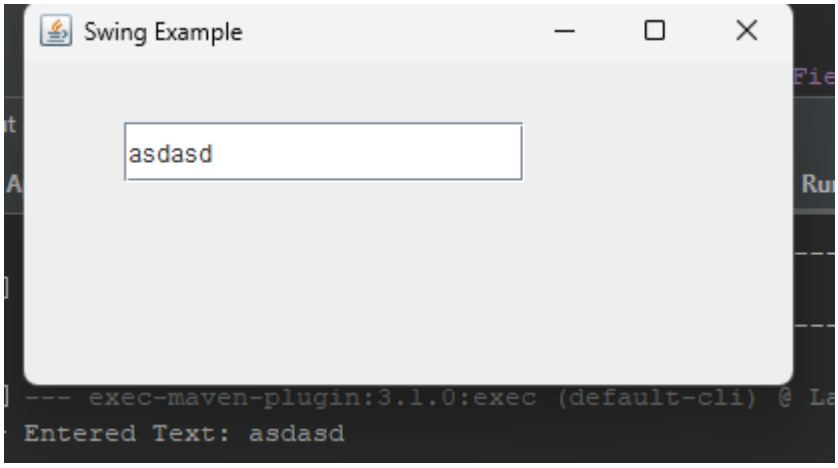
1. WAP to illustrate
a. handling of action event that is generated when enter key is pressed in swing text field

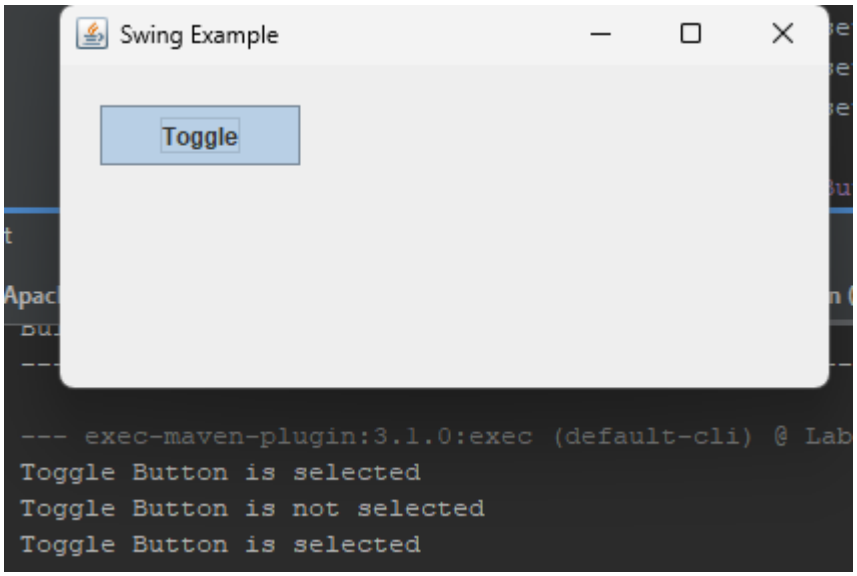
```
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import javax.swing.*;
public class SwingExample {

    JFrame frame;
    JTextField textField;

    SwingExample() {
        frame = new JFrame("Swing Example");
        frame.setSize(400, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(null);

        textField = new JTextField();
        textField.setBounds(50, 30, 200, 30);

        textField.addKeyListener(new KeyListener() {
            @Override
            public void keyTyped(KeyEvent e) {
            }

            @Override
            public void keyPressed(KeyEvent e) {
                if (e.getKeyCode() == KeyEvent.VK_ENTER) {
                    String text = textField.getText();
                    System.out.println("Entered Text: " + text);
                }
            }

            @Override
            public void keyReleased(KeyEvent e) {
            }
        });

        frame.add(textField);
        frame.setVisible(true);
    }

    public static void main(String[] args) {
        new SwingExample();
    }
}
```



b. toggle button in swing

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

public class OneB {

    JFrame frame;
    JToggleButton toggleButton;

