

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI-590014**



**A Report On
“JAVA ACTIVITIES”**

Submitted in partial fulfillment of the requirement in 6th Semester
ADVANCED JAVA (BCS613D)

Department of Computer Science & Engineering

By

**SHARON A DOBBIN
1AT22CS120**

**Under the Guidance of
Prof. Reena D K**

Assistant Professor, Dept. of CSE, AIT



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING ATRIA
INSTITUTE OF TECHNOLOGY
BANGALORE- 560024
2024-2025**

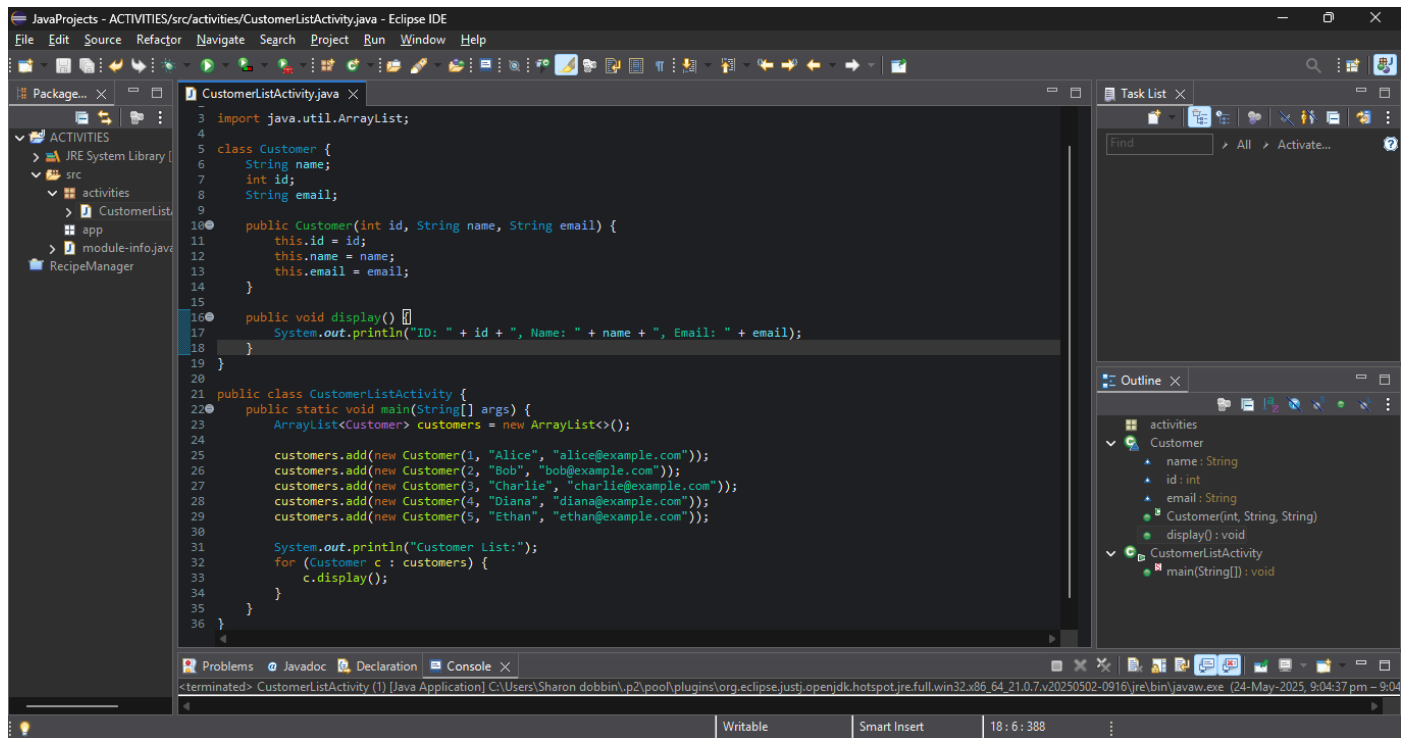
TABLE OF CONTENTS

<u>CHAPTER</u>	<u>TITLE</u>	<u>PAGE NO.</u>
ACTIVITY 1	CUSTOMER LIST	1
ACTIVITY 2	CHARACTER COUNT	2
ACTIVITY 3	PALINDROME CHECK	3
ACTIVITY 4	BUBBLE SORT	4
ACTIVITY 5	CHARACTER ANALYSIS	5
ACTIVITY 6	SIMPLE CALCULATOR	6
ACTIVITY 7	EMPLOYEE ENTRY	7 – 8
ACTIVITY 8	WEB APP	9 – 10

ACTIVITY 1: CUSTOMER LIST

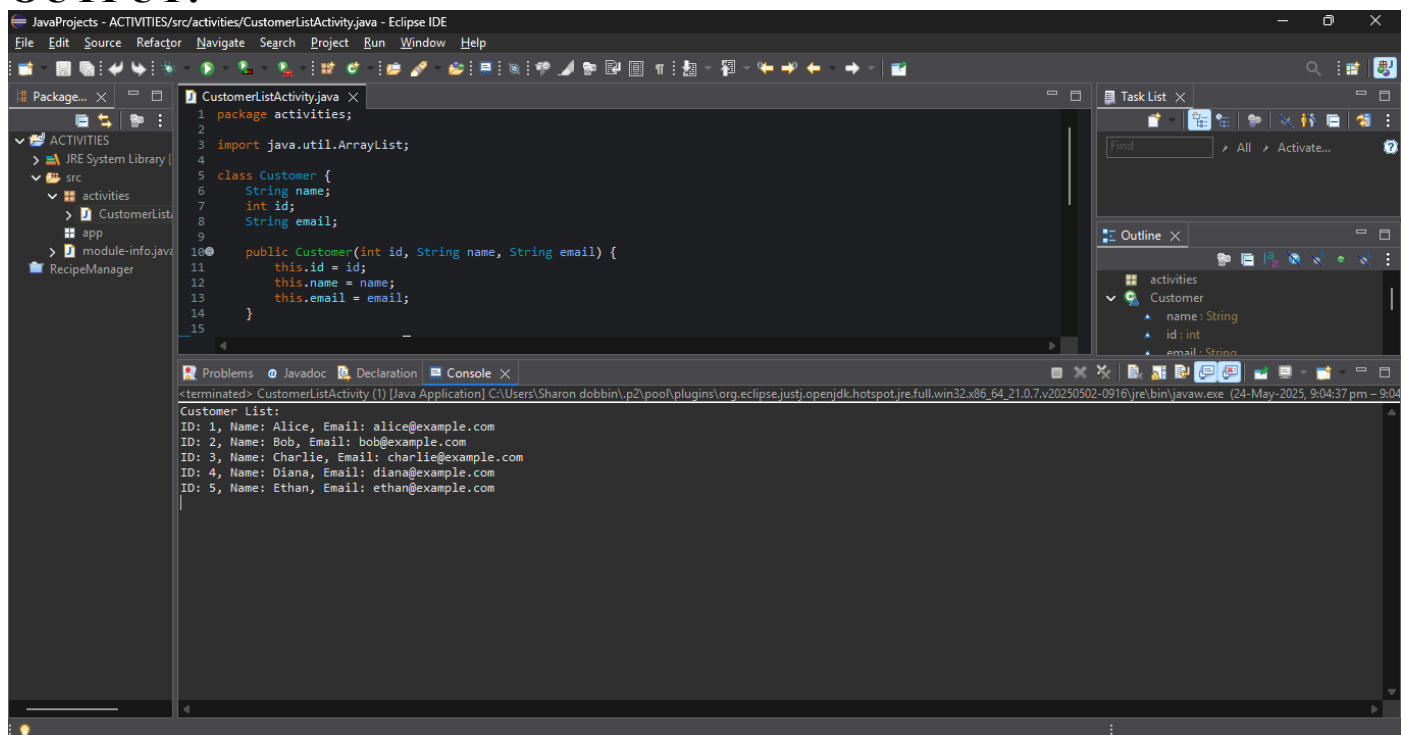
OBJECTIVE: Create a Class by name Customer, then add five objects of the Customer class into an Array List.

CODE:



```
1 import java.util.ArrayList;
2
3 class Customer {
4     String name;
5     int id;
6     String email;
7
8     public Customer(int id, String name, String email) {
9         this.id = id;
10        this.name = name;
11        this.email = email;
12    }
13
14    public void display() {
15        System.out.println("ID: " + id + ", Name: " + name + ", Email: " + email);
16    }
17 }
18
19 public class CustomerListActivity {
20     public static void main(String[] args) {
21         ArrayList<Customer> customers = new ArrayList<>();
22
23         customers.add(new Customer(1, "Alice", "alice@example.com"));
24         customers.add(new Customer(2, "Bob", "bob@example.com"));
25         customers.add(new Customer(3, "Charlie", "charlie@example.com"));
26         customers.add(new Customer(4, "Diana", "diana@example.com"));
27         customers.add(new Customer(5, "Ethan", "ethan@example.com"));
28
29         System.out.println("Customer List:");
30         for (Customer c : customers) {
31             c.display();
32         }
33     }
34 }
35
36 }
```

OUTPUT:

