PROPERTY MANAGEMENT

A MINI-PROJECT BY:

SANJANA R 230701285

SRE VARSHA N 230701330

in partial fulfilment of the award of the degree

OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING

COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2024

BONAFIDE CERTIFICATE

Certified that this project **"PROPERTY MANAGEMENT"** is the bona fide work of **"SANJANA R, SRE VARSHA N"** who carried out the project work under my supervision.

Submitted for the practical examination held on ____________

SIGNATURE	SIGNATURE
Mr. G SARAVANA GOKUL Assistant Professor (SS), Computer Science and Engineering, Rajalakshmi Engineering College (Autonomous), Thandalam, Chennai-602105	Ms. V. JANANEE Assistant Professor (SG), Computer Science and Engineering, Rajalakshmi Engineering College (Autonomous), Thandalam, Chennai-602105
INTERNAL EXAMINER	**EXTERNAL EXAMINER**

ABSTRACT

The Property Management System is a Java-based mini-project designed to modernize and streamline property management operations. This application simplifies the management of property-related data, enhancing both accuracy and efficiency while eliminating manual record-keeping. It maintains detailed records for each property, including owner information, tenant details, lease agreements, and maintenance history, all stored securely in a centralized database.

The application integrates Java with a database backend to ensure seamless data storage and retrieval. Authorized users can add, view, update, or delete property records with robust data security ensured through user authentication. The system includes advanced search and filtering capabilities to facilitate quick retrieval of information based on specific criteria. Additionally, it offers report generation features, aiding property managers and landlords in analyzing data trends and making informed decisions.

Employing modern programming techniques and an intuitive user interface, the project demonstrates how technology can transform property management. It serves as a prototype for scalable real-world applications, promoting administrative efficiency and providing a reliable, error-free repository of critical property data.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 INTRODUCTION

Property Management System is a Java-based application designed to efficiently manage and organize property-related information. It streamlines the process of recording, storing, and retrieving details about properties, tenants, leases, and maintenance activities, ensuring smooth and effective management.

1.2 IMPLEMENTATION

The **Property Management System** project is implemented using Java Swing for the user interface and MySQL for database management.

1.3 SCOPE OF THE PROJECT

This project holds immense potential to revolutionize **property management** by offering a seamless, secure, and intuitive platform for organizing property-related information. Its core focus is to provide a centralized, efficient solution that empowers property managers, landlords, and real estate professionals to easily store, access, and update critical details about properties, tenants, leases, and maintenance, all while ensuring a smooth and professional user experience.

2. SYSTEM SPECIFICATIONS

2.1. HARDWARE SPECIFICATIONS:

PROCESSOR: Intel i5

MEMORY SIZE: 16GB

HARD DISK: 500 GB of free space

2.2. SOFTWARE SPECIFICATIONS:

PROGRAMMING LANGUAGE: Java, SQL

FRONT-END: Java Swing

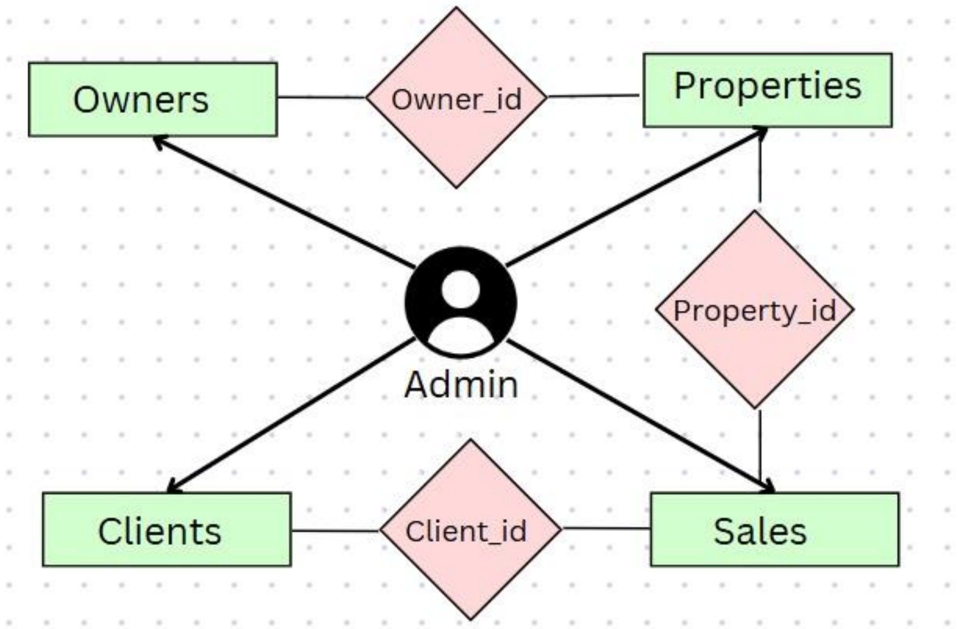
BACK-END: MySQL

OPERATING SYSTEM: Windows 11

3. ENTITY RELATION MODEL

3.1. ER DIAGRAM

An Entity Relationship (ER) diagram outlines the structure of the database by representing its entities and their relationships. In this **Property Management System**, the Admin can log in to manage key operations, such as adding property details, viewing all property records, and deleting outdated or irrelevant data. In this simplified model, there are no complex relationships between the Property entity and other entities. All essential information, such as property type, location, owner details, and rental status, is directly associated with the property record.