    OneB() {
        frame = new JFrame("Swing Example");
        frame.setSize(400, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(null);

        toggleButton = new JToggleButton("Toggle");
        toggleButton.setBounds(20, 20, 100, 30);
        toggleButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                boolean selected = toggleButton.isSelected();
                if (selected) {
                    System.out.println("Toggle Button is selected");
                } else {
                    System.out.println("Toggle Button is not selected");
                }
            }
        });

        frame.add(toggleButton);

        frame.setVisible(true);
    }
```

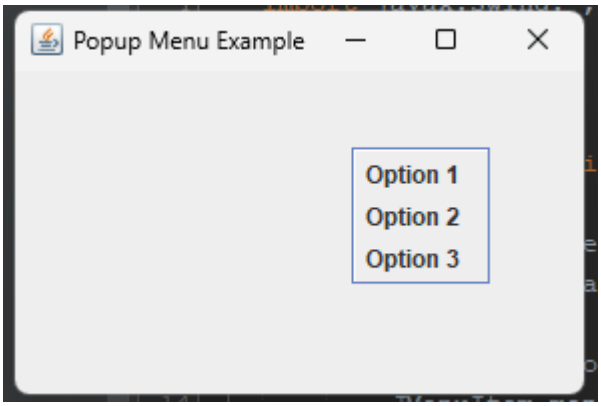
```
    public static void main(String[] args) {
        new OneB();
    }
}
```



2>WAP to illustrate:
a. the concept of popup menu using mouse event handling

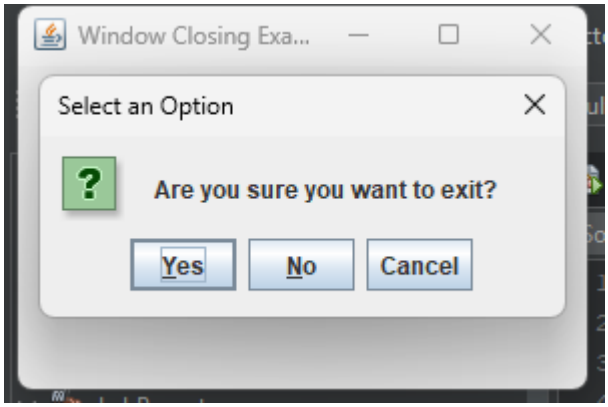
```
import java.awt.event.*;
import javax.swing.*;

public class TwoA {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Popup Menu Example");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPopupMenu popupMenu = new JPopupMenu();
        JMenuItem menuItem1 = new JMenuItem("Option 1");
        JMenuItem menuItem2 = new JMenuItem("Option 2");
        JMenuItem menuItem3 = new JMenuItem("Option 3");

        popupMenu.add(menuItem1);
        popupMenu.add(menuItem2);
        popupMenu.add(menuItem3);

        frame.addMouseListener(new MouseAdapter() {
            @Override
            public void mouseReleased(MouseEvent e) {
                if (e.isPopupTrigger()) {
                    popupMenu.show(e.getComponent(), e.getX(), e.getY());
                }
```

```
            }
        });

        frame.setVisible(true);
    }
}
```



b. to handle windows closing event.

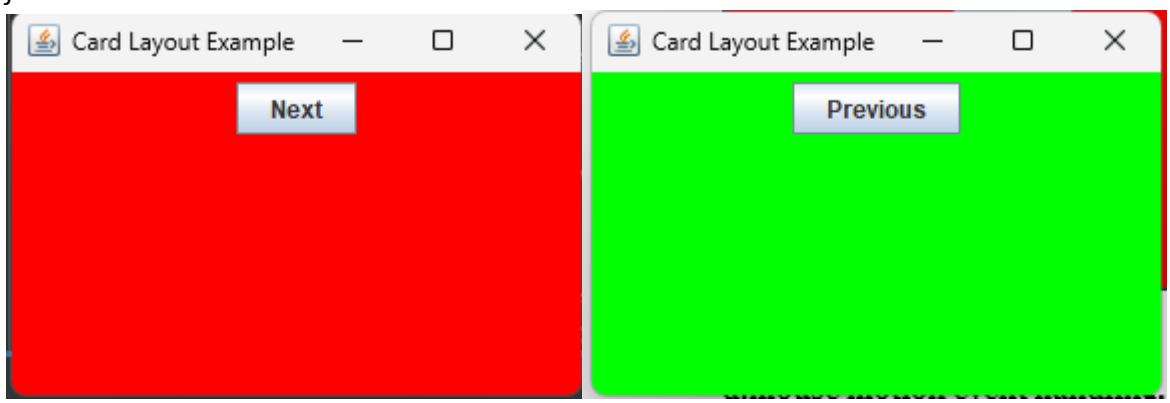
```
import java.awt.event.*;
import javax.swing.*;