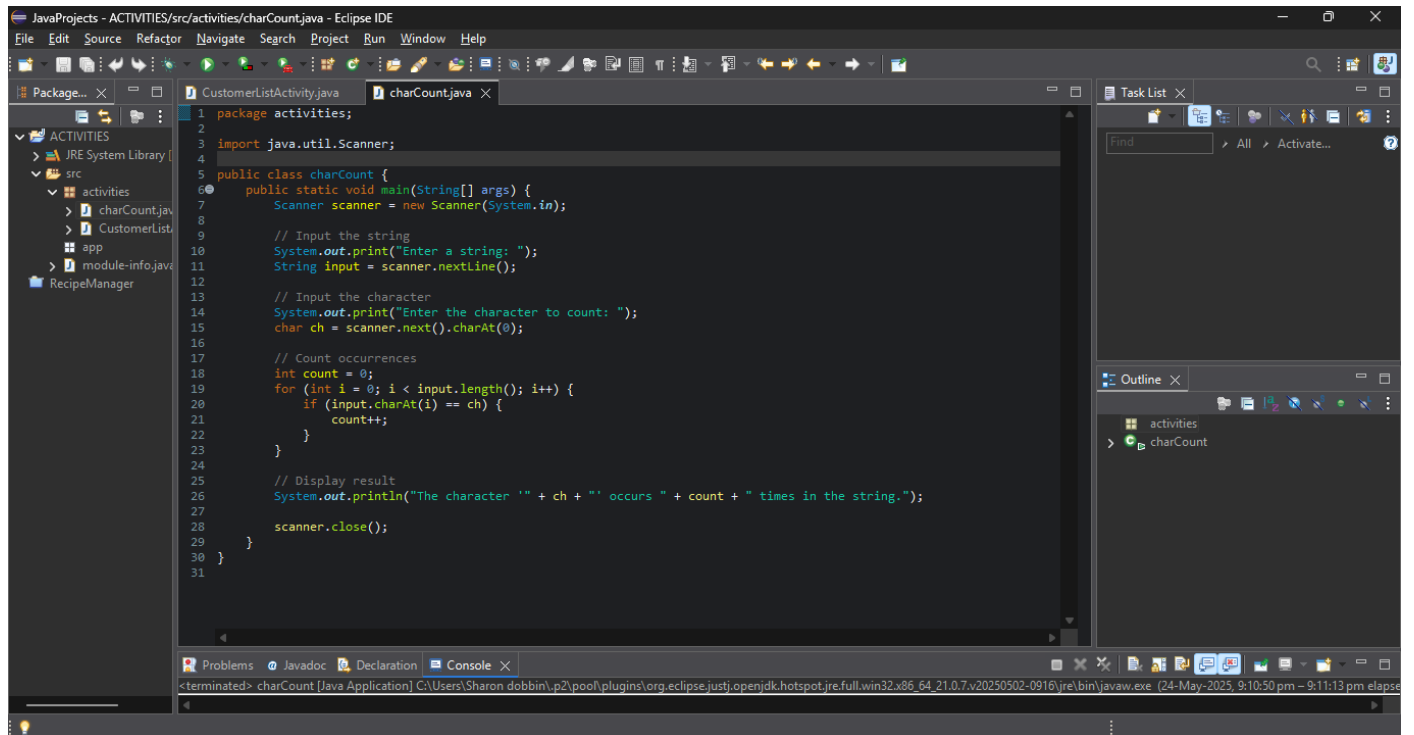


```
<terminated> CustomerListActivity (1) [Java Application] C:\Users\Sharon\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.21.0.7.v20250502-0916\jre\bin\javaw.exe (24-May-2025, 9:04:37 pm - 9:04:37 pm)
Customer List:
ID: 1, Name: Alice, Email: alice@example.com
ID: 2, Name: Bob, Email: bob@example.com
ID: 3, Name: Charlie, Email: charlie@example.com
ID: 4, Name: Diana, Email: diana@example.com
ID: 5, Name: Ethan, Email: ethan@example.com
```

ACTIVITY 2: CHARACTER COUNT

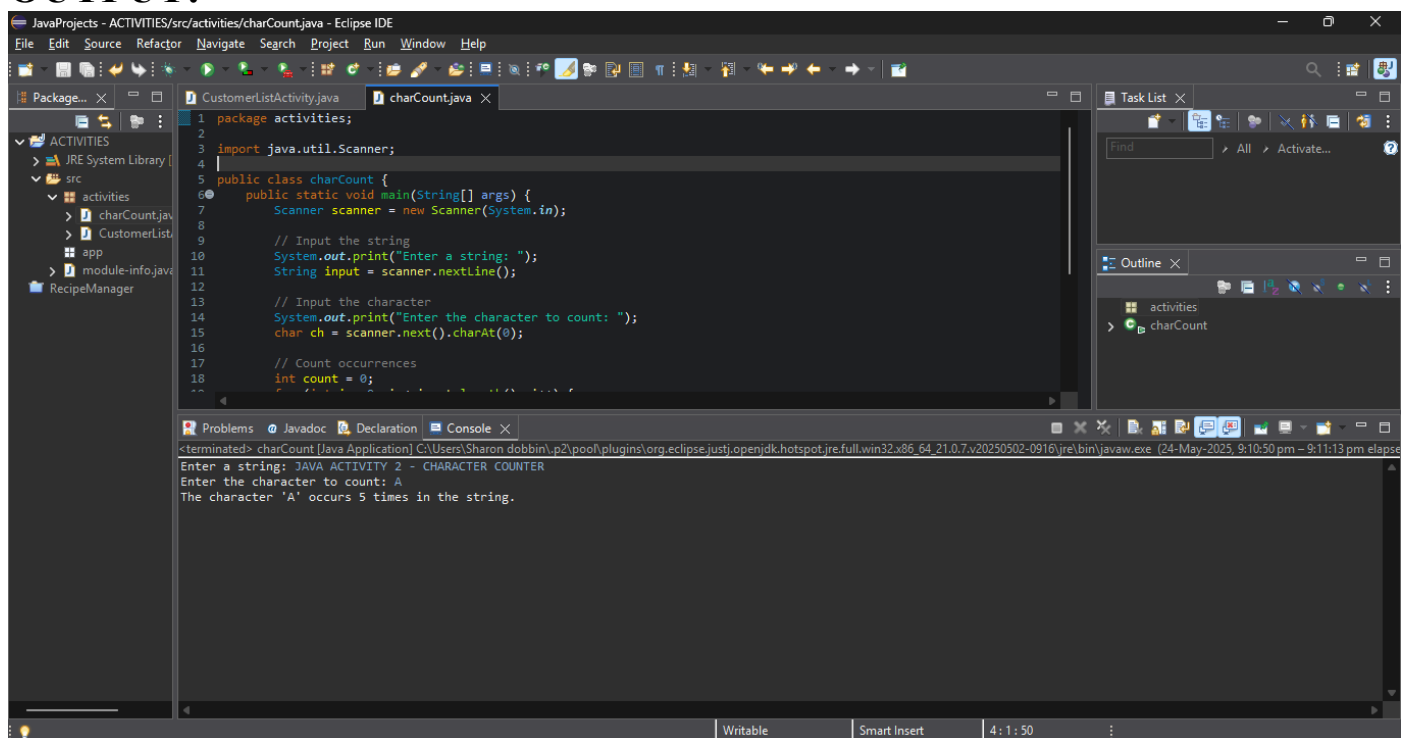
OBJECTIVE: Write a program to find how many times a particular character occurs in a string.

CODE:



```
1 package activities;
2
3 import java.util.Scanner;
4
5 public class charCount {
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8
9         // Input the string
10        System.out.print("Enter a string: ");
11        String input = scanner.nextLine();
12
13        // Input the character
14        System.out.print("Enter the character to count: ");
15        char ch = scanner.next().charAt(0);
16
17        // Count occurrences
18        int count = 0;
19        for (int i = 0; i < input.length(); i++) {
20            if (input.charAt(i) == ch) {
21                count++;
22            }
23        }
24
25        // Display result
26        System.out.println("The character '" + ch + "' occurs " + count + " times in the string.");
27        scanner.close();
28    }
29 }
30
31
```

OUTPUT:

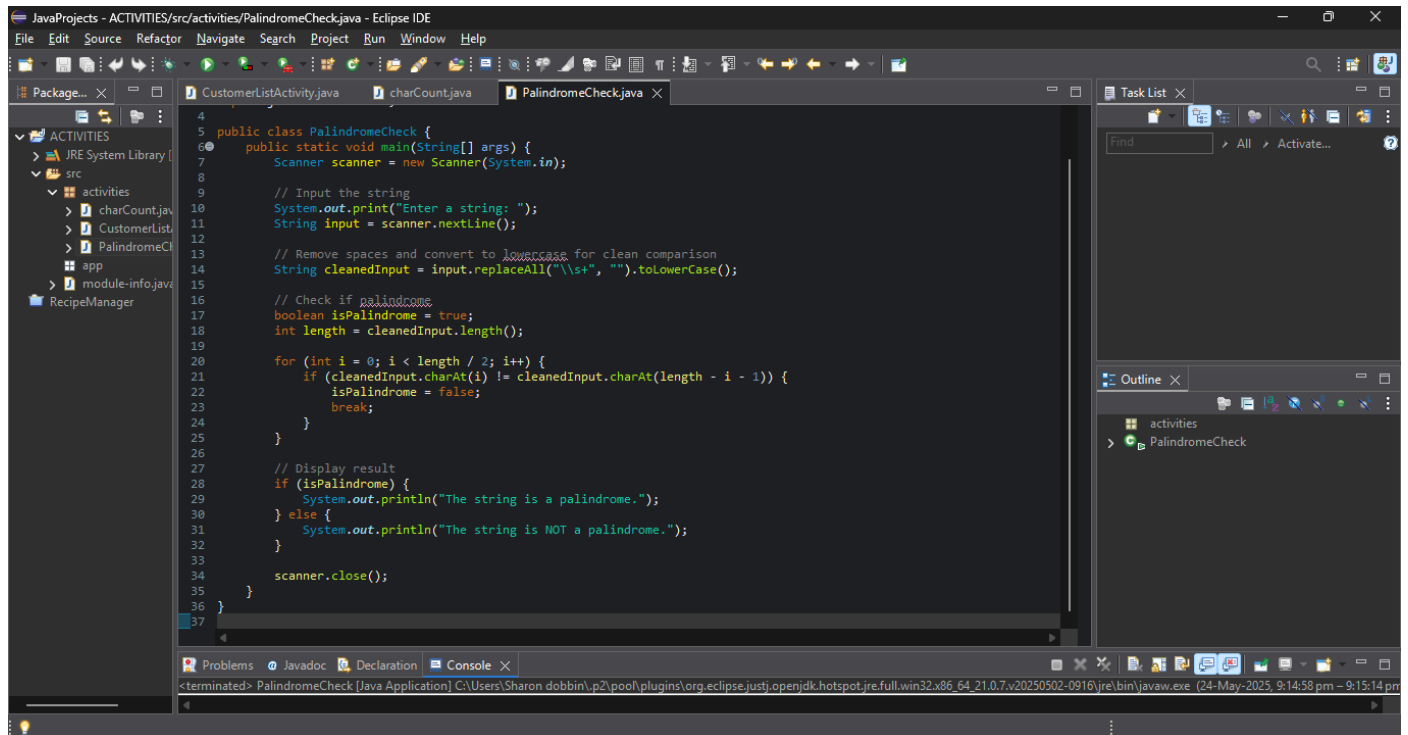


```
<terminated> charCount [Java Application] C:\Users\Sharon.dobbin\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_21.0.7.v20250502-0916\jre\bin\javaw.exe (24-May-2025, 9:10:50 pm - 9:11:13 pm elapsed)
Enter a string: JAVA ACTIVITY 2 - CHARACTER COUNTER
Enter the character to count: A
The character 'A' occurs 5 times in the string.
```

