

# HOUSE RENTAL SYSTEM



## A PROJECT REPORT

*Submitted by*

**THARANITHARAN K (2303811710421169)**

*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-621112**

**NOVEMBER- 2024**

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

**SAMAYAPURAM-621112**

**BONAFIDE CERTIFICATE**

Certified that this project report on “**HOUSE RENTAL SYSTEM**” is the bonafide work of **THARANITHARAN K (2303811710421169)** 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. A. MALARMANNAN A, M.E.,  
ASSISTANT PROFESSOR

**SIGNATURE**

Mr. A. Malarmannan, M.E.,

**SUPERVISOR**

**ASSISTANT PROFESSOR**

Department of CSE

K.Ramakrishnan College of Technology  
(Autonomous)

Samayapuram-621112.

Submitted for the viva-voce examination held on 06/12/2024.

CGB1201-JAVA PROGRAMMING  
Mr. M. RAJANAN, M.E.,  
EXTERNAL EXAMINER  
ASSISTANT PROFESSOR

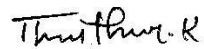
**INTERNAL EXAMINER**

CGB1201-JAVA PROGRAMMING  
Mr. R. K. SATHI, M.E.,  
EXTERNAL EXAMINER  
ASSISTANT PROFESSOR  
8138-SCE, TRICHY.

**EXTERNAL EXAMINER**

## DECLARATION

I declare that the project report on “**HOUSE RENTAL 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**

**THARANITHARAN K**

---

Place: Samayapuram

Date: 06/12/2024

## 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 adequateduration 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 supervisor **MR. A. MALARMANNAN,M.E.**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patiencewhich 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**

The House Rental System is a comprehensive Java-based application designed to simplify the property rental process. It allows users to search for available properties, book rentals, and manage their reservations efficiently. By utilizing core Object-Oriented Programming (OOP) concepts such as classes, objects, inheritance, polymorphism, and encapsulation, the system offers a modular and maintainable structure. Key features include property search, booking management, payment processing, and error handling, ensuring a seamless user experience. The system's future scope includes advanced features like AI-based recommendations, dynamic pricing, mobile app integration, and enhanced property management tools. This project serves as a practical example of Java programming in real-world application development.

.



### ABSTRACT WITH POs AND PSOs MAPPING

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

ABSTRACT	POs MAPPED	PSOs MAPPED
The House Rental System is an innovative software application designed to simplify and streamline the process of renting properties for both tenants and rental agencies. This system provides tenants with a user-friendly platform to search for available properties, view rental details, and make reservations. Simultaneously, it empowers rental agencies to manage their properties efficiently, track bookings, and handle payments securely. With features such as an intuitive interface, property management tools, and online payment integration, the system ensures convenience, transparency, and reliability for all users.	<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>	viii
<b>1</b>	<b>INTRODUCTION</b>	
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming concepts	2
<b>2</b>	<b>PROJECT METHODOLOGY</b>	
	2.1 Proposed Work	2
	2.2 Block Diagram	3
<b>3</b>	<b>MODULE DESCRIPTION</b>	
	3.1 User Interface (UI)Module	4
	3.2House rental system Module	4
	3.3House rental system and Processing Module	4
	3.4ErrorHandlingandValidationModule	4
	3.5SystemConfigurationand Maintenance Module	4
<b>4</b>	<b>CONCLUSION &amp; FUTURE SCOPE</b>	
	4.1 Conclusion	5
	4.2 Future Scope	5
	<b>APPENDIX A (SOURCE CODE)</b>	6
	<b>APPENDIX B (SCREENSHOTS)</b>	10
	<b>REFERENCES</b>	11

# CHAPTER 1

## INTRODUCTION

### 1.1 Objective

The objective of the House Rental System is to simplify the process of renting properties by providing a platform that streamlines tasks for both tenants and property rental agencies. It allows users to search for available properties, book rentals, and manage reservations through an intuitive interface. For rental agencies, the system offers efficient property management, booking tracking, and payment processing tools. By using modern software practices, the House Rental System provides a seamless experience that is both accessible and reliable

### 1.2 Overview

The House Rental System is a Java-based application designed to make the property rental process straightforward and efficient. It enables tenants to search for rental properties, compare options, and confirm bookings, while property rental agencies can manage their listings and track bookings.