public class TwoB {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Window Closing Example");
        frame.setSize(300, 200);

        frame.addWindowListener(new WindowAdapter() {
            @Override
            public void windowClosing(WindowEvent e) {
                int option = JOptionPane.showConfirmDialog(frame, "Are you sure you want to
exit?");
                if (option == JOptionPane.YES_OPTION) {
                    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
                } else {
                    frame.setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
                }
            }
        });

        frame.setVisible(true);
    }
}
```



c. to demonstrate Card Layout using event handling

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

public class TwoC {

    private static JPanel cardPanel;
    private static CardLayout cardLayout;

    public static void main(String[] args) {
        JFrame frame = new JFrame("Card Layout Example");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        cardPanel = new JPanel();
        cardLayout = new CardLayout();
        cardPanel.setLayout(cardLayout);

        JPanel card1 = new JPanel();
        card1.setBackground(Color.RED);
        JButton nextButton = new JButton("Next");
        nextButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                cardLayout.next(cardPanel);
            }
        });
        card1.add(nextButton);

        JPanel card2 = new JPanel();
        card2.setBackground(Color.GREEN);
        JButton prevButton = new JButton("Previous");
        prevButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                cardLayout.previous(cardPanel);
```

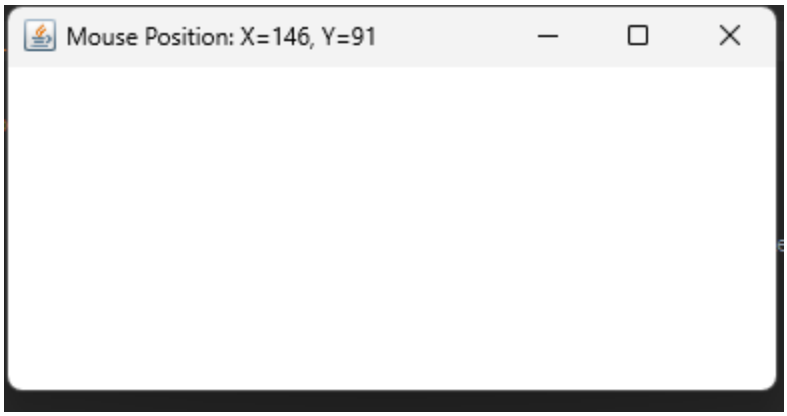
```
        }
    });
    card2.add(prevButton);

    cardPanel.add(card1, "Card 1");
    cardPanel.add(card2, "Card 2");

    frame.getContentPane().add(cardPanel);
    frame.setVisible(true);
  }
}
```



d.mouse motion event handling.

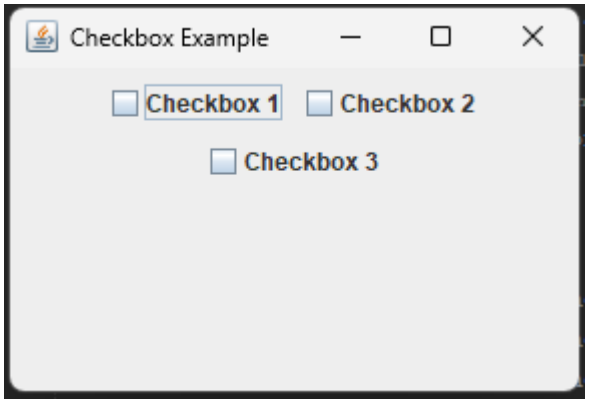
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class TwoD {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Mouse Motion Example");
        frame.setSize(400, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel = new JPanel();
        panel.setBackground(Color.WHITE);

        panel.addMouseMotionListener(new MouseMotionAdapter() {
            @Override
            public void mouseMoved(MouseEvent e) {
                int x = e.getX();
                int y = e.getY();
                frame.setTitle("Mouse Position: X=" + x + ", Y=" + y);
            }
        });
```

```
        frame.getContentPane().add(panel);
        frame.setVisible(true);
    }