ACTIVITY 3: PALINDROME CHECK

OBJECTIVE: Write a program to find if a string is a palindrome.

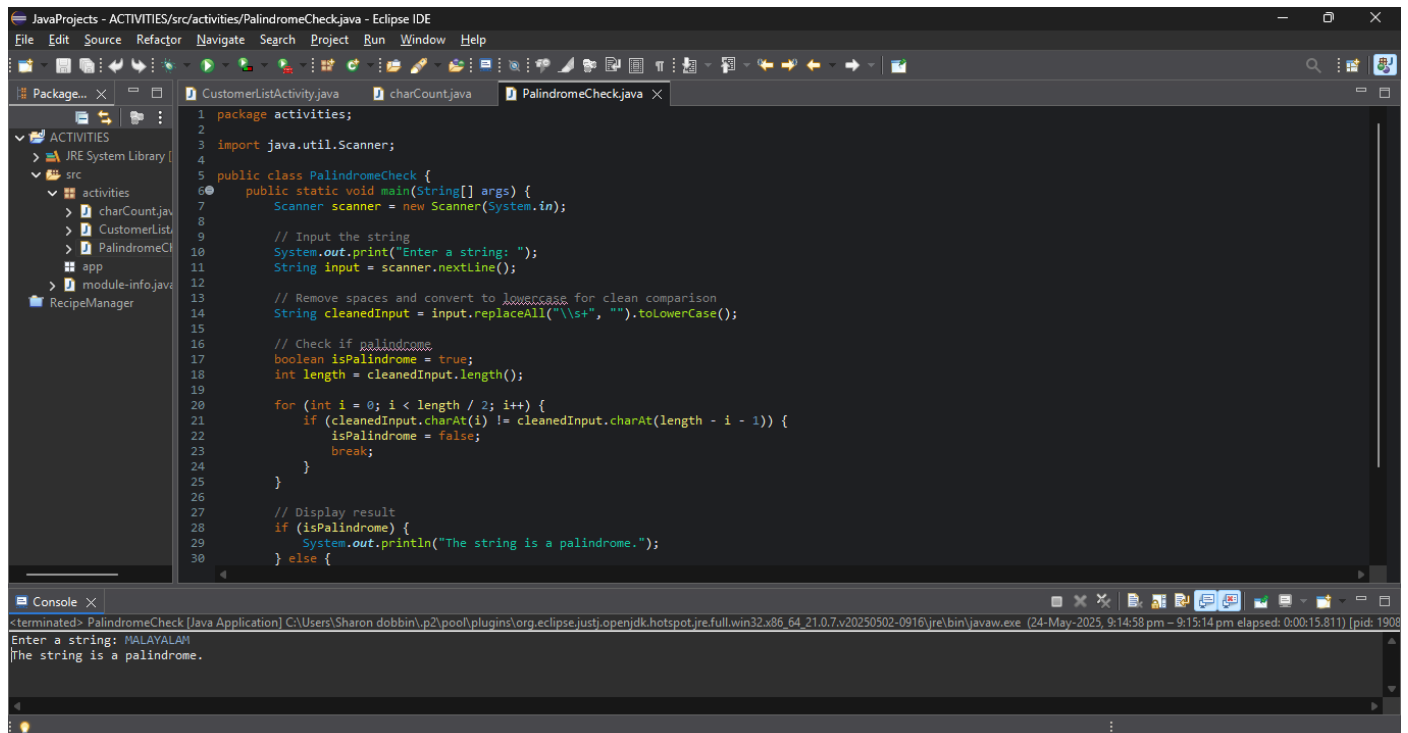
CODE:



The screenshot shows the Eclipse IDE with the file PalindromeCheck.java open. The code is as follows:

```
4 public class PalindromeCheck {
5     public static void main(String[] args) {
6         Scanner scanner = new Scanner(System.in);
7
8         // Input the string
9         System.out.print("Enter a string: ");
10        String input = scanner.nextLine();
11
12        // Remove spaces and convert to lowercase for clean comparison
13        String cleanedInput = input.replaceAll("\\s+", "").toLowerCase();
14
15        // Check if palindrome
16        boolean isPalindrome = true;
17        int length = cleanedInput.length();
18
19        for (int i = 0; i < length / 2; i++) {
20            if (cleanedInput.charAt(i) != cleanedInput.charAt(length - i - 1)) {
21                isPalindrome = false;
22                break;
23            }
24        }
25
26        // Display result
27        if (isPalindrome) {
28            System.out.println("The string is a palindrome.");
29        } else {
30            System.out.println("The string is NOT a palindrome.");
31        }
32        scanner.close();
33    }
34 }
35
36
37
```

OUTPUT:



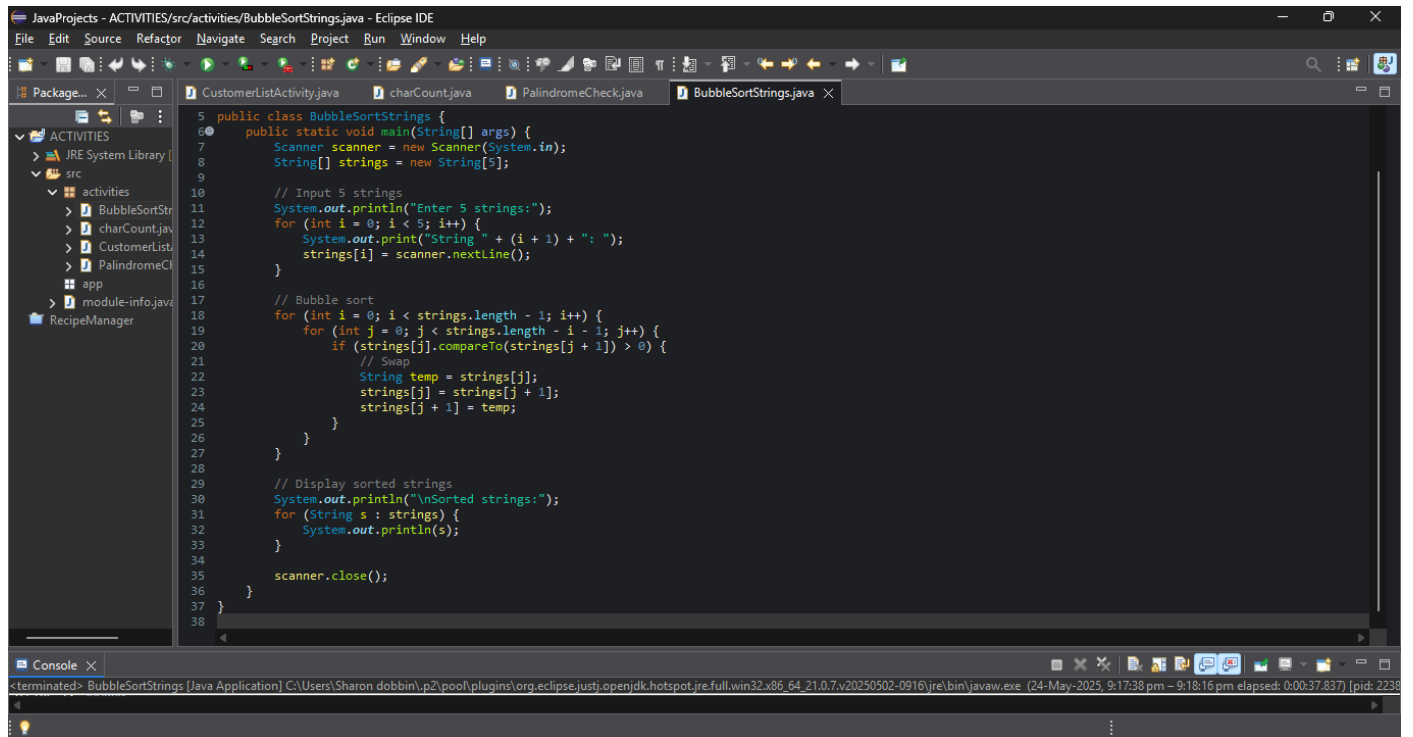
The screenshot shows the Eclipse IDE with the PalindromeCheck.java code open. The console output is as follows:

```
<terminated> PalindromeCheck [Java Application] C:\Users\Sharon.dobbin\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.21.0.7.v20250502-0916\jre\bin\javaw.exe (24-May-2025, 9:14:58 pm - 9:15:14 pm elapsed: 0:00:15.811) [pid: 1908]
Enter a string: MALAYALAM
The string is a palindrome.
```