4. SAMPLE CODE

4.1. LOGIN PAGE:

```
package org.example;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.*;
public class LoginPage extends JFrame implements ActionListener {
    private JTextField usernameField;
    private JPasswordField passwordField;
    private JButton loginButton;
    private JButton cancelButton;
    private Image backgroundImage;
    public LoginPage() {
        setTitle("Login Page");
        setSize(700, 400);
        setLocationRelativeTo(null);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setResizable(false);
        ImageIcon icon = new
ImageIcon(getClass().getResource("/images/loginimg.png"));
        backgroundImage = icon.getImage().getScaledInstance(getWidth(), getHeight(),
Image.SCALE_SMOOTH);
        JPanel backgroundPanel = new JPanel() {
            @Override
            protected void paintComponent(Graphics g) {
                super.paintComponent(g);
                g.drawImage(backgroundImage, 0, 0, getWidth(), getHeight(), this);
                    }
        };
        backgroundPanel.setLayout(null);
        JLabel usernameLabel = new JLabel("Username:");
        usernameLabel.setBounds(200, 100, 100, 25);
        usernameLabel.setForeground(Color.WHITE);
        backgroundPanel.add(usernameLabel);
        usernameField = new JTextField();
        usernameField.setBounds(300, 100, 200, 25);
        backgroundPanel.add(usernameField);
        JLabel passwordLabel = new JLabel("Password:");
        passwordLabel.setBounds(200, 140, 100, 25);
        passwordLabel.setForeground(Color.WHITE);
        backgroundPanel.add(passwordLabel);
        passwordField = new JPasswordField();
        passwordField.setBounds(300, 140, 200, 25);
        backgroundPanel.add(passwordField);
        loginButton = new JButton("Login");
        loginButton.setBounds(250, 200, 90, 25);
        loginButton.addActionListener(this);
        backgroundPanel.add(loginButton);
        cancelButton = new JButton("Cancel");
        cancelButton.setBounds(360, 200, 90, 25);
        cancelButton.addActionListener(e -> clearFields());
        backgroundPanel.add(cancelButton);
        add(backgroundPanel);
            }
    private void clearFields() {
        usernameField.setText("");
        passwordField.setText("");
    }
    public void actionPerformed(ActionEvent e) {
        String username = usernameField.getText();
        String password = String.valueOf(passwordField.getPassword());
        if (authenticateUser(username, password)) {
            JOptionPane.showMessageDialog(this, "Login successful!");
            Dashboard dashboard = new Dashboard();
            dashboard.setVisible(true);
```

```
            dispose();
        } else {
            JOptionPane.showMessageDialog(this, "Invalid username or password.");
        }
    }
    private boolean authenticateUser(String username, String password) {
        String url = "jdbc:mysql://localhost:3306/db";
        String dbUser = "root";
        String dbPassword = "";
        boolean isAuthenticated = false;
        try (Connection conn = DriverManager.getConnection(url, dbUser,dbPassword))
{
String query = "SELECT * FROM users WHERE username = ? AND password =?";
            PreparedStatement statement = conn.prepareStatement(query);
            statement.setString(1, username);
            statement.setString(2, password);

            ResultSet resultSet = statement.executeQuery();
            if (resultSet.next()) {
                isAuthenticated = true;
            }
        } catch (Exception ex) {
            ex.printStackTrace();
        }
        return isAuthenticated;
    }
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            LoginPage loginPage = new LoginPage();
            loginPage.setVisible(true);
        });
    }
}
```