}
```



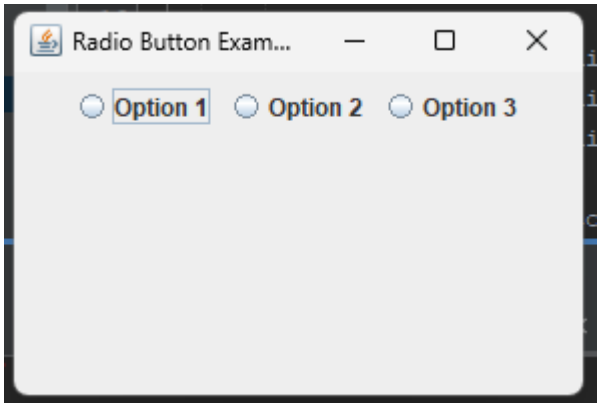
3. Write GUI program using Swing
a. to show the check boxes

```
import javax.swing.*;

public class ThreeA {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Checkbox Example");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel = new JPanel();

        JCheckBox checkBox1 = new JCheckBox("Checkbox 1");
        JCheckBox checkBox2 = new JCheckBox("Checkbox 2");
        JCheckBox checkBox3 = new JCheckBox("Checkbox 3");

        panel.add(checkBox1);
        panel.add(checkBox2);
        panel.add(checkBox3);

        frame.getContentPane().add(panel);
        frame.setVisible(true);
    }
}
```



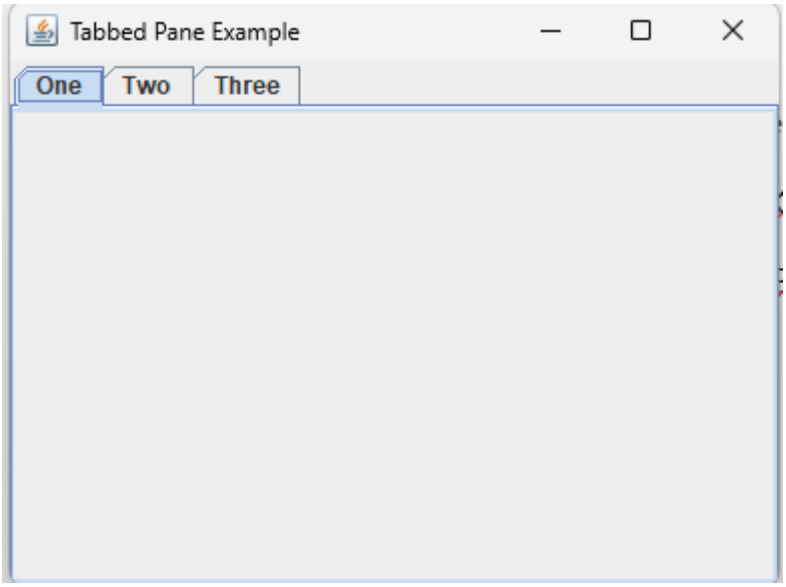
b. to show radio buttons

```
import javax.swing.*;
public class ThreeB {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Radio Button Example");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JPanel panel = new JPanel();

        JRadioButton radioButton1 = new JRadioButton("Option 1");
        JRadioButton radioButton2 = new JRadioButton("Option 2");
        JRadioButton radioButton3 = new JRadioButton("Option 3");

        ButtonGroup buttonGroup = new ButtonGroup();
        buttonGroup.add(radioButton1);
        buttonGroup.add(radioButton2);
        buttonGroup.add(radioButton3);

        panel.add(radioButton1);
        panel.add(radioButton2);
        panel.add(radioButton3);

        frame.getContentPane().add(panel);
        frame.setVisible(true);
    }
}
```



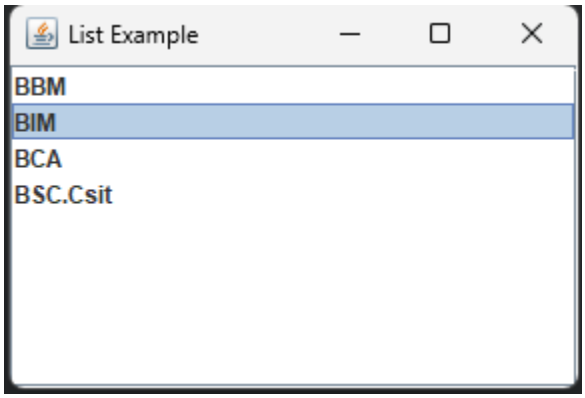
c. containing 3 tabbed panes named “one”, “two” and “three”

