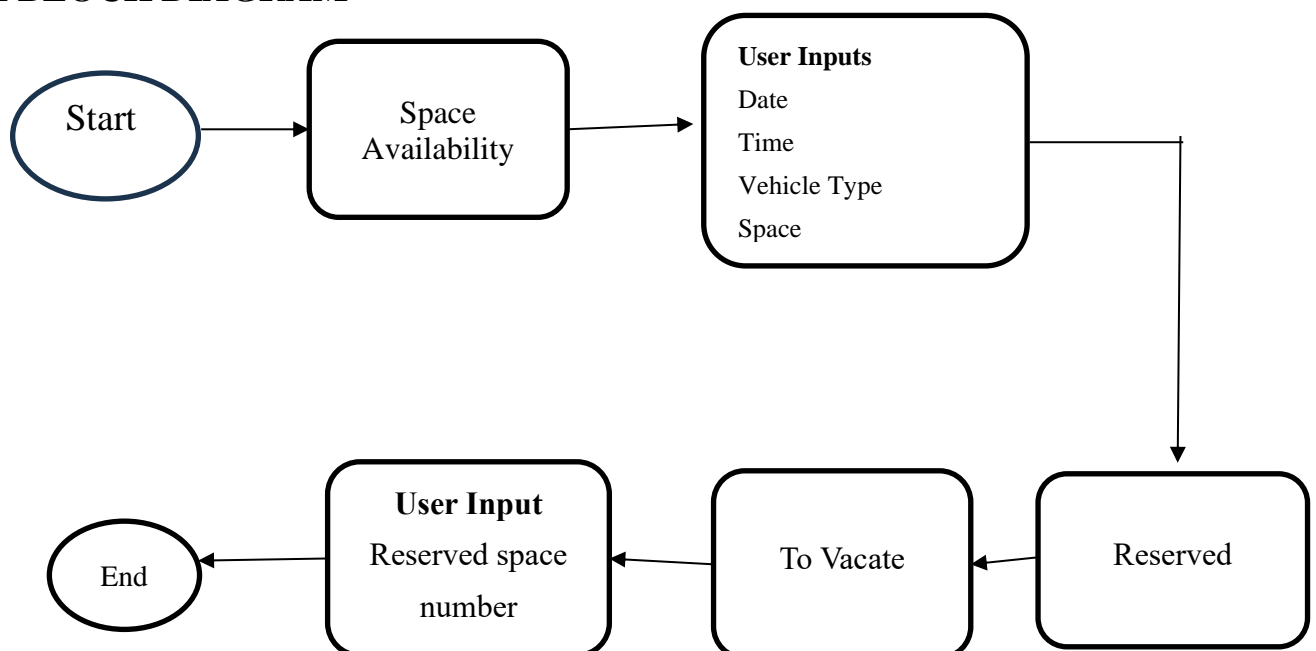
## CHAPTER 2

### PROJECT METHODOLOGY

#### 2.1 PROPOSED WORK

The proposed work for the Smart Parking System aims to develop a comprehensive Java-based parking space reservation system to enhance the management of parking spaces in urban environments. The application monitors real-time occupancy, dynamically adjusting pricing based on demand to optimize space usage and revenue. It features a user-friendly graphical interface that allows users to view available parking spaces, select vehicle types, and reserve spaces for specific time slots. Additionally, users can vacate spaces, updating their status to available. The dynamic pricing model adjusts fees based on occupancy levels and vehicle types, ensuring balanced utilization. The system employs advanced programming techniques, including object-oriented design and efficient data structures, to ensure robustness and scalability. This application is adaptable to various parking lot sizes and configurations, making it a versatile tool for different environments. Real-time updates provide users with the latest information on availability and pricing, enhancing convenience. By integrating real-time monitoring, dynamic pricing, and efficient reservation capabilities, this Java application offers a practical, innovative solution to modern parking management challenges, improving user experience and operational efficiency.

#### 2.2 BLOCK DIAGRAM



## CHAPTER 3

### MODULE DESCRIPTION

#### 3.1 REAL-TIME OCCUPANCY MONITORING

- **Functionality:** This module tracks and updates the occupancy status of each parking space in real-time as vehicles enter and exit. It provides accurate and current information on space availability.
- **Components:** Sensors or manual input to detect vehicle presence, data processing unit to update occupancy status, and a display system to show current availability.