ACTIVITY 4: BUBBLE SORT

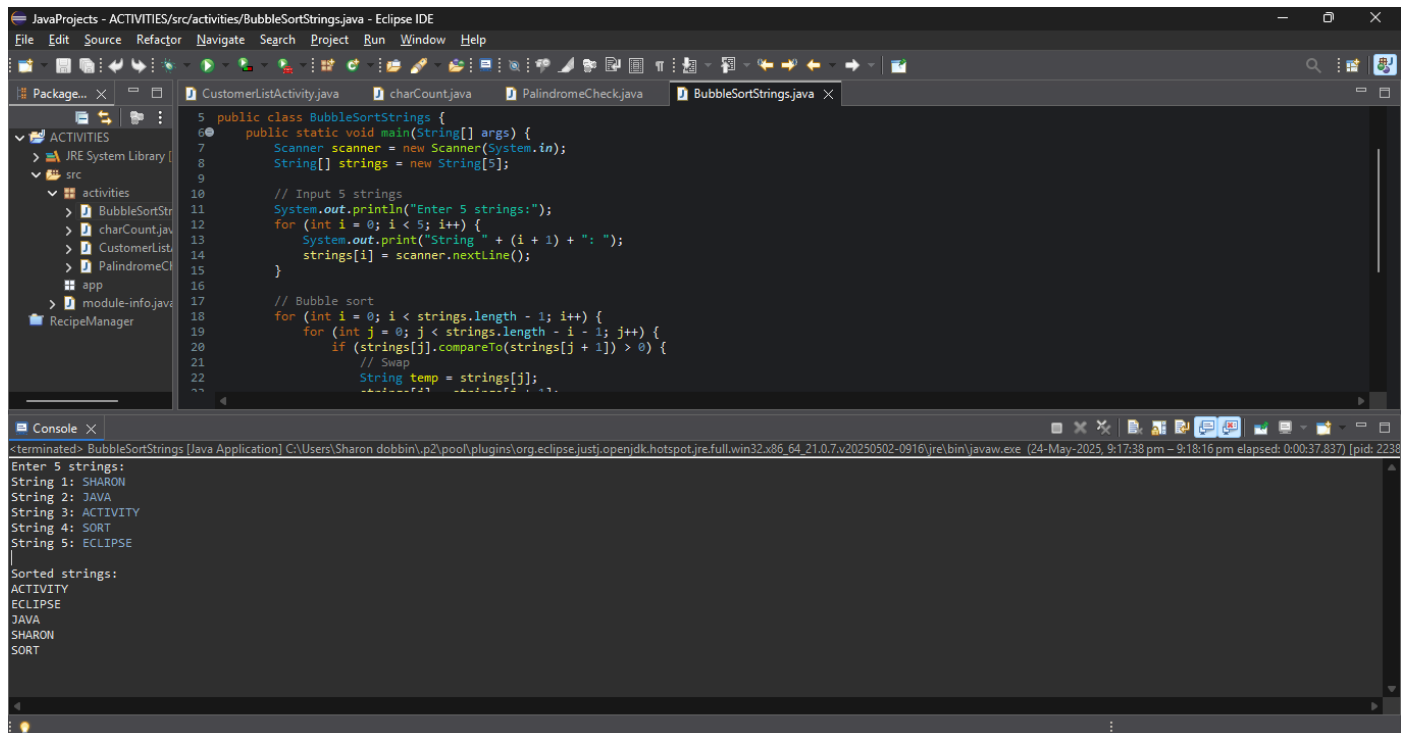
OBJECTIVE: Write a program to sort five strings using bubble sort.

CODE:



```
5 public class BubbleSortStrings {
6     public static void main(String[] args) {
7         Scanner scanner = new Scanner(System.in);
8         String[] strings = new String[5];
9
10        // Input 5 strings
11        System.out.println("Enter 5 strings:");
12        for (int i = 0; i < 5; i++) {
13            System.out.print("String " + (i + 1) + ": ");
14            strings[i] = scanner.nextLine();
15        }
16
17        // Bubble sort
18        for (int i = 0; i < strings.length - 1; i++) {
19            for (int j = 0; j < strings.length - i - 1; j++) {
20                if (strings[j].compareTo(strings[j + 1]) > 0) {
21                    // Swap
22                    String temp = strings[j];
23                    strings[j] = strings[j + 1];
24                    strings[j + 1] = temp;
25                }
26            }
27        }
28
29        // Display sorted strings
30        System.out.println("\nSorted strings:");
31        for (String s : strings) {
32            System.out.println(s);
33        }
34        scanner.close();
35    }
36 }
37
38 }
```

OUTPUT:



```
<terminated> BubbleSortStrings [Java Application] C:\Users\Sharon.dobbin\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.21.0.7.v20250502-0916\jre\bin\javaw.exe (24-May-2025, 9:17:38 pm - 9:18:16 pm elapsed: 0:00:37.837) [pid: 2238]

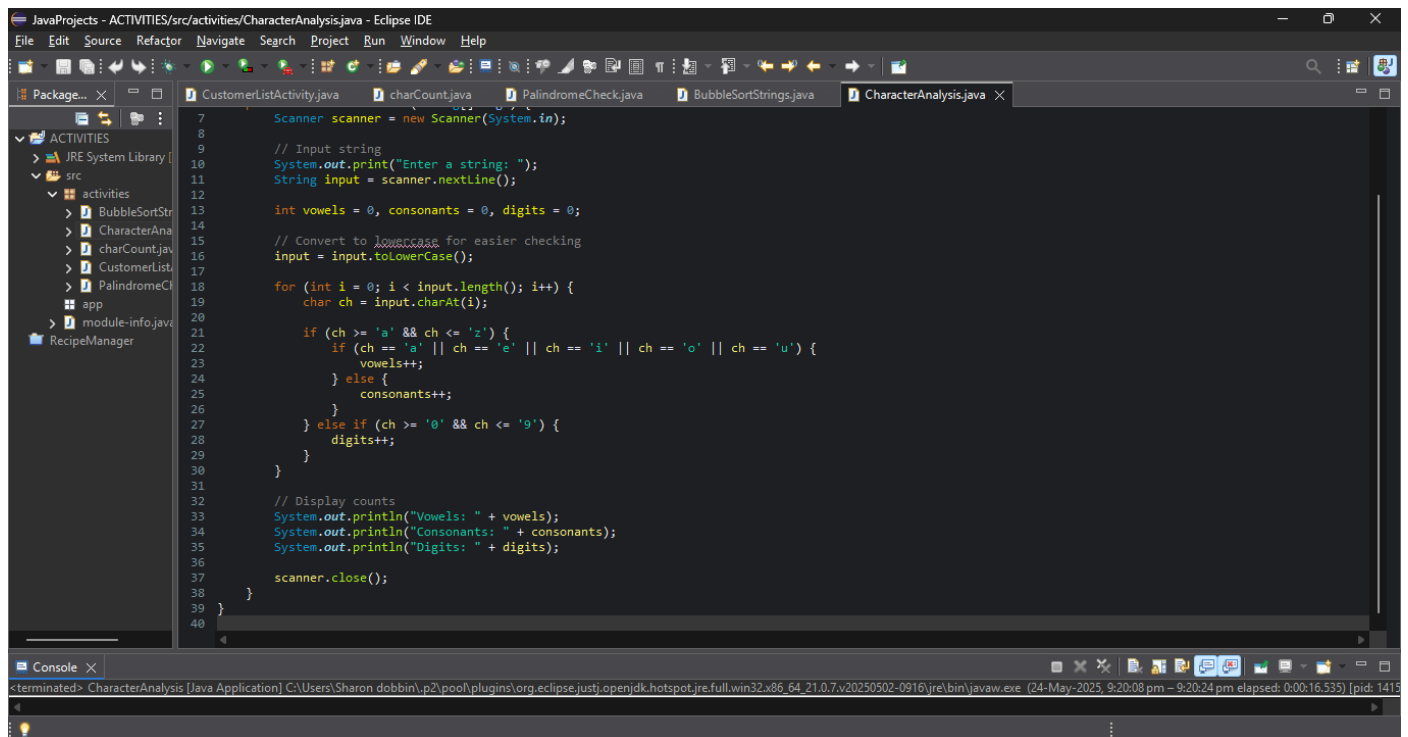
Enter 5 strings:
String 1: SHARON
String 2: JAVA
String 3: ACTIVITY
String 4: SORT
String 5: ECLIPSE

Sorted strings:
ACTIVITY
ECLIPSE
JAVA
SHARON
SORT
```