4.2. DASHBOARD DESIGN:

```
package org.example;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class Dashboard extends JFrame implements ActionListener {
    private JPanel sidebar;
    private JPanel contentPanel;
    public Dashboard() {
        setTitle("Dashboard");
        setSize(800, 500);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        setLocationRelativeTo(null);
        sidebar = new JPanel();
        sidebar.setPreferredSize(new Dimension(200, 500));
        sidebar.setBackground(new Color(1, 54, 94));         sidebar.setLayout(new
BoxLayout(sidebar, BoxLayout.Y_AXIS));
  JLabel logoLabel = new JLabel("SkyLine Estates", JLabel.CENTER);
        logoLabel.setFont(new Font("Arial", Font.BOLD, 18));
        logoLabel.setForeground(Color.WHITE);
        logoLabel.setAlignmentX(Component.CENTER_ALIGNMENT);
        sidebar.add(Box.createRigidArea(new Dimension(0, 20)));
        sidebar.add(logoLabel);
        addSidebarButton("Property");
        addSidebarButton("Images");
        addSidebarButton("Owner");
        addSidebarButton("Client");
        addSidebarButton("Sale");
        add(sidebar, BorderLayout.WEST);
        contentPanel = new JPanel();
        contentPanel.setLayout(new BorderLayout());
        add(contentPanel, BorderLayout.CENTER);
        addImageToContentPanel();

        setVisible(true);
    }

    private void addSidebarButton(String text) {
        JButton button = new JButton(text);
        button.setAlignmentX(Component.CENTER_ALIGNMENT);
        button.setMaximumSize(new Dimension(180, 40));
        button.setForeground(Color.BLACK);
        button.setBackground(new Color(198, 219, 255));
        button.setFocusPainted(false);
        button.addActionListener(this);
        sidebar.add(Box.createRigidArea(new Dimension(0, 20)));
        sidebar.add(button);
    }
    private void addImageToContentPanel() {
        ImageIcon imageIcon = new ImageIcon (getClass(). getResource
                        ("/images/dashImag e.png"));
        JLabel imageLabel = new JLabel(imageIcon);
        Image originalImage = imageIcon.getImage();
        Image scaledImage = originalImage.getScaledInstance(500, 280,
    Image.SCALE_SMOOTH);
        imageLabel.setIcon(new ImageIcon(scaledImage));
        imageLabel.setHorizontalAlignment(JLabel.CENTER);
        contentPanel.add(imageLabel, BorderLayout.CENTER);
    }
    @Override
    public void actionPerformed(ActionEvent e) {
        String action = e.getActionCommand();
        if ("Owner".equals(action)) {
            openOwnerPage();
        } else if ("Client".equals(action)) {
            openClientPage();
        } else if ("Property".equals(action)) {
            openPropertyPage();
        } else if ("Images".equals(action)) {
            openPropertyTypes();
        } else if ("Sale".equals(action)) {
            openSalesPage();
        } else {
            displayContent(action);
        }
    }
    private void openOwnerPage() {
        Owner ownerPage = new Owner();
        ownerPage.setVisible(true);
            }

    private void openClientPage() {
        Client clientPage = new Client();
        clientPage.setVisible(true);
    }
    private void openPropertyPage() {
        JFrame propertyFrame = new JFrame("Property Page");
        propertyFrame.setSize(800, 500);
        propertyFrame.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        propertyFrame.add(new PropertyPage());
        propertyFrame.setLocationRelativeTo(this);
        propertyFrame.setVisible(true);
    }
    private void openPropertyTypes() {
        JFrame propertyTypeFrame = new JFrame("Property Types");
        propertyTypeFrame.setSize(600, 400);
        propertyTypeFrame.add(new PropertyTypes());
        propertyTypeFrame.setLocationRelativeTo(this);
        propertyTypeFrame.setVisible(true);
    }
    private void openSalesPage() {
        JFrame salesPageFrame = new JFrame("Sales Management");
        salesPageFrame.setSize(800, 500);
        salesPageFrame.add(new SalesPage());
        salesPageFrame.setLocationRelativeTo(this);
        salesPageFrame.setVisible(true);

    }
    private void displayContent(String title) {
        contentPanel.removeAll();
        JLabel titleLabel = new JLabel(title, JLabel.CENTER);
        titleLabel.setFont(new Font("Arial", Font.BOLD, 24));
                contentPanel.add(titleLabel, BorderLayout.CENTER);
        contentPanel.revalidate();
        contentPanel.repaint();
    }
}
}
```