#### 3.2 RESERVATION MODULE

- **Functionality:** Allows users to book parking spaces for specific time slots. It handles booking requests, confirms reservations, and updates the availability of spaces.
- **Components:** Reservation form (user input), booking confirmation system, and an updated availability tracker

#### 3.3 PARKING SPACE STATUS

- **Functionality:** Updates the status of parking spaces based on reservations and vacating actions. Ensures the system reflects the current state of each parking space.
- **Components:** Status update logic, reservation details, and a user interface to vacate spaces.

#### 3.4 USER INTERFACE

- **Functionality:** Updates the status of parking spaces based on reservations and vacating actions. Ensures the system reflects the current state of each parking space.
- **Components:** Status update logic, reservation details, and a user interface to vacate spaces.

## **CHAPTER 4**

### **CONCLUSION & FUTURE SCOPE**

#### **4.1 CONCLUSION**

In conclusion, Smart Parking System successfully develops a Java-based application aimed at enhancing the management of parking spaces in urban settings. By incorporating real-time occupancy monitoring, dynamic pricing adjustments, and a comprehensive reservation system, the application addresses the need for efficient and user-friendly parking management solutions. The intuitive graphical user interface ensures ease of use, while advanced programming techniques guarantee the system's robustness, scalability, and adaptability to various parking lot sizes and configurations. Ultimately, this innovative solution improves the user experience and operational efficiency, offering a practical approach to modern parking management challenges. The project demonstrates the effective application of Java programming in solving real-world problems, highlighting its potential for further advancements and implementations in diverse environments.

#### **4.2 FUTURE SCOPE**

The future scope of this Java-based parking space reservation system is expansive, with numerous potential enhancements and expansions to elevate its functionality, user experience, and adaptability. Integrating IoT sensors can automate real-time occupancy detection, enhancing accuracy and reducing manual input. Developing mobile applications for Android and iOS will allow users to manage parking reservations on-the-go. Implementing advanced data analytics can provide insights into usage patterns, peak times, and revenue trends, aiding in operational and pricing optimization. Enhancing dynamic pricing algorithms with machine learning can predict demand more accurately, adjusting prices in real-time. Integration with GPS and navigation systems will offer real-time directions to reserved spaces, improving user convenience. Adding secure online payment options for reservation fees will streamline the process. User account management features can enable profile creation, reservation history viewing, and efficient booking management. Scaling the system to handle larger facilities, such as multi-level garages and airport parking, is another avenue. Supporting green initiatives by reserving spaces for electric vehicles and prioritizing eco-friendly cars can promote sustainability. These advancements will render the parking reservation system more robust, user-friendly, and adaptable, addressing modern parking management challenges comprehensively.

## REFERENCES

### Java Books:

#### 1. "Head First Java" by Kathy Sierra and Bert Bates

This book is a great resource for beginners learning Java, with a focus on object-oriented programming concepts and real-world application development.

#### 2. "Effective Java" by Joshua Bloch

A deeper dive into best practices for writing clean, maintainable Java code. It covers advanced topics like Java collections, concurrency, and design patterns that could be applied to more complex payroll systems.

### Websites:

#### 1. GeeksforGeeks - Java Tutorials

- URL: <https://www.geeksforgeeks.org/java/>
- A comprehensive collection of tutorials on Java, covering topics like classes, objects, inheritance, encapsulation, and more. Great for learning the core concepts of Java and applying them in projects like EPMS.

#### 2. W3Schools - Java Tutorial

- URL: <https://www.w3schools.com/java/>
- A beginner-friendly resource that offers tutorials on Java programming, including object-oriented principles and core Java concepts.

### YouTube Links:

#### 1. Java for Beginners - Java Brains

- URL: <https://www.youtube.com/user/koushks>
- Offers Java tutorials from the basics to advanced concepts. The channel provides detailed guides on Java programming, including working with objects and classes, which are crucial for building an EPMS.



## APPENDIX A (SOURCE CODE)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.HashMap;
import java.util.Map;

public class ParkingSystem {
    private static Map<String, ParkingSpace> parkingSpaces = new HashMap<>();
    private static JFrame frame;
    private static JTextArea textArea;
    private static JTextField dateField;
    private static JTextField timeSlotField;
    private static JTextField vacateField;
    private static JComboBox<String> vehicleTypeBox;
    private static JComboBox<String> spaceSelectionBox;
    private static JButton reserveButton;
    private static JButton vacateButton;

    public static void main(String[] args) {
        initializeParkingSpaces();
        createUI();
    }

    // Parking space class to hold additional data
    static class ParkingSpace {
        boolean isAvailable;
        double price;
        String reservedFor;
        String vehicleType;