ACTIVITY 5: CHARACTER ANALYSIS

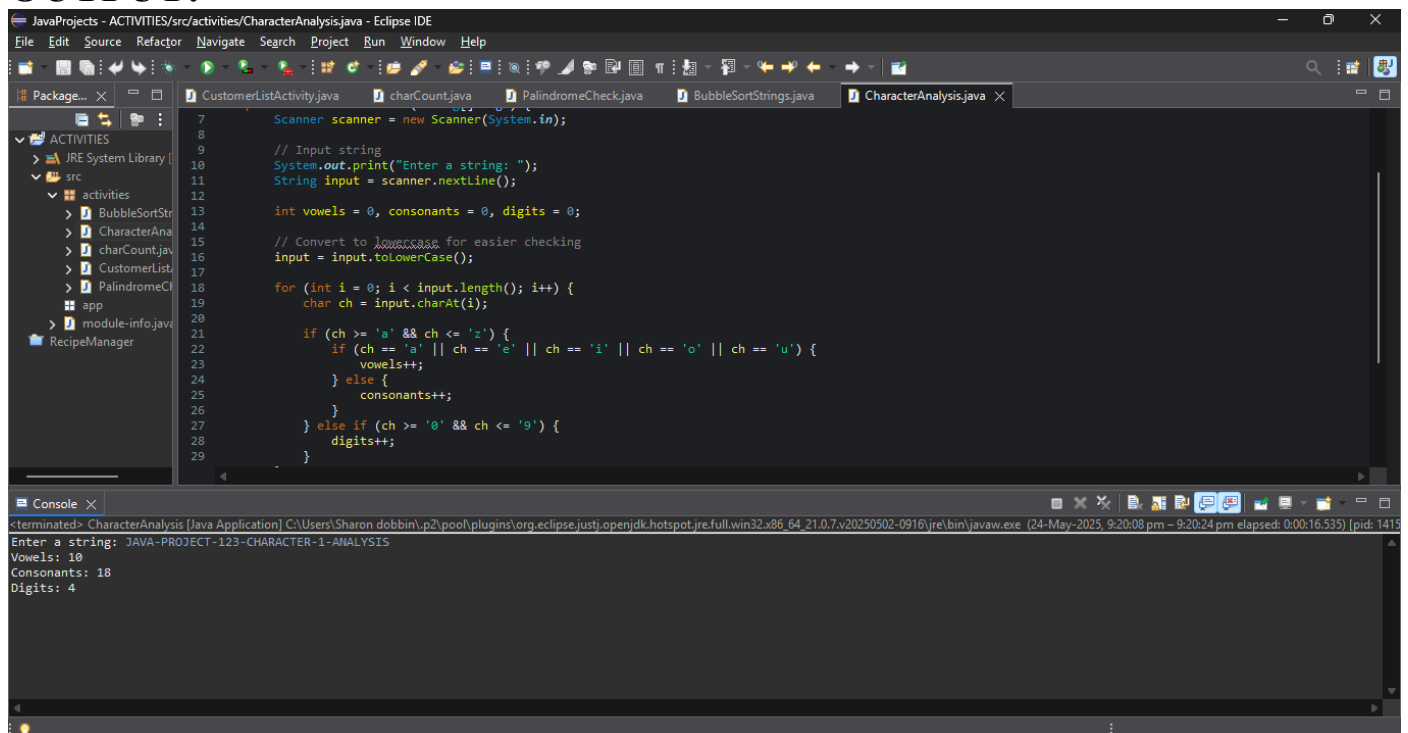
OBJECTIVE: Write a program to find the counts of vowels and consonants and digits in a string.

CODE:



```
7 Scanner scanner = new Scanner(System.in);
8
9 // Input string
10 System.out.print("Enter a string: ");
11 String input = scanner.nextLine();
12
13 int vowels = 0, consonants = 0, digits = 0;
14
15 // Convert to lowercase for easier checking
16 input = input.toLowerCase();
17
18 for (int i = 0; i < input.length(); i++) {
19     char ch = input.charAt(i);
20
21     if (ch >= 'a' && ch <= 'z') {
22         if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
23             vowels++;
24         } else {
25             consonants++;
26         }
27     } else if (ch >= '0' && ch <= '9') {
28         digits++;
29     }
30 }
31
32 // Display counts
33 System.out.println("Vowels: " + vowels);
34 System.out.println("Consonants: " + consonants);
35 System.out.println("Digits: " + digits);
36
37 scanner.close();
38 }
39 }
40 }
```

OUTPUT:

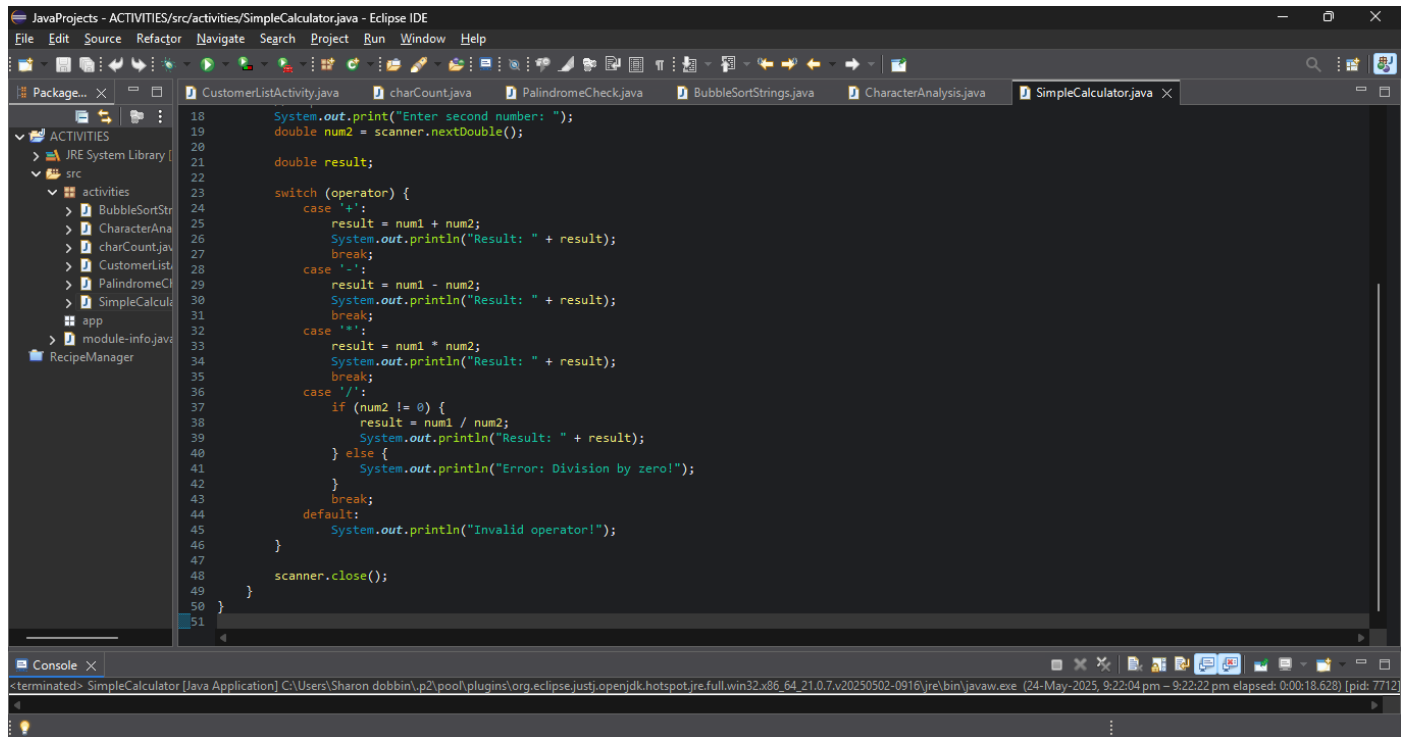


```
<terminated> CharacterAnalysis [Java Application] C:\Users\Sharon.dobbin\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.21.0.7.v20250502-0916\jre\bin\javaw.exe (24-May-2025, 9:20:08 pm - 9:20:24 pm elapsed: 0:00:16.535) [pid: 1415]
Enter a string: JAVA-PROJECT-123-CHARACTER-1-ANALYSIS
Vowels: 10
Consonants: 18
Digits: 4
```

ACTIVITY 6: SIMPLE CALCULATOR

OBJECTIVE: Write a program to create a simple calculator.

CODE:

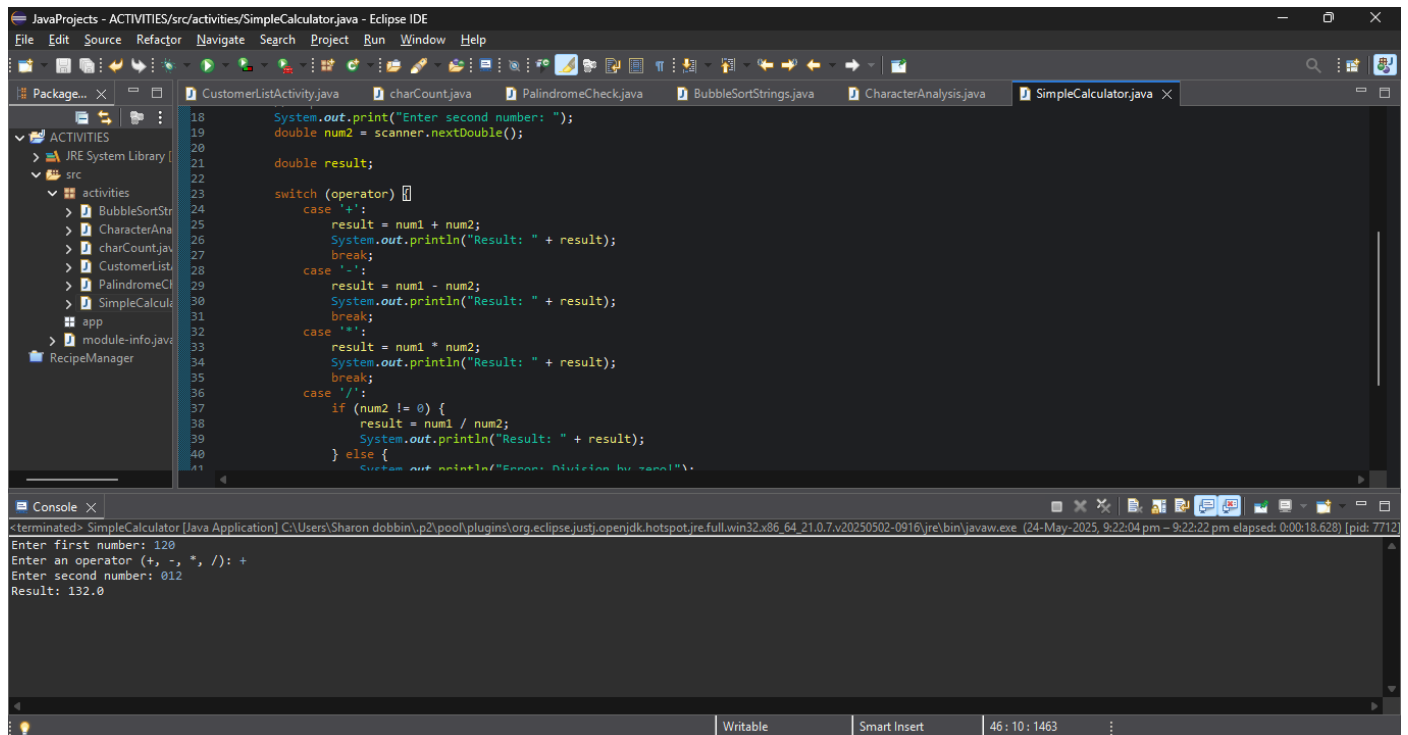


The screenshot shows the Eclipse IDE with the SimpleCalculator.java file open. The code is as follows:

```
18      System.out.print("Enter second number: ");
19      double num2 = scanner.nextDouble();
20
21      double result;
22
23      switch (operator) {
24          case '+':
25              result = num1 + num2;
26              System.out.println("Result: " + result);
27              break;
28          case '-':
29              result = num1 - num2;
30              System.out.println("Result: " + result);
31              break;
32          case '*':
33              result = num1 * num2;
34              System.out.println("Result: " + result);
35              break;
36          case '/':
37              if (num2 != 0) {
38                  result = num1 / num2;
39                  System.out.println("Result: " + result);
40              } else {
41                  System.out.println("Error: Division by zero!");
42              }
43              break;
44          default:
45              System.out.println("Invalid operator!");
46      }
47
48      scanner.close();
49  }
50  }
51  }
```