4.3. OWNER PAGE:

```
package org.example;
import javax.swing.*;
import java.awt.*;
import java.sql.*;
import java.util.Vector;

public class Owner extends JFrame {
    private JPanel ownerPanel;
    private JTable table;
    private JTextField idField, firstNameField, lastNameField, phoneField, emailField;
    private JTextArea addressArea;
    private static final String URL = "jdbc:mysql://localhost:3306/db";
    private static final String USER = "root";
```

```java
private static final String PASSWORD = "";

public Owner() {
    setTitle("Owner Management");
    setSize(800, 500);
    setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
    setLocationRelativeTo(null);

    ownerPanel = new JPanel(new BorderLayout());
    ownerPanel.setBackground(new Color(173, 216, 230));
    JPanel formPanel = new JPanel();
    formPanel.setLayout(new GridLayout(6, 2, 10, 10));
    formPanel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
    formPanel.setBackground(new Color(195, 215, 240));
    idField = new JTextField();
    firstNameField = new JTextField();
    lastNameField = new JTextField();
    phoneField = new JTextField();
    emailField = new JTextField();
    addressArea = new JTextArea();
    JScrollPane addressScrollPane = new JScrollPane(addressArea);
    Dimension textFieldSize = new Dimension(150, 25);
    idField.setPreferredSize(textFieldSize);
    firstNameField.setPreferredSize(textFieldSize);
    lastNameField.setPreferredSize(textFieldSize);
    phoneField.setPreferredSize(textFieldSize);
    emailField.setPreferredSize(textFieldSize);
    addressArea.setPreferredSize(new Dimension(150, 50));
    formPanel.add(new JLabel("ID:"));
    formPanel.add(idField);
    formPanel.add(new JLabel("First Name:"));
    formPanel.add(firstNameField);
    formPanel.add(new JLabel("Last Name:"));
    formPanel.add(lastNameField);
```

```
        formPanel.add(new JLabel("Phone:"));
        formPanel.add(phoneField);
        formPanel.add(new JLabel("Email:"));
        formPanel.add(emailField);
        formPanel.add(new JLabel("Address:"));
        formPanel.add(addressScrollPane);
        ownerPanel.add(formPanel, BorderLayout.WEST);
        table = new JTable();
        loadDataFromDatabase();
        JScrollPane tableScrollPane = new JScrollPane(table);
        ownerPanel.add(tableScrollPane, BorderLayout.CENTER);
        JPanel buttonPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, 10,
10));
        buttonPanel.setBackground(new Color(202,202,202));
        JButton addButton = new JButton("Add");
        JButton editButton = new JButton("Edit");
        JButton removeButton = new JButton("Remove");
        JButton clearButton = new JButton("Clear");
        addButton.addActionListener(e -> addRecord());
        editButton.addActionListener(e -> editRecord());
        removeButton.addActionListener(e -> removeRecord());
        clearButton.addActionListener(e -> clearFormFields());
        buttonPanel.add(addButton);
        buttonPanel.add(editButton);
        buttonPanel.add(removeButton);
        buttonPanel.add(clearButton);
        ownerPanel.add(buttonPanel, BorderLayout.SOUTH);
        table.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                int selectedRow = table.getSelectedRow();
                if (selectedRow != -1) {
                    fillFormFields(selectedRow);
                }
            }
        });
        add(ownerPanel);
    }
    private void loadDataFromDatabase() {
        Connection connection = null;
        Statement statement = null;
        ResultSet resultSet = null;

        try {
            connection = DriverManager.getConnection(URL, USER, PASSWORD);
            statement = connection.createStatement();
            resultSet = statement.executeQuery("SELECT * FROM owners");
            ResultSetMetaData metaData = resultSet.getMetaData();
            int columnCount = metaData.getColumnCount();
            Vector<String> columnNames = new Vector<>();
            for (int i = 1; i <= columnCount; i++) {
                columnNames.add(metaData.getColumnName(i));
            }
            Vector<Vector<Object>> data = new Vector<>();
            while (resultSet.next()) {
                Vector<Object> row = new Vector<>();
                for (int i = 1; i <= columnCount; i++) {
                    row.add(resultSet.getObject(i));
                }
                data.add(row);
            }
            table.setModel(new javax.swing.table.DefaultTableModel(data,
columnNames));

        } catch (SQLException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null, "Error loading data from database.",
"DatabaseError", JOptionPane.ERROR_MESSAGE);
        } finally {
            try {
```

```
            if (resultSet != null) resultSet.close();
            if (statement != null) statement.close();
            if (connection != null) connection.close();
         } catch (SQLException ex) {
            ex.printStackTrace();
         }
      }
   }

   private void fillFormFields(int selectedRow) {
      idField.setText(table.getValueAt(selectedRow, 0).toString());
      firstNameField.setText(table.getValueAt(selectedRow, 1).toString());
      lastNameField.setText(table.getValueAt(selectedRow, 2).toString());
      phoneField.setText(table.getValueAt(selectedRow, 3).toString());
      emailField.setText(table.getValueAt(selectedRow, 4).toString());
      addressArea.setText(table.getValueAt(selectedRow, 5).toString());
   }

   private void addRecord() {
      try (Connection connection = DriverManager.getConnection(URL, USER,
PASSWORD)) {
         String sql = "INSERT INTO owners (id,first_name, last_name, phone, email,
address)VALUES (?,?, ?, ?, ?, ?)";
         PreparedStatement preparedStatement = connection.prepareStatement(sql);
         preparedStatement.setInt(1, Integer.parseInt(idField.getText()));
         preparedStatement.setString(2, firstNameField.getText());
         preparedStatement.setString(3, lastNameField.getText());
         preparedStatement.setString(4, phoneField.getText());
         preparedStatement.setString(5, emailField.getText());
         preparedStatement.setString(6, addressArea.getText());
         preparedStatement.executeUpdate();
         loadDataFromDatabase();
         clearFormFields();
      } catch (SQLException e) {
         e.printStackTrace();
```