#### **Key Features:**

**Property Search:** Tenant scan input criteria such as location, proper type, or rental dates to find available properties.

**Booking Management:** Once a property is selected, the system facilitates instant booking with real-time availability updates.

**Property Tracking:** Rental agencies can update property availability, track bookings, and schedule maintenance.

**Secure Payments:** Tenant scan make payments through integrated, secure gate ways.

## 1.3 Java Programming Concept

### Object-Oriented Programming(OOPS)Concepts:

Object-Oriented Programming (OOP) is a programming paradigm based on the concept of "objects", which contain both data (attributes) and methods (functions). Java, being an object-oriented language, follows the OOP principles to structure and design software. Below are the key OOP concepts in Java: follows the OOP principles to structure and design software. Below are the key OOP concepts in Java:

#### 1. Class and Object

**Class:** A class is a blue print or template for creating objects. It defines the properties (fields) and behaviors (methods) that objects created from the class will have.

**Object:** An object is an instance of a class. It represents a real-world entity with attributes (state) and methods (behavior).

#### 2. Encapsulation

Encapsulation is the concept of bundling data (variables) and methods (functions) that operate on the data into a single unit (class). It also restricts direct access to some of the object's components to protect the integrity of the data, typically using private access modifiers and providing public getter and setter methods.

#### 3. Inheritance

Inheritance allows a new class (subclass) to inherit properties and behaviors (methods) from an class (superclass). It promotes code reusability and establishes an "is-a" relationship between the parent and child classes.

#### 4. Polymorphism

Polymorphism means "many forms." It allows as sub class to provide a specific implementation of a method that is already defined in its super class.