The console output shows the program execution details, including the file path and execution time.

OUTPUT:



The screenshot shows the Eclipse IDE with the SimpleCalculator.java file open. The code is the same as in the previous screenshot. The console output shows the program execution details, including the file path and execution time.

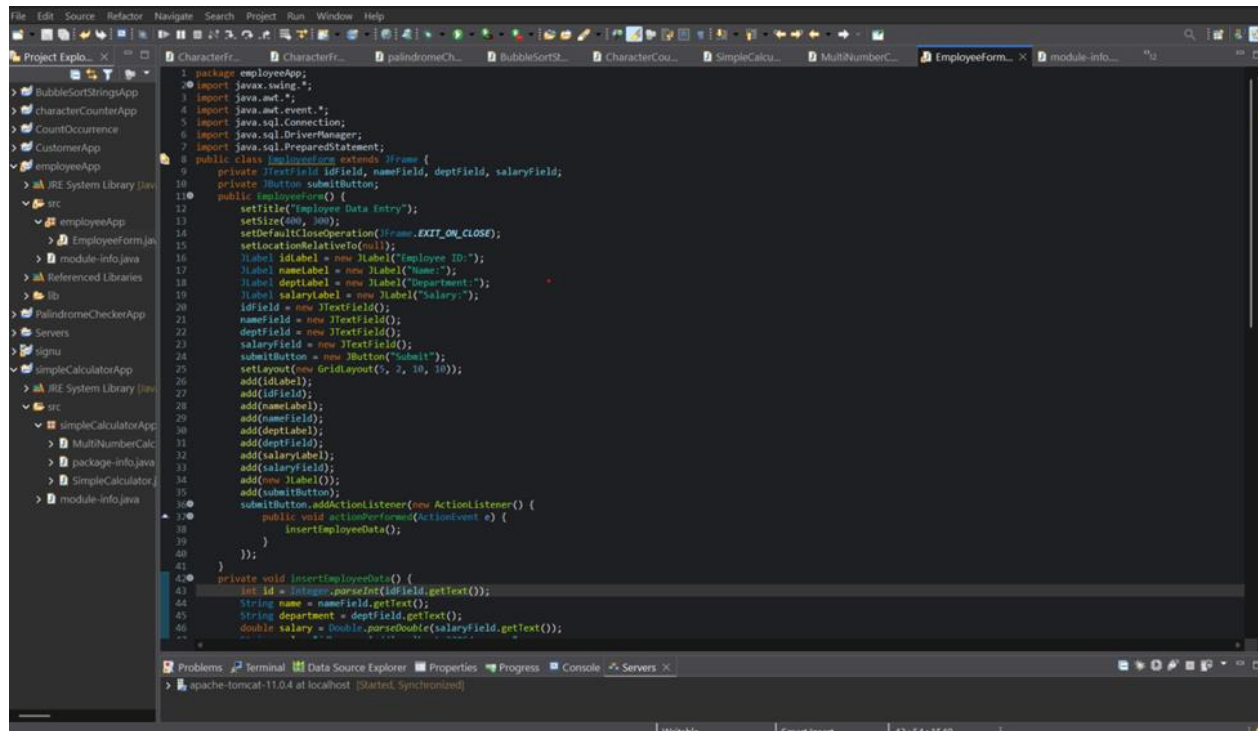
```
<terminated> SimpleCalculator [Java Application] C:\Users\Sharon.dobbin\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.21.0.7.v20250502-0916\jre\bin\javaw.exe (24-May-2025, 9:22:04 pm - 9:22:22 pm elapsed: 0:00:18.628) [pid: 7712]
```

The console output shows the program execution details, including the file path and execution time.

ACTIVITY 7: EMPLOYEE ENTRY

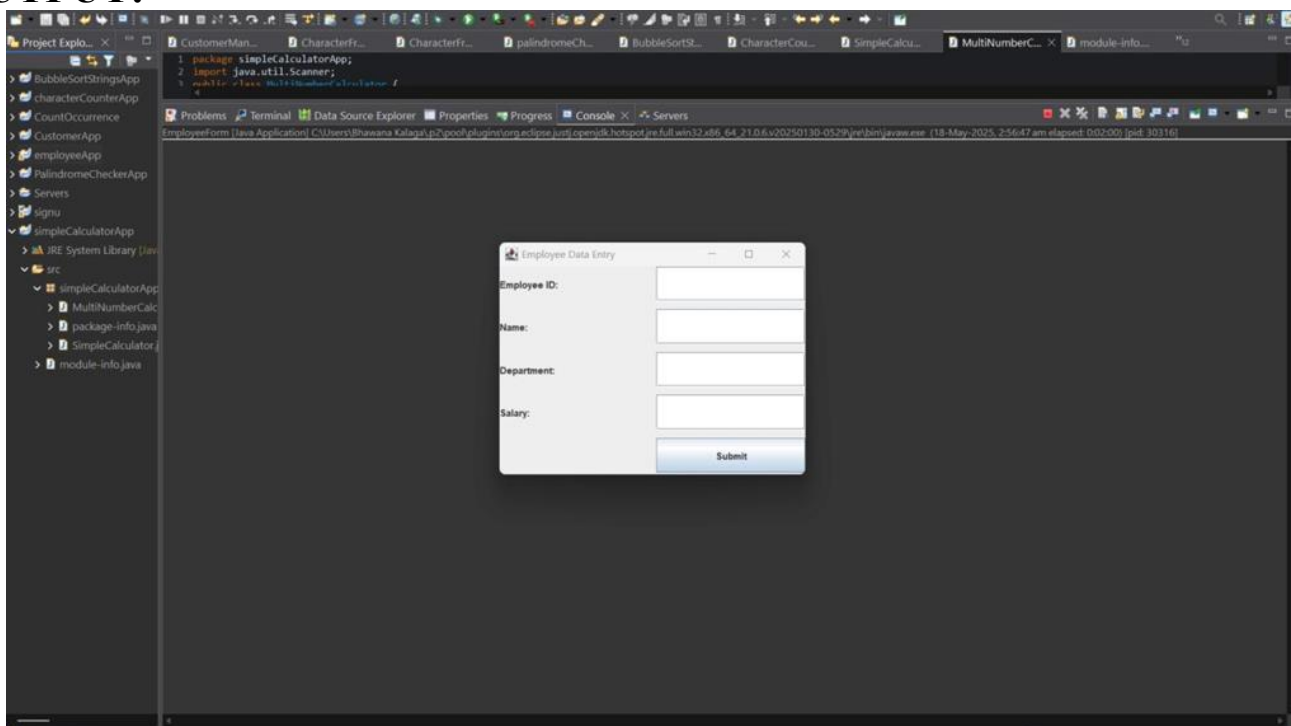
OBJECTIVE: Write a program to create a JFrame to insert the data into the employee table.

CODE:



```
1 package employeeApp;
2 import javax.swing.*;
3 import java.awt.*;
4 import java.awt.event.*;
5 import java.sql.Connection;
6 import java.sql.DriverManager;
7 import java.sql.PreparedStatement;
8 public class EmployeeForm extends JFrame {
9     private JTextField idField, nameField, deptField, salaryField;
10    private JButton submitButton;
11    public EmployeeForm() {
12        setTitle("Employee Data Entry");
13        setSize(400, 300);
14        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
15        setLocationRelativeTo(null);
16        JLabel idLabel = new JLabel("Employee ID:");
17        JLabel nameLabel = new JLabel("Name:");
18        JLabel deptLabel = new JLabel("Department:");
19        JLabel salaryLabel = new JLabel("Salary:");
20        idField = new JTextField();
21        nameField = new JTextField();
22        deptField = new JTextField();
23        salaryField = new JTextField();
24        submitButton = new JButton("Submit");
25        setLayout(new GridLayout(5, 2, 10, 10));
26        add(idLabel);
27        add(idField);
28        add(nameLabel);
29        add(nameField);
30        add(deptLabel);
31        add(deptField);
32        add(salaryLabel);
33        add(salaryField);
34        add(submitButton);
35        submitButton.addActionListener(new ActionListener() {
36            public void actionPerformed(ActionEvent e) {
37                insertEmployeeData();
38            }
39        });
40    }
41    private void insertEmployeeData() {
42        int id = Integer.parseInt(idField.getText());
43        String name = nameField.getText();
44        String department = deptField.getText();
45        double salary = Double.parseDouble(salaryField.getText());
46    }
```