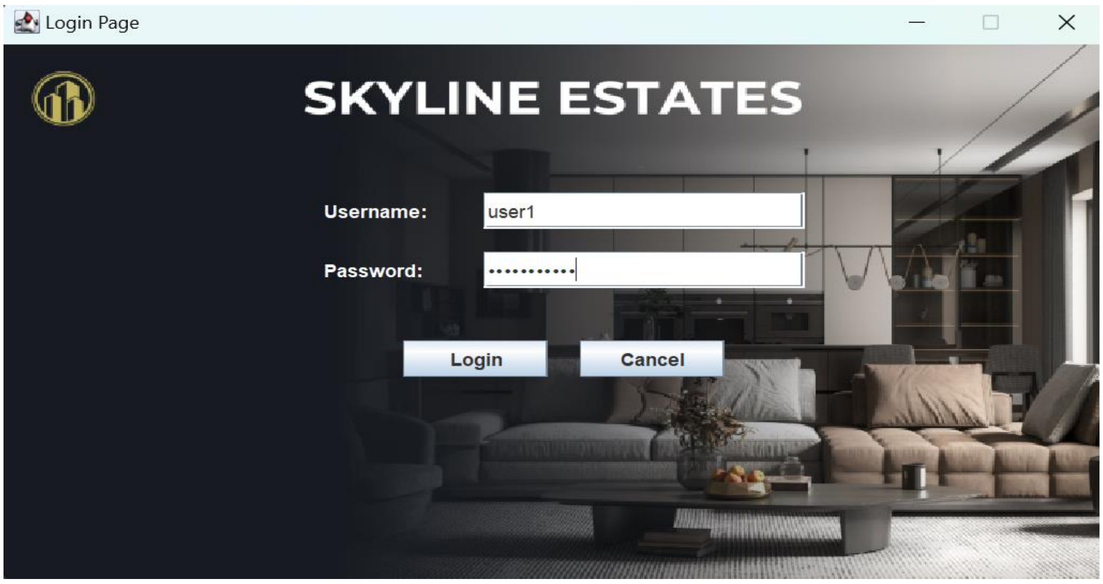
```
            JOptionPane.showMessageDialog(null, "Error adding record to database.", "DatabaseError", JOptionPane.ERROR_MESSAGE);
        }
    }

    private void editRecord() {
        try (Connection connection = DriverManager.getConnection(URL, USER, PASSWORD)) {
            String sql = "UPDATE owners SET first_name = ?, last_name = ?, phone = ?, email = ?, address = ? WHERE id = ?";
            PreparedStatement preparedStatement = connection.prepareStatement(sql);
            preparedStatement.setString(1, firstNameField.getText());
            preparedStatement.setString(2, lastNameField.getText());
            preparedStatement.setString(3, phoneField.getText());
            preparedStatement.setString(4, emailField.getText());
            preparedStatement.setString(5, addressArea.getText());
            preparedStatement.setInt(6, Integer.parseInt(idField.getText()));
            preparedStatement.executeUpdate();
            loadDataFromDatabase();
            clearFormFields();
        } catch (SQLException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null, "Error editing record in database.", "DatabaseError", JOptionPane.ERROR_MESSAGE);
        }
    }

    private void removeRecord() {
        try (Connection connection = DriverManager.getConnection(URL, USER, PASSWORD)) {
            String sql = "DELETE FROM owners WHERE id = ?";
            PreparedStatement preparedStatement = connection.prepareStatement(sql);
            preparedStatement.setInt(1, Integer.parseInt(idField.getText()));
            preparedStatement.executeUpdate();
            loadDataFromDatabase();
            clearFormFields();
        } catch (SQLException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null, "Can't delete thid owner. Delete the
owner's property first.", "Database Error", JOptionPane.ERROR_MESSAGE);
        }
    }
    private void clearFormFields() {
        idField.setText("");
        firstNameField.setText("");
        lastNameField.setText("");
        phoneField.setText("");
        emailField.setText("");
        addressArea.setText("");
    }
}
```