        ParkingSpace(boolean isAvailable, double price) {
            this.isAvailable = isAvailable;
            this.price = price;
            this.reservedFor = "";
            this.vehicleType = "";
        }
    }

    private static void initializeParkingSpaces() {
        for (int i = 1; i <= 10; i++) {
            parkingSpaces.put("Space " + i, new ParkingSpace(true, 50 + i * 5)); // Base prices vary slightly
        }
    }

    private static void createUI() {
        frame = new JFrame("Parking Space Reservation System");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(600, 500);
        frame.setLayout(new FlowLayout());

        textArea = new JTextArea(20, 50);
        textArea.setEditable(false);
        frame.add(new JScrollPane(textArea));
    }
}
```

```

dateField = new JTextField(10);
frame.add(new JLabel("Enter Date (yyyy-MM-dd:"));
frame.add(dateField);

timeSlotField = new JTextField(10);
frame.add(new JLabel("Enter Time Slot (e.g., 10:00-11:00:"));
frame.add(timeSlotField);

String[] vehicleTypes = { "Car", "Two-Wheeler", "Lorry", "Bus", "Auto" };
vehicleTypeBox = new JComboBox<>(vehicleTypes);
frame.add(new JLabel("Select Vehicle Type:"));
frame.add(vehicleTypeBox);

spaceSelectionBox = new JComboBox<>(parkingSpaces.keySet().toArray(new
String[0]));
frame.add(new JLabel("Select Parking Space:"));
frame.add(spaceSelectionBox);

reserveButton = new JButton("Reserve Parking Space");
reserveButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        reserveParkingSpace();
    }
});
frame.add(reserveButton);

vacateField = new JTextField(10);
frame.add(new JLabel("Enter Space to Vacate (e.g., Space 1:"));
frame.add(vacateField);

vacateButton = new JButton("Vacate Parking Space");
vacateButton.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        vacateParkingSpace();
    }
});
frame.add(vacateButton);

frame.setVisible(true);
updateParkingStatus();
}
private static void reserveParkingSpace() {
    String date = dateField.getText().trim();
    String timeSlot = timeSlotField.getText().trim();
    String vehicleType = (String) vehicleTypeBox.getSelectedItem();
    String selectedSpace = (String) spaceSelectionBox.getSelectedItem();

    if (date.isEmpty() || timeSlot.isEmpty() || vehicleType.isEmpty() ||
selectedSpace.isEmpty()) {
        textArea.append("Please enter valid date, time slot, vehicle type, and select a parking
space.\n");
        return;
    }
}

```

```

ParkingSpace space = parkingSpaces.get(selectedSpace);
if (!space.isAvailable) {
    textArea.append("Selected space is already occupied.\n");
    return;
}

double vehicleTypeMultiplier = getVehicleTypeMultiplier(vehicleType);
double finalPrice = space.price * vehicleTypeMultiplier;

space.isAvailable = false;
space.reservedFor = date + " " + timeSlot;
space.vehicleType = vehicleType;

textArea.append("Reserved " + selectedSpace + " for " + date + " " + timeSlot +
    " for a " + vehicleType + " at $" + finalPrice + "\n");
updateParkingStatus();
}

private static double getVehicleTypeMultiplier(String vehicleType) {
    switch (vehicleType) {
        case "Two-Wheeler":
            return 0.5;
        case "Lorry":
            return 2.0;
        case "Bus":
            return 2.5;
        case "Auto":
            return 0.75;
        case "Car":
            return 1.0;
        default:
            return 1.0;
    }
}

private static void vacateParkingSpace() {
    String spaceToVacate = vacateField.getText().trim();
    if (spaceToVacate.isEmpty() || !parkingSpaces.containsKey(spaceToVacate)) {
        textArea.append("Invalid parking space: " + spaceToVacate + "\n");
        return;
    }

    ParkingSpace space = parkingSpaces.get(spaceToVacate);
    if (!space.isAvailable) {
        space.isAvailable = true;
        space.reservedFor = "";
        space.vehicleType = "";
        textArea.append(spaceToVacate + " has been vacated.\n");
        updateParkingStatus();
    } else {
        textArea.append(spaceToVacate + " is already available.\n");
    }
}