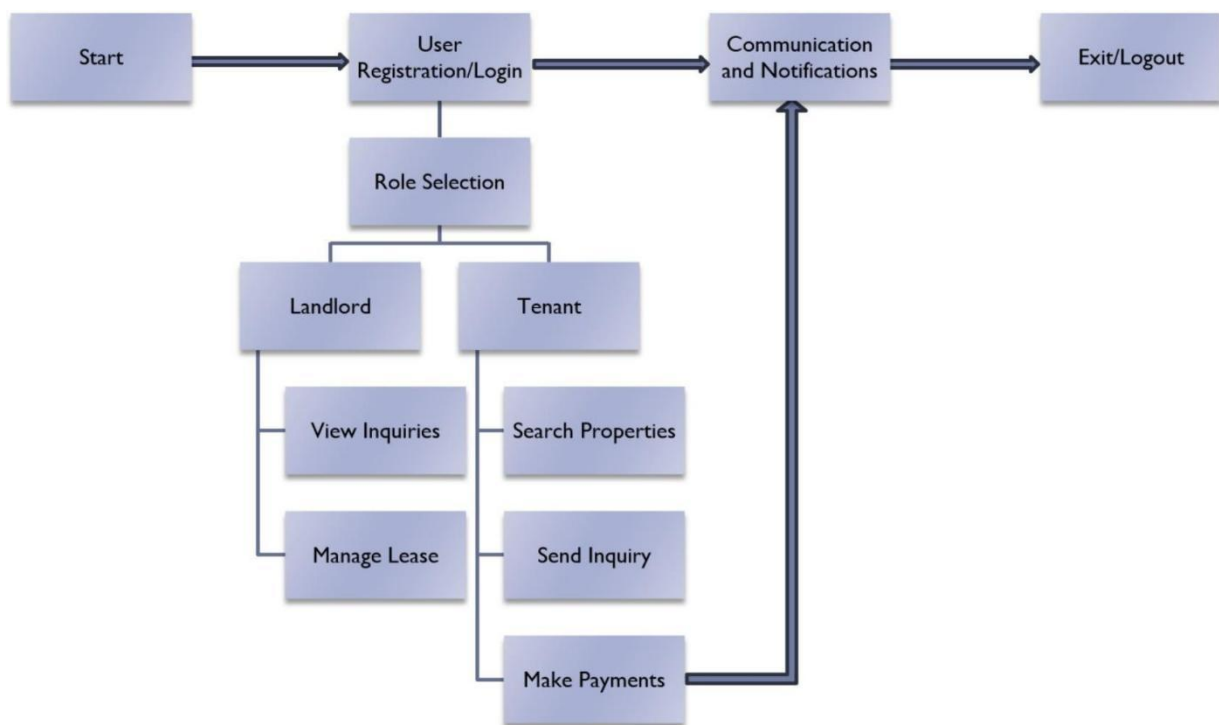
## CHAPTER 2

### PROJECT METHODOLOGY

#### 2.1 Proposed Work

1. **Objective:** Define the problem or functionality to be implemented, specifying the desired outcome.
2. **Requirements Gathering:** Identify the tools, frameworks, and resources needed to execute the code.
3. **Design & Planning:** Outline the logical structure, algorithms, and overall approach to achieve the objective.
4. **Implementation:** Write, debug, and refine the code while adhering to best practices and standards.
5. **Testing & Documentation:** Validate functionality through rigorous testing and document the code for future reference.

#### 2.2 Block Diagram



## **CHAPTER 3**

### **MODULE DESCRIPTION**

#### **3.1 User Interface Module:**

The User Interface Module is responsible for the design and implementation of the graphical user interface (GUI) of the Car Rental System. This module provides the necessary components to interact with the system, such as search fields for cars, buttons for booking actions, and display areas for rental details and user feedback.

#### **3.2 House rental system Module:**

The Car Search and Retrieval Module is responsible for making requests to the database and retrieving the list of available cars for a specified location or criteria. This module will handle queries, parse the results, and return the necessary car information to the user interface.

#### **3.3 House rental system and Processing Module:**

The Booking and Transaction Processing Module is designed to handle the raw booking requests from users and process payments. It validates the booking details, checks car availability, and confirms the transaction before updating the booking records.

#### **3.4 Error Handling and Validation Module:**

The Error Handling and Validation Module is responsible for ensuring that the system can gracefully handle errors and edge cases, such as invalid user input, failed booking requests, or payment issues. It provides meaningful error messages to the user to improve the user experience.

#### **3.5 System Configuration and Maintenance Module:**

The Error Handling and Validation Module is responsible for ensuring that the system can gracefully handle errors and edge cases, such as invalid user input, failed booking requests, or payment issues. It provides meaningful error messages to the user to improve the user experience.

## **CHAPTER 4**

### **CONCLUSION & FUTURE SCOPE**

#### **4.1 CONCLUSION**

The House Rental System streamlines the entire property rental process, making it easier for users to find, book, and manage rental properties. By incorporating intuitive features, secure payment processing, and robust error handling, the system offers a seamless and reliable experience, ensuring both users and property owners can efficiently manage rental transactions. The application's well-structured modules, including property search, booking management, and reservation tracking, enable users to quickly find suitable rental options and complete transactions with ease. With features like real-time availability updates, booking confirmation notifications, and integrated payment gateways, the system ensures transparency and security at every step. Overall, the House Rental System enhances the rental experience, reducing complexity and improving efficiency for all parties involved.

#### **4.2 FUTURE SCOPE**

The future scope of the House Rental System lies in expanding its functionality to include advanced features such as AI-powered property recommendations, dynamic pricing based on demand, and integration with smart home technologies. Enhancements like multi-language support, user reviews and ratings, and the ability to handle long-term rentals will further improve the system's accessibility and user satisfaction. Additionally, incorporating mobile app integration and geographic location-based search capabilities will provide users with greater flexibility and convenience. As the system evolves, it can leverage data analytics to offer insights into market trends and customer preferences, paving the way for smarter property management solutions.