```
import javax.swing.*;

public class ThreeC {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Tabbed Pane Example");
        frame.setSize(400, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JTabbedPane tabbedPane = new JTabbedPane();
        JPanel panel1 = new JPanel();
        JPanel panel2 = new JPanel();
        JPanel panel3 = new JPanel();


        tabbedPane.addTab("One", panel1);
        tabbedPane.addTab("Two", panel2);
        tabbedPane.addTab("Three", panel3);

        frame.getContentPane().add(tabbedPane);
        frame.setVisible(true);
    }
}
```



d. containing a list having the values as: BBM, BIM, BCA, BSC. Csit

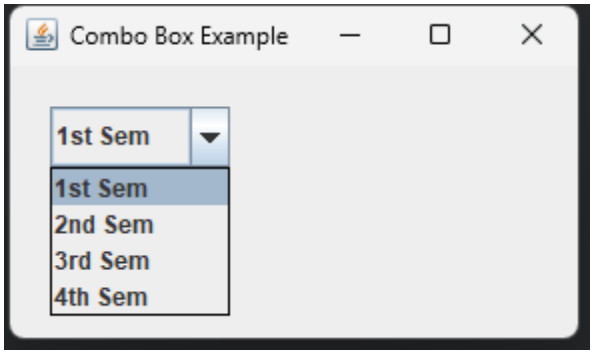
```
import javax.swing.*;

public class ThreeD {

    public static void main(String[] args) {
        JFrame frame = new JFrame("List Example");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        String[] values = {"BBM", "BIM", "BCA", "BSC.Csit"};
        JList<String> list = new JList<>(values);

        frame.getContentPane().add(new JScrollPane(list));
        frame.setVisible(true);
    }
}
```



e. containing combo box having the values: 1st Sem, 2nd Sem, 3rd Sem, 4th Sem

```
package sumir.labreport;

import javax.swing.*;
```

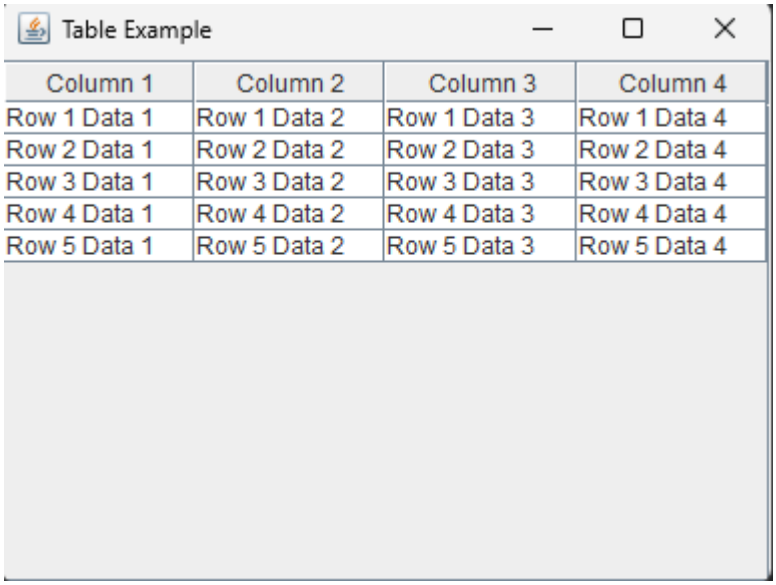
```
public class ThreeE {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Combo Box Example");
        frame.setSize(300, 200);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(null);

        String[] values = {"1st Sem", "2nd Sem", "3rd Sem", "4th Sem"};
        JComboBox<String> comboBox = new JComboBox<>(values);

        comboBox.setBounds(20, 20, 90, 30);
        frame.add(comboBox);
        frame.setVisible(true);
    }
}
```



f. containing table showing a table with 5 rows and 4 columns of data

```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;

public class ThreeF {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Table Example");
        frame.setSize(400, 300);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        String[] columnNames = {"Column 1", "Column 2", "Column 3", "Column 4"};
        Object[][] rowData = {
            {"Row 1 Data 1", "Row 1 Data 2", "Row 1 Data 3", "Row 1 Data 4"},
            {"Row 2 Data 1", "Row 2 Data 2", "Row 2 Data 3", "Row 2 Data 4"},
            {"Row 3 Data 1", "Row 3 Data 2", "Row 3 Data 3", "Row 3 Data 4"},
            {"Row 4 Data 1", "Row 4 Data 2", "Row 4 Data 3", "Row 4 Data 4"},
            {"Row 5 Data 1", "Row 5 Data 2", "Row 5 Data 3", "Row 5 Data 4"}
        };