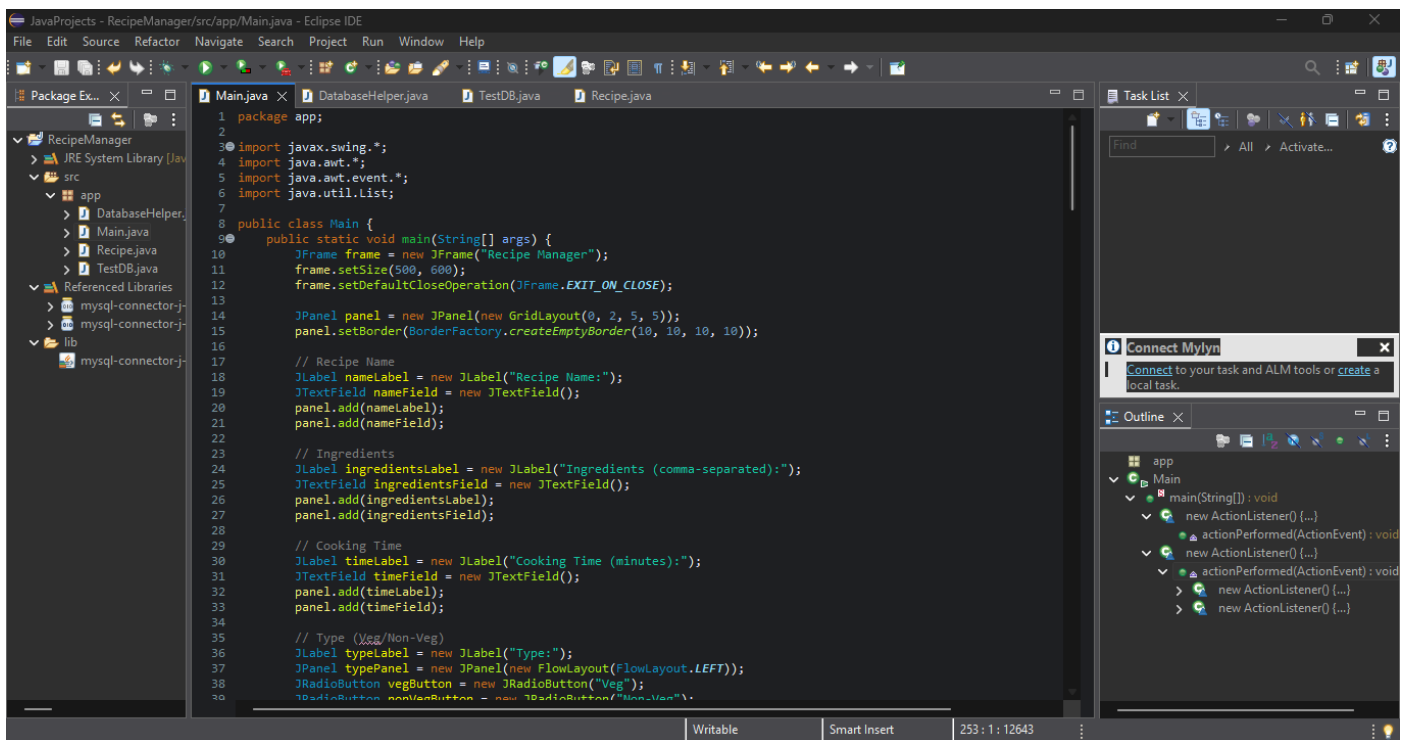
OUTPUT:



ACTIVITY 8: WEB APP

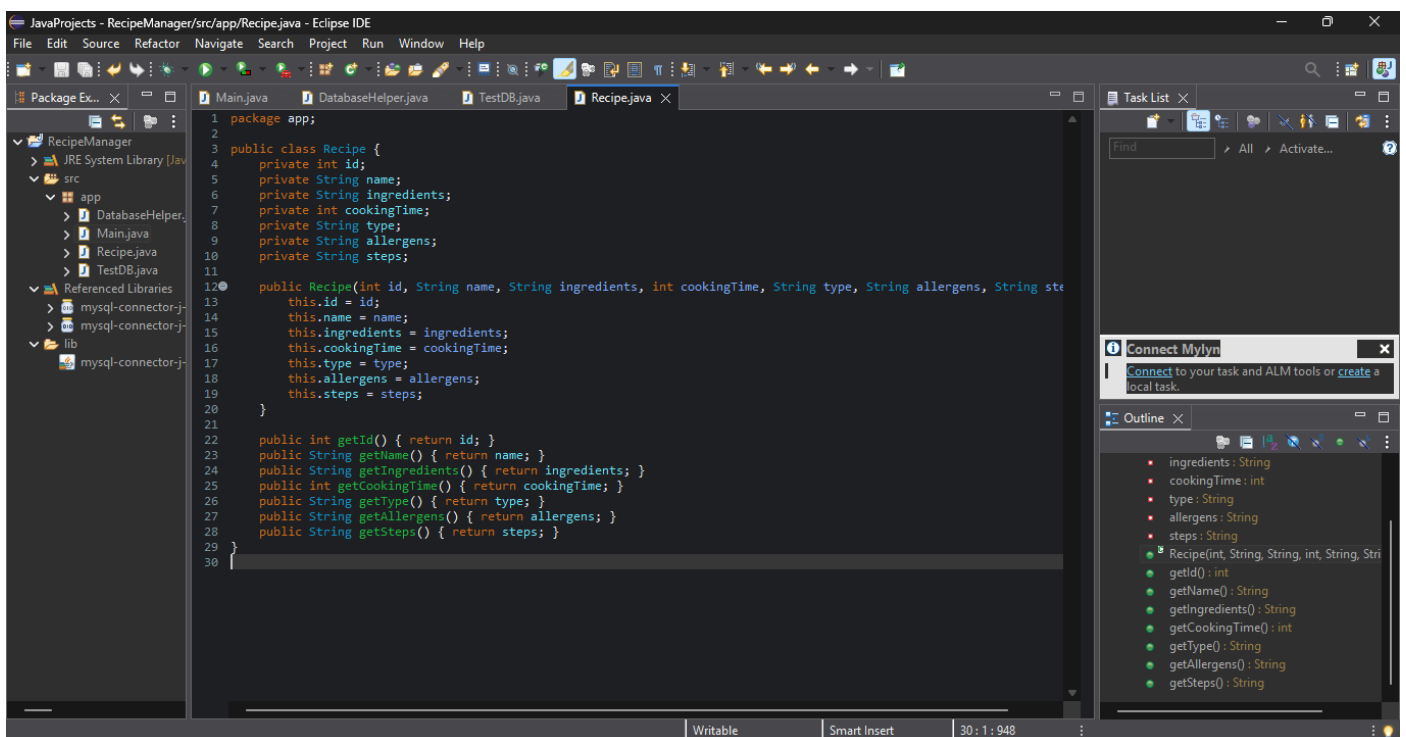
OBJECTIVE: Create a simple application to show the user of Servlets and JSP and try to show the use of a database.

CODE:



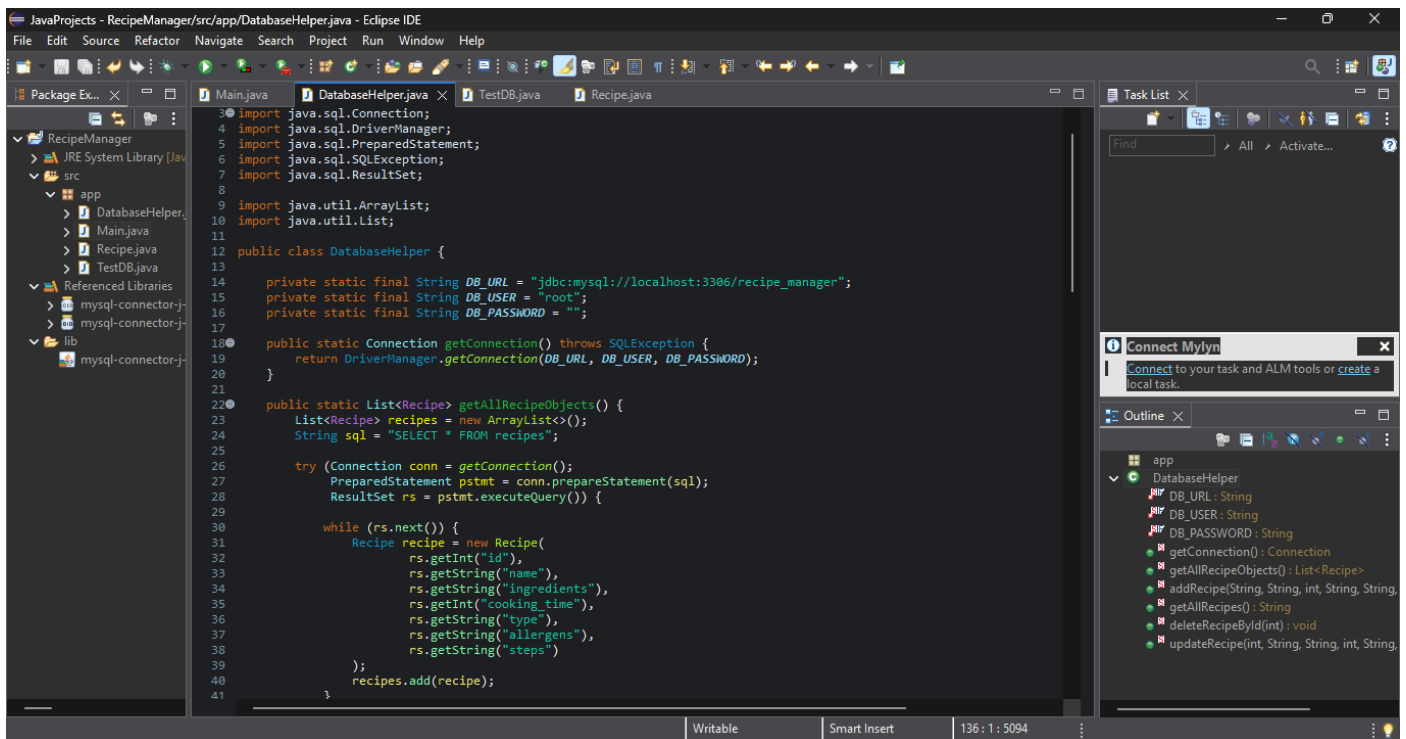
The screenshot shows the Eclipse IDE with the 'Main.java' file open. The code defines a 'Main' class with a 'main' method that creates a 'JFrame' titled 'Recipe Manager'. Inside the frame, a 'JPanel' is created with a 'GridLayout'. The panel contains three sections: 'Recipe Name' with a 'JLabel' and a 'JTextField'; 'Ingredients' with a 'JLabel' and a 'JTextField'; and 'Cooking Time' with a 'JLabel' and a 'JTextField'. There is also a 'Type' section with a 'JLabel' and a 'JPanel' containing two 'JRadioButton' buttons: 'Veg' and 'Non-Veg'.

```
1 package app;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.awt.event.*;
6 import java.util.List;
7
8 public class Main {
9     public static void main(String[] args) {
10         JFrame frame = new JFrame("Recipe Manager");
11         frame.setSize(500, 600);
12         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
13
14         JPanel panel = new JPanel(new GridLayout(0, 2, 5, 5));
15         panel.setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));
16
17         // Recipe Name
18         JLabel nameLabel = new JLabel("Recipe Name:");
19         JTextField nameField = new JTextField();
20         panel.add(nameLabel);
21         panel.add(nameField);
22
23         // Ingredients
24         JLabel ingredientsLabel = new JLabel("Ingredients (comma-separated):");
25         JTextField ingredientsField = new JTextField();
26         panel.add(ingredientsLabel);
27         panel.add(ingredientsField);
28
29         // Cooking Time
30         JLabel timeLabel = new JLabel("Cooking Time (minutes):");
31         JTextField timeField = new JTextField();
32         panel.add(timeLabel);
33         panel.add(timeField);
34
35         // Type (Veg/Non-Veg)
36         JLabel typeLabel = new JLabel("Type:");
37         JPanel typePanel = new JPanel(new FlowLayout(FlowLayout.LEFT));
38         JRadioButton vegButton = new JRadioButton("Veg");
39         JRadioButton nonVegButton = new JRadioButton("Non-Veg");
```