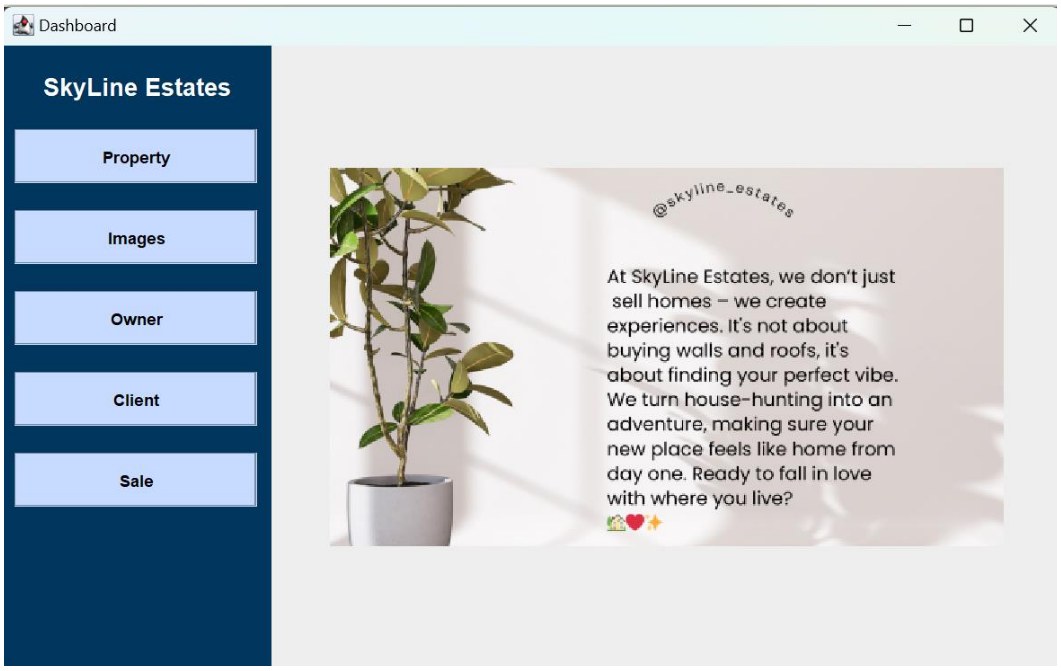
5. SNAPSHOTS

5.1. LOGIN PAGE:



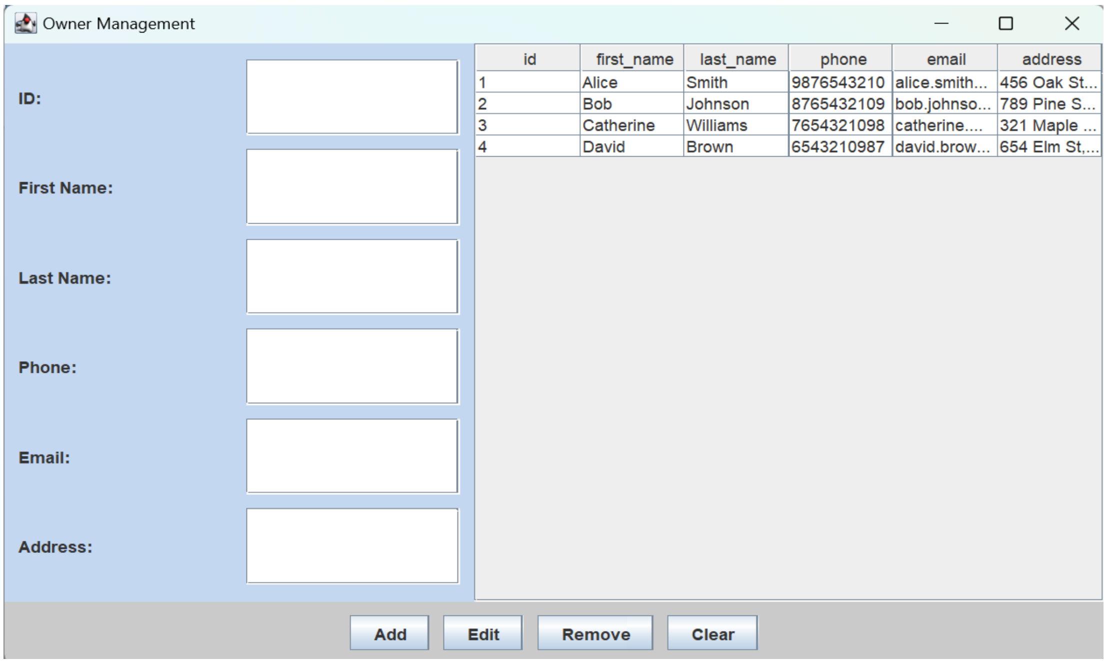
Username: user1

5.2. DASHBOARD



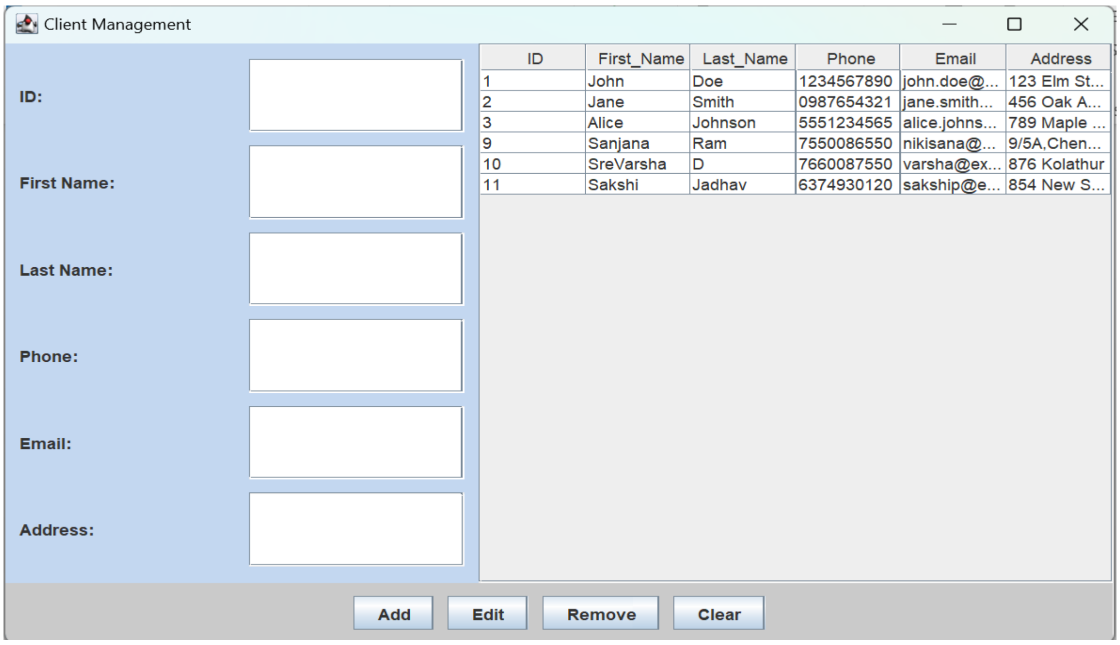
At SkyLine Estates, we don't just sell homes – we create experiences. It's not about buying walls and roofs, it's about finding your perfect vibe. We turn house-hunting into an adventure, making sure your new place feels like home from day one. Ready to fall in love with where you live? 🏡❤️✨