        DefaultTableModel model = new DefaultTableModel(rowData, columnNames);
        JTable table = new JTable(model);
```

```
        JScrollPane scrollPane = new JScrollPane(table);

        frame.getContentPane().add(scrollPane);
        frame.setVisible(true);
    }
}
```



4. WAP using PreparedStatement to display the records from a table of given database (Suppose "myrecords" database is in MySql server at ip: 198.54.67.78 on port 3306). Assume the following table: salary(emp_id , emp_name ,emp_salary) The program should read the employee id value from console and display the corresponding record.

```
import java.sql.*;
import java.util.Scanner;

public class Four {

    public static void main(String[] args) throws ClassNotFoundException {
        String url = "jdbc:mysql://198.54.67.78:3306/myrecords";
        String username = "root";
        String password = "root";

        try {
            Class.forName("com.mysql.jdbc.Driver");
            Connection connection = DriverManager.getConnection(url, username, password);

            String sql = "SELECT * FROM salary WHERE emp_id = ?";
            PreparedStatement statement = connection.prepareStatement(sql);

            Scanner scanner = new Scanner(System.in);
            System.out.print("Enter employee ID: ");
            int empId = scanner.nextInt();
```

```
            statement.setInt(1, empId);

            ResultSet resultSet = statement.executeQuery();

            if (resultSet.next()) {
                int id = resultSet.getInt("emp_id");
                String name = resultSet.getString("emp_name");
                double salary = resultSet.getDouble("emp_salary");

                System.out.println("Employee ID: " + id);
                System.out.println("Employee Name: " + name);
                System.out.println("Employee Salary: " + salary);
            } else {
                System.out.println("No record found for the given employee ID.");
            }

            resultSet.close();
            statement.close();
            connection.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

```
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver
Enter employee ID: 1
Employee ID: 1
Employee Name: Hari
Employee Salary: 200000.0
--------------------------------------------------------------------------
```

5.Develop a CRUD application using swing and JDBC. Your UI must contain (name) text field, (gender) radio button and (faculty) combo box . Assume MySQL server at 192.168.89.99 on port 9876 having database name "records" and table named "students".

Student .java

```
package five;

public class Student {

    private int id;
    private String name;
    private String gender;
    private String faculty;
```

```
    public Student(int id, String name, String gender, String faculty) {
        this.id = id;
        this.name = name;
        this.gender = gender;
        this.faculty = faculty;
    }

    public Student(String name, String gender, String faculty) {
        this.name = name;
        this.gender = gender;
        this.faculty = faculty;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getGender() {
        return gender;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }

    public String getFaculty() {
        return faculty;
    }

    public void setFaculty(String faculty) {
        this.faculty = faculty;
    }
}
```