## APPENDIX A

### (SOURCE CODE)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;

public class HouseRentalSystem extends JFrame {
    private ArrayList<House> houses;
    private DefaultListModel<String> houseListModel;

    public HouseRentalSystem() {
        houses = new ArrayList<>();
        houseListModel = new DefaultListModel<>();

        setTitle("House Rental System");
        setSize(600, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());

        JPanel topPanel = new JPanel();
        topPanel.setLayout(new GridLayout(1, 3, 5, 5));
        JLabel titleLabel = new JLabel("House Rental System", JLabel.CENTER);
        titleLabel.setFont(new Font("Arial", Font.BOLD, 20));
        topPanel.add(titleLabel);

        JPanel centerPanel = new JPanel(new BorderLayout());
        JLabel availableHousesLabel = new JLabel("Available Houses:", JLabel.LEFT);
        JList<String> houseList = new JList<>(houseListModel);
        houseList.setVisibleRowCount(10);
        JScrollPane scrollPane = new JScrollPane(houseList);

        centerPanel.add(availableHousesLabel, BorderLayout.NORTH);
        centerPanel.add(scrollPane, BorderLayout.CENTER);

        JPanel bottomPanel = new JPanel();
        JButton addHouseButton = new JButton("Add House");
        JButton searchHouseButton = new JButton("Search House");
        JButton exitButton = new JButton("Exit");

        bottomPanel.add(addHouseButton);
        bottomPanel.add(searchHouseButton);
        bottomPanel.add(exitButton);

        add(topPanel, BorderLayout.NORTH);
        add(centerPanel, BorderLayout.CENTER);
        add(bottomPanel, BorderLayout.SOUTH);

        addHouseButton.addActionListener(e -> addHouse());
        searchHouseButton.addActionListener(e -> searchHouse());
        exitButton.addActionListener(e -> System.exit(0));
    }
}
```



```

    }

    private void addHouse() {
        JTextField locationField = new JTextField();
        JTextField priceField = new JTextField();
        JTextField descriptionField = new JTextField();

        Object[] message = {
            "Location:", locationField,
            "Price:", priceField,
            "Description:", descriptionField,
        };

        int option = JOptionPane.showConfirmDialog(null, message, "Add New House",
        JOptionPane.OK_CANCEL_OPTION);
        if (option == JOptionPane.OK_OPTION) {
            String location = locationField.getText();
            String price = priceField.getText();
            String description = descriptionField.getText();

            if (!location.isEmpty() && !price.isEmpty() && !description.isEmpty()) {
                House house = new House(location, price, description);
                houses.add(house);
                houseListModel.addElement(house.toString());
                JOptionPane.showMessageDialog(this, "House added successfully!");
            } else {
                JOptionPane.showMessageDialog(this, "Please fill all fields.", "Error",
                JOptionPane.ERROR_MESSAGE);
            }
        }
    }

    private void searchHouse() {
        String location = JOptionPane.showInputDialog(this, "Enter location to search:");
        if (location != null && !location.isEmpty()) {
            StringBuilder result = new StringBuilder("Search Results:\n");
            boolean found = false;

            for (House house : houses) {
                if (house.getLocation().equalsIgnoreCase(location)) {
                    result.append(house).append("\n");
                    found = true;
                }
            }

            if (found) {
                JOptionPane.showMessageDialog(this, result.toString());
            } else {
                JOptionPane.showMessageDialog(this, "No houses found in the given location.");
            }
        } else {
            JOptionPane.showMessageDialog(this, "Location cannot be empty.", "Error",
            JOptionPane.ERROR_MESSAGE);
        }
    }
}

```

```

public static void main(String[] args) {
    SwingUtilities.invokeLater(() -> {
        HouseRentalSystem frame = new HouseRentalSystem();
        frame.setVisible(true);
    });
}

class House {
    private String location;
    private String price;
    private String description;

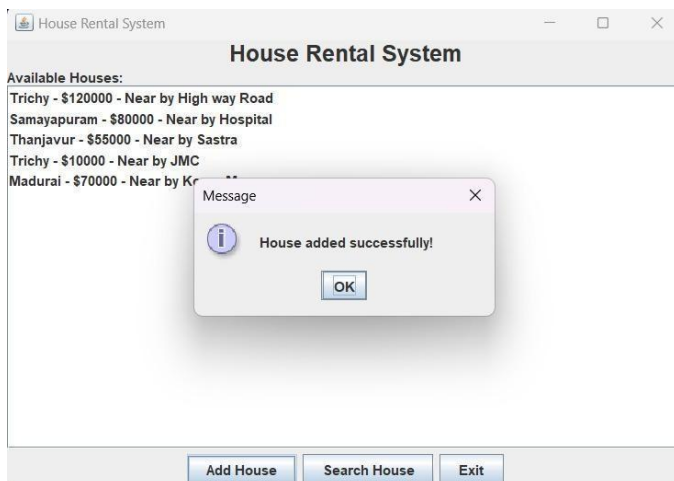
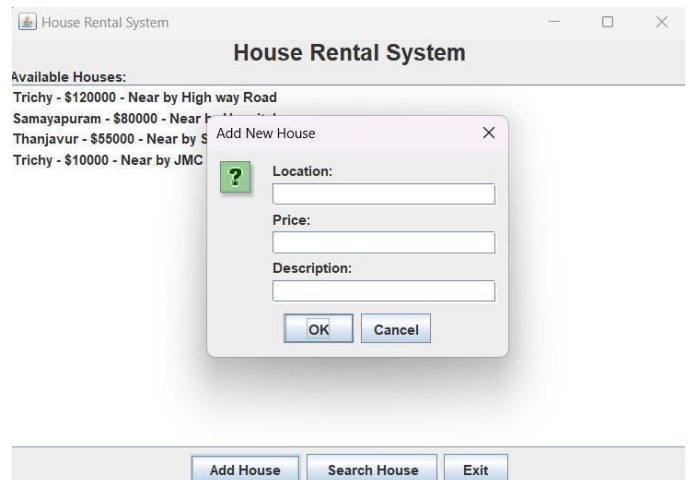
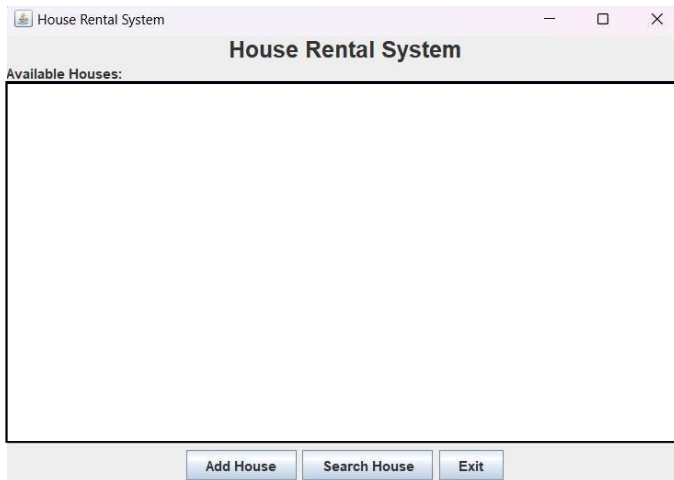
    public House(String location, String price, String description) {
        this.location = location;
        this.price = price;
        this.description = description;
    }

    public String getLocation() {
        return location;
    }

    @Override
    public String toString() {
        return location + " - $" + price + " - " + description;
    }
}

```

## APPENDIX B



## REFERENCES

1. Herbert Schildt, Java: The Complete Reference, McGraw Hill Education, 11<sup>th</sup> Edition 2019.
2. Cay S. Horstmann, Core Java Volume I – Fundamentals, Pearson Education, 11<sup>th</sup> Edition, 2018.
3. Oracle Java Documentation: <https://docs.oracle.com/en/java>
4. GeeksforGeeks Java Tutorials: <https://www.geeksforgeeks.org/java>
5. Telusko – Java Programming Tutorials
6. Programming with Mosh – Java for Beginners