5.3. OWNER PAGE



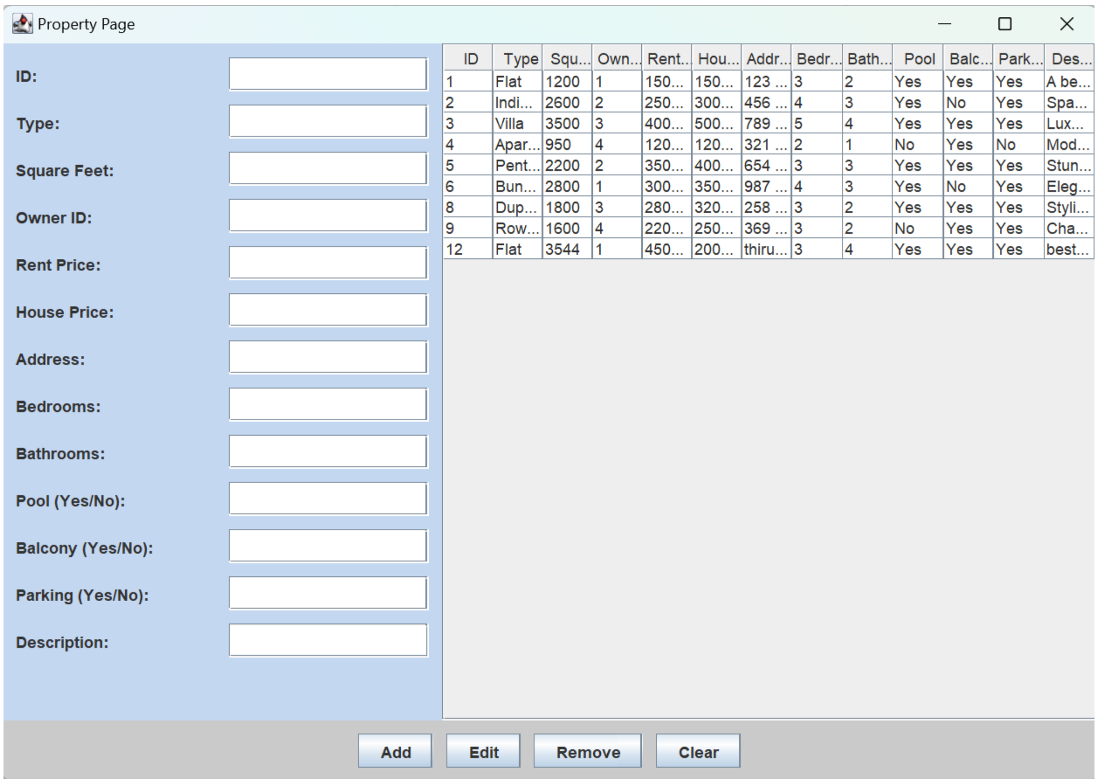
id	first_name	last_name	phone	email	address
1	Alice	Smith	9876543210	alice.smith...	456 Oak St...
2	Bob	Johnson	8765432109	bob.johnso...	789 Pine S...
3	Catherine	Williams	7654321098	catherine....	321 Maple ...
4	David	Brown	6543210987	david.brow...	654 Elm St,...

5.4. CLIENT PAGE



ID	First_Name	Last_Name	Phone	Email	Address
1	John	Doe	1234567890	john.doe@...	123 Elm St...
2	Jane	Smith	0987654321	jane.smith...	456 Oak A...
3	Alice	Johnson	5551234565	alice.johns...	789 Maple ...
9	Sanjana	Ram	7550086550	nikisana@...	9/5A,Chen...
10	SreVarsha	D	7660087550	varsha@ex...	876 Kolathur
11	Sakshi	Jadhav	6374930120	sakship@e...	854 New S...