StudentDAO.java

```
package five;
```

```
import java.sql.*;

public class StudentDAO {

    private Connection connection;
    private PreparedStatement insertStatement;
    private PreparedStatement updateStatement;
    private PreparedStatement deleteStatement;

    public StudentDAO() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            connection =
DriverManager.getConnection("jdbc:mysql://192.168.89.99:9876/records", "root", "root");

            insertStatement = connection.prepareStatement("INSERT INTO students (name,
gender, faculty) VALUES (?, ?, ?)");
            updateStatement = connection.prepareStatement("UPDATE students SET name = ?,
gender = ?, faculty = ? WHERE id = ?");
            deleteStatement = connection.prepareStatement("DELETE FROM students WHERE
id = ?");
        } catch (ClassNotFoundException | SQLException e) {
            e.printStackTrace();
        }
    }

    public void addStudent(Student student) {
        try {
            insertStatement.setString(1, student.getName());
            insertStatement.setString(2, student.getGender());
            insertStatement.setString(3, student.getFaculty());
            insertStatement.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public void updateStudent(Student student) {
        try {
            updateStatement.setString(1, student.getName());
            updateStatement.setString(2, student.getGender());
            updateStatement.setString(3, student.getFaculty());
            updateStatement.setInt(4, student.getId());
            updateStatement.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
```

```
    }

    public void deleteStudent(int studentId) {
        try {
            deleteStatement.setInt(1, studentId);
            deleteStatement.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public Student getStudentByName(String name) {
        try {
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery("SELECT * FROM students WHERE
name = '" + name + "'");
            if (resultSet.next()) {
                int id = resultSet.getInt("id");
                String gender = resultSet.getString("gender");
                String faculty = resultSet.getString("faculty");
                return new Student(id, name, gender, faculty);
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

StudentCRUDAppUI.java

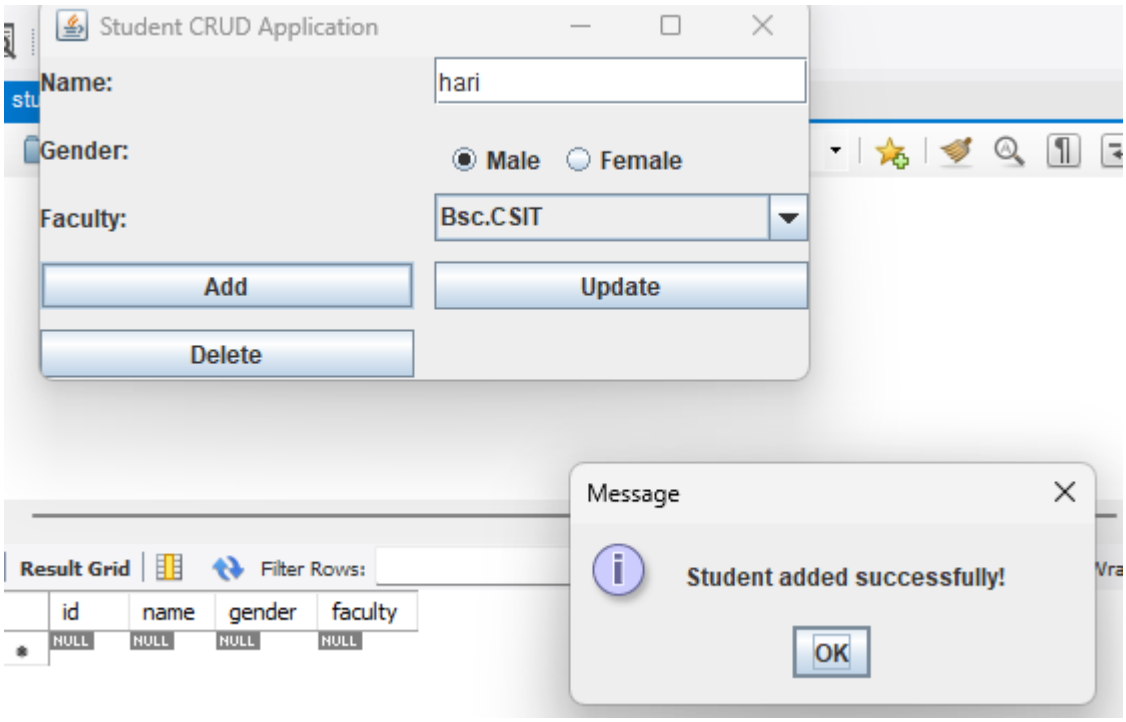
```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class StudentCRUDAppUI {

    private JTextField nameField;
    private JRadioButton maleRadio;
    private JRadioButton femaleRadio;
    private JComboBox<String> facultyCombo;

    private StudentDAO studentDAO;

    public StudentCRUDAppUI() {
```

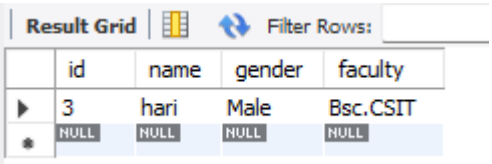
```
    studentDAO = new StudentDAO();
    initializeUI();
}

private void initializeUI() {
    JFrame frame = new JFrame("Student CRUD Application");
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400, 200);

    JPanel mainPanel = new JPanel(new GridLayout(5, 2, 10, 10));
    frame.add(mainPanel);

    JLabel nameLabel = new JLabel("Name:");
    mainPanel.add(nameLabel);

    nameField = new JTextField();
    mainPanel.add(nameField);

    JLabel genderLabel = new JLabel("Gender:");
    mainPanel.add(genderLabel);

    ButtonGroup genderGroup = new ButtonGroup();
    maleRadio = new JRadioButton("Male");
    femaleRadio = new JRadioButton("Female");
    genderGroup.add(maleRadio);
    genderGroup.add(femaleRadio);
    JPanel genderPanel = new JPanel(new FlowLayout(FlowLayout.LEFT));
    genderPanel.add(maleRadio);
    genderPanel.add(femaleRadio);
    mainPanel.add(genderPanel);

    JLabel facultyLabel = new JLabel("Faculty:");
    mainPanel.add(facultyLabel);

    String[] facultyOptions = {"BCA", "Bsc.CSIT", "BBS"};
    facultyCombo = new JComboBox<>(facultyOptions);
    mainPanel.add(facultyCombo);

    JButton addButton = new JButton("Add");
    addButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            addStudent();
        }
    });
    mainPanel.add(addButton);

    JButton updateButton = new JButton("Update");
```

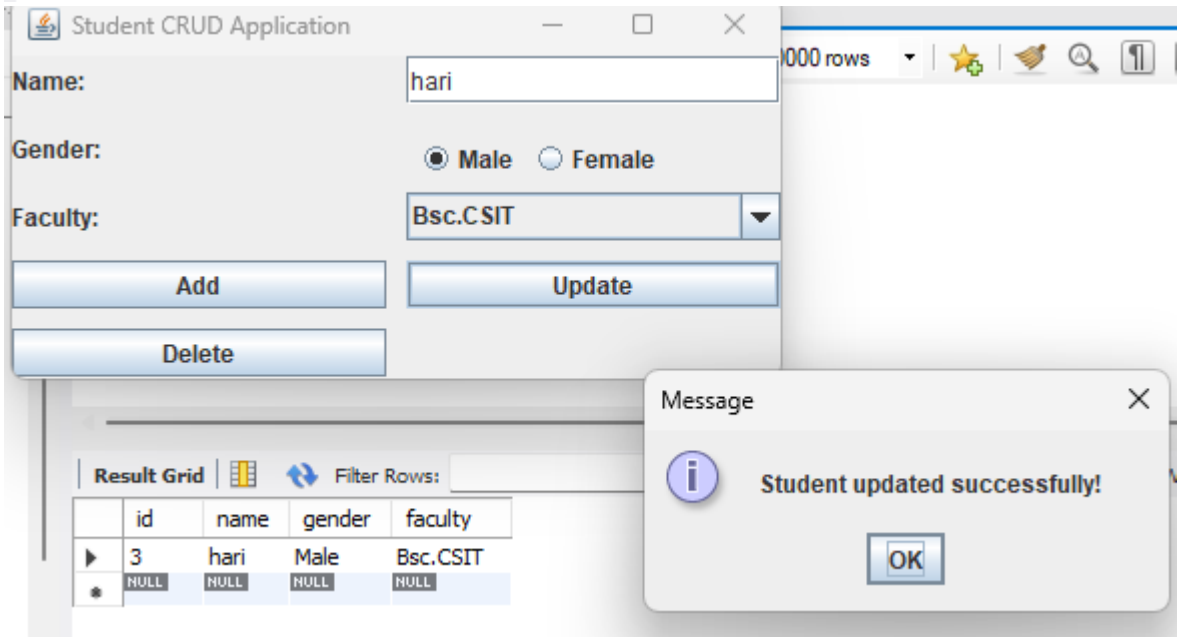
```
    updateButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            updateStudent();
        }
    });
    mainPanel.add(updateButton);

    JButton deleteButton = new JButton("Delete");
    deleteButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            deleteStudent();
        }
    });
    mainPanel.add(deleteButton);

    frame.setVisible(true);
}

private void addStudent() {
    String name = nameField.getText();
    String gender = maleRadio.isSelected() ? "Male" : "Female";
    String faculty = (String) facultyCombo.getSelectedItem();

    if (name.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Please enter a name.");
        return;
    }

    Student student = new Student(name, gender, faculty);
    studentDAO.addStudent(student);
    JOptionPane.showMessageDialog(null, "Student added successfully!");
    clearForm();
}

private void updateStudent() {
    String name = nameField.getText();
    String gender = maleRadio.isSelected() ? "Male" : "Female";
    String faculty = (String) facultyCombo.getSelectedItem();

    if (name.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Please enter a name.");
        return;
    }

    Student student = studentDAO.getStudentByName(name);
    if (student != null) {
```