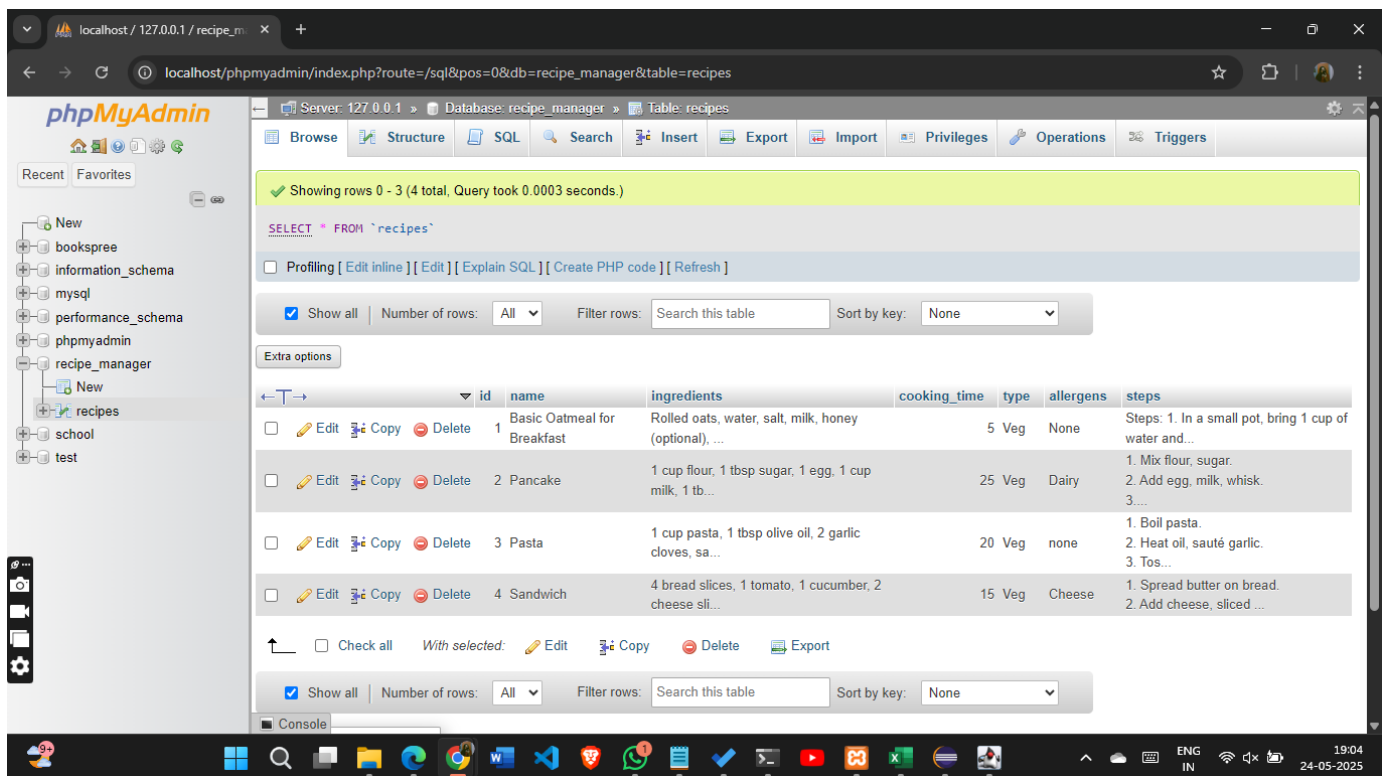


The screenshot shows the Eclipse IDE with the 'Recipe.java' file open. The code defines a 'Recipe' class with private attributes for 'id', 'name', 'ingredients', 'cookingTime', 'type', 'allergens', and 'steps'. It includes a constructor that initializes these attributes and several getter methods to retrieve their values.

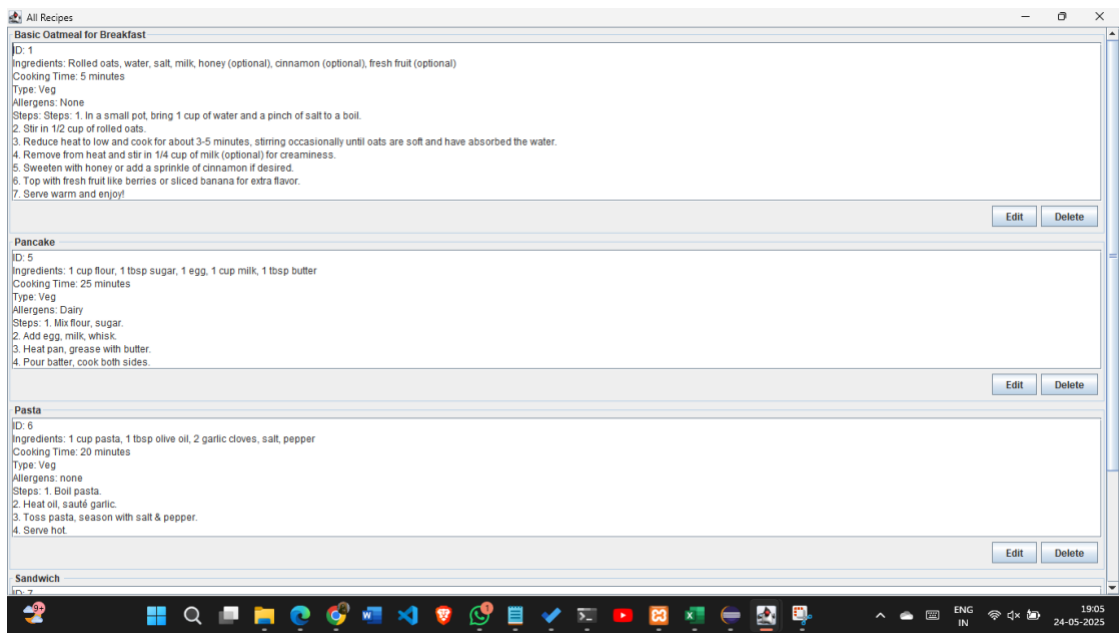
```
1 package app;
2
3 public class Recipe {
4     private int id;
5     private String name;
6     private String ingredients;
7     private int cookingTime;
8     private String type;
9     private String allergens;
10    private String steps;
11
12    public Recipe(int id, String name, String ingredients, int cookingTime, String type, String allergens, String steps) {
13        this.id = id;
14        this.name = name;
15        this.ingredients = ingredients;
16        this.cookingTime = cookingTime;
17        this.type = type;
18        this.allergens = allergens;
19        this.steps = steps;
20    }
21
22    public int getId() { return id; }
23    public String getName() { return name; }
24    public String getIngredients() { return ingredients; }
25    public int getCookingTime() { return cookingTime; }
26    public String getType() { return type; }
27    public String getAllergens() { return allergens; }
28    public String getSteps() { return steps; }
29 }
30
```



DATABASE:



VIEW RECIPE:



ADD RECIPE:

Recipe Manager

Recipe Name:

Aloo Paratha

Ingredients (comma-separated):

2 cups wheat flour, 2 potatoes, spices, butter

Cooking Time (minutes):

30

Type:

☒ Veg

☐ Non-Veg

Allergens (optional):

Butter

Cooking Steps:

1. Knead dough.

2. Mash potatoes, add spices.

3. Stuff dough, roll paratha.

4. Roast with butter.

Add Recipe

View Recipes