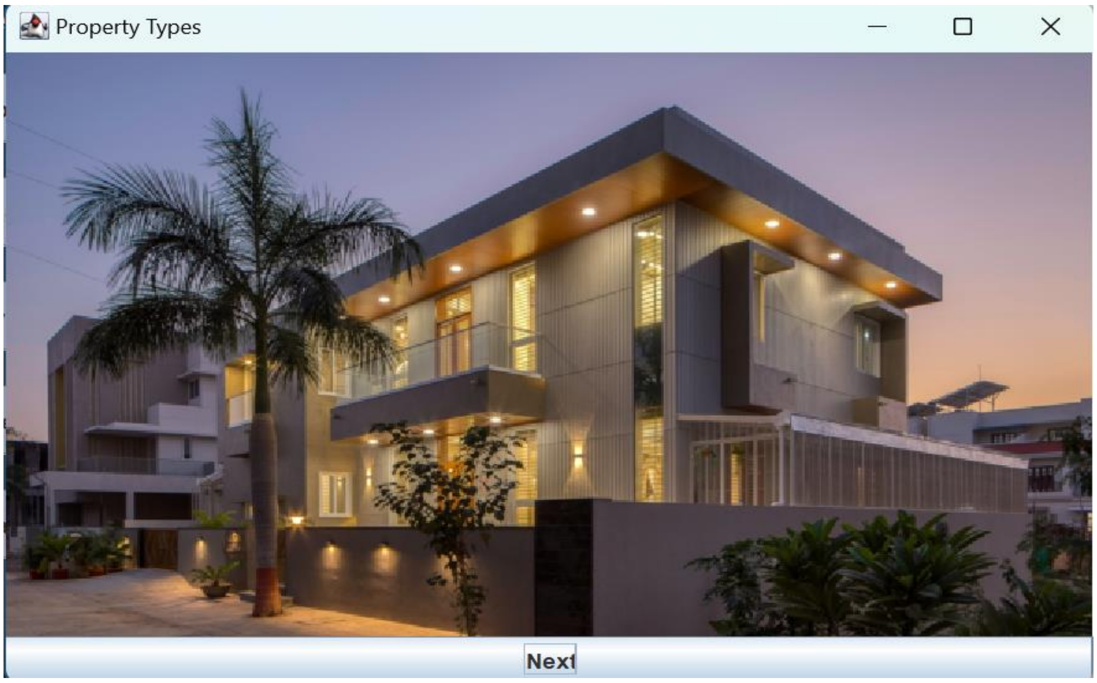
5.5. PROPERTY PAGE



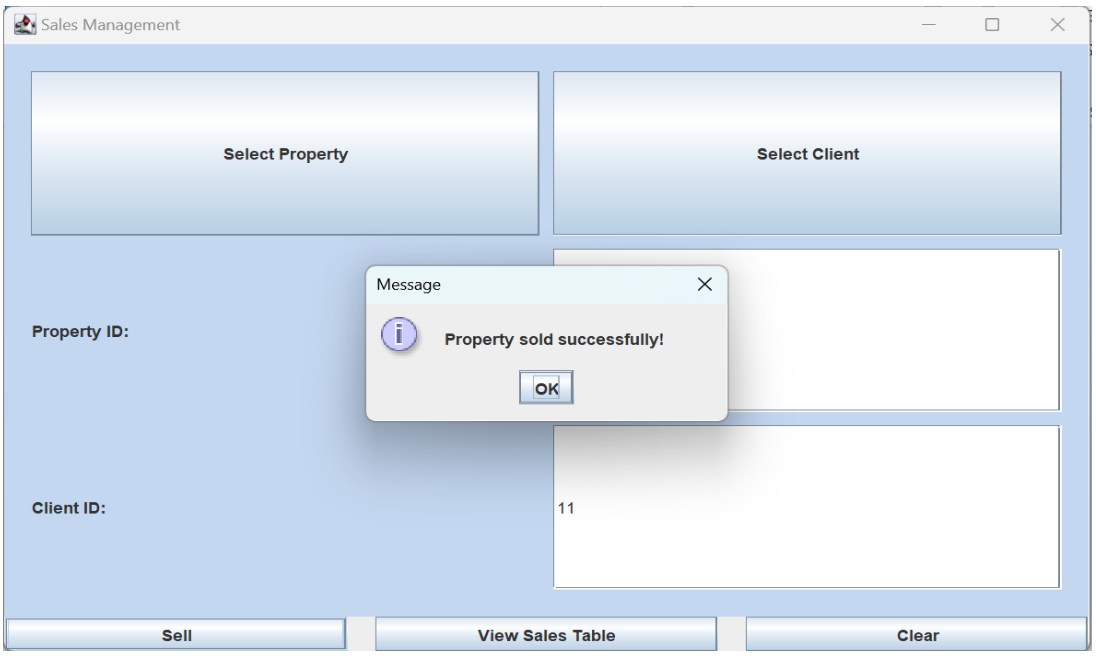
ID:	
Type:	
Square Feet:	
Owner ID:	
Rent Price:	
House Price:	
Address:	
Bedrooms:	
Bathrooms:	
Pool (Yes/No):	
Balcony (Yes/No):	
Parking (Yes/No):	
Description:	

ID	Type	Squ...	Own...	Rent...	Hou...	Addr...	Bedr...	Bath...	Pool	Balc...	Park...	Des...
1	Flat	1200	1	150...	150...	123 ...	3	2	Yes	Yes	Yes	A be...
2	Indi...	2600	2	250...	300...	456 ...	4	3	Yes	No	Yes	Spa...
3	Villa	3500	3	400...	500...	789 ...	5	4	Yes	Yes	Yes	Lux...
4	Apar...	950	4	120...	120...	321 ...	2	1	No	Yes	No	Mod...
5	Pent...	2200	2	350...	400...	654 ...	3	3	Yes	Yes	Yes	Stun...
6	Bun...	2800	1	300...	350...	987 ...	4	3	Yes	No	Yes	Eleg...
8	Dup...	1800	3	280...	320...	258 ...	3	2	Yes	Yes	Yes	Styli...
9	Row...	1600	4	220...	250...	369 ...	3	2	No	Yes	Yes	Cha...
12	Flat	3544	1	450...	200...	thiru...	3	4	Yes	Yes	Yes	best...

5.6. PROPERTY TYPES- IMAGES



5.7. SALES PAGE



Message

Property sold successfully!

OK

11

CONCLUSION

The project offers a comprehensive solution for efficient **property management**. By automating various tasks, this system enhances the accuracy and efficiency of property management operations. It provides a centralized platform for storing, retrieving, and analysing property-related data, enabling informed decision-making and streamlined management. The system features a user-friendly interface for easy data input and retrieval, ensuring a smooth experience for users.

REFERENCES

1. https://www.javatpoint.com/javatutorial

2. *https://www.wikipedia.org/*

3. *https://www.w3schools.com/sql/*