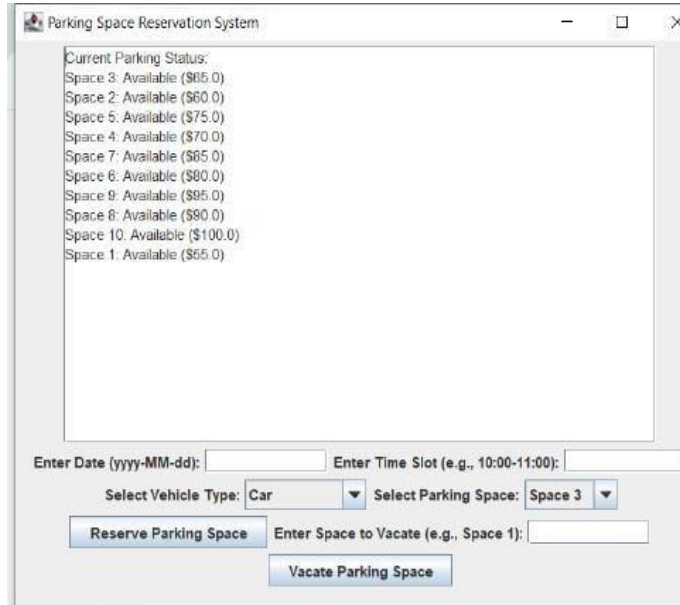
```

```

private static void updateParkingStatus() {
    textArea.setText(""); // Clear the text area before updating
    textArea.append("Current Parking Status:\n");
    for (Map.Entry<String, ParkingSpace> entry : parkingSpaces.entrySet()) {
        textArea.append(entry.getKey() + ": " +
            (entry.getValue().isAvailable ? "Available ($" + entry.getValue().price + ") " :
            "Occupied (" + entry.getValue().reservedFor + " by " +
entry.getValue().vehicleType + ")") + "\n");
        }
        textArea.append("\n");
    }
}

```

## APPENDIX B(SCREENSHOS)



**Parking Space Reservation System**

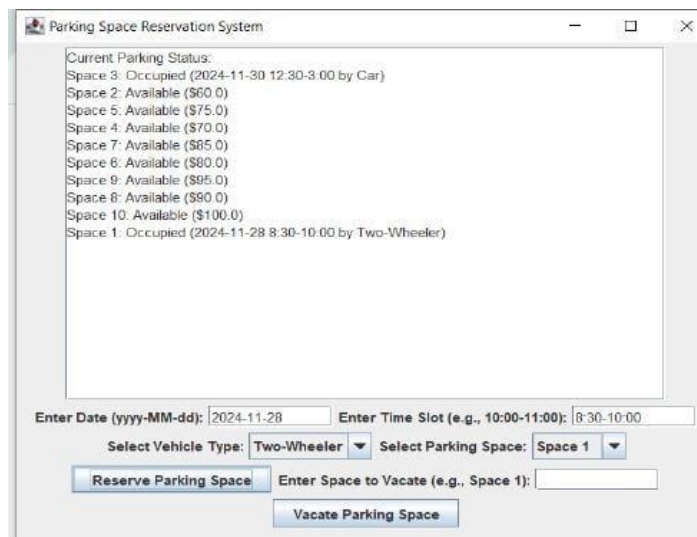
Current Parking Status:

- Space 3: Available (\$85.0)
- Space 2: Available (\$60.0)
- Space 5: Available (\$75.0)
- Space 4: Available (\$70.0)
- Space 7: Available (\$85.0)
- Space 6: Available (\$80.0)
- Space 9: Available (\$95.0)
- Space 8: Available (\$90.0)
- Space 10: Available (\$100.0)
- Space 1: Available (\$55.0)

Enter Date (yyyy-MM-dd):  Enter Time Slot (e.g., 10:00-11:00):

Select Vehicle Type:  Select Parking Space:

Enter Space to Vacate (e.g., Space 1):



**Parking Space Reservation System**

Current Parking Status:

- Space 3: Occupied (2024-11-30 12:30-3:00 by Car)
- Space 2: Available (\$60.0)
- Space 5: Available (\$75.0)
- Space 4: Available (\$70.0)
- Space 7: Available (\$85.0)
- Space 6: Available (\$80.0)
- Space 9: Available (\$95.0)
- Space 8: Available (\$90.0)
- Space 10: Available (\$100.0)
- Space 1: Occupied (2024-11-28 8:30-10:00 by Two-Wheeler)

Enter Date (yyyy-MM-dd):  Enter Time Slot (e.g., 10:00-11:00):

Select Vehicle Type:  Select Parking Space:

Enter Space to Vacate (e.g., Space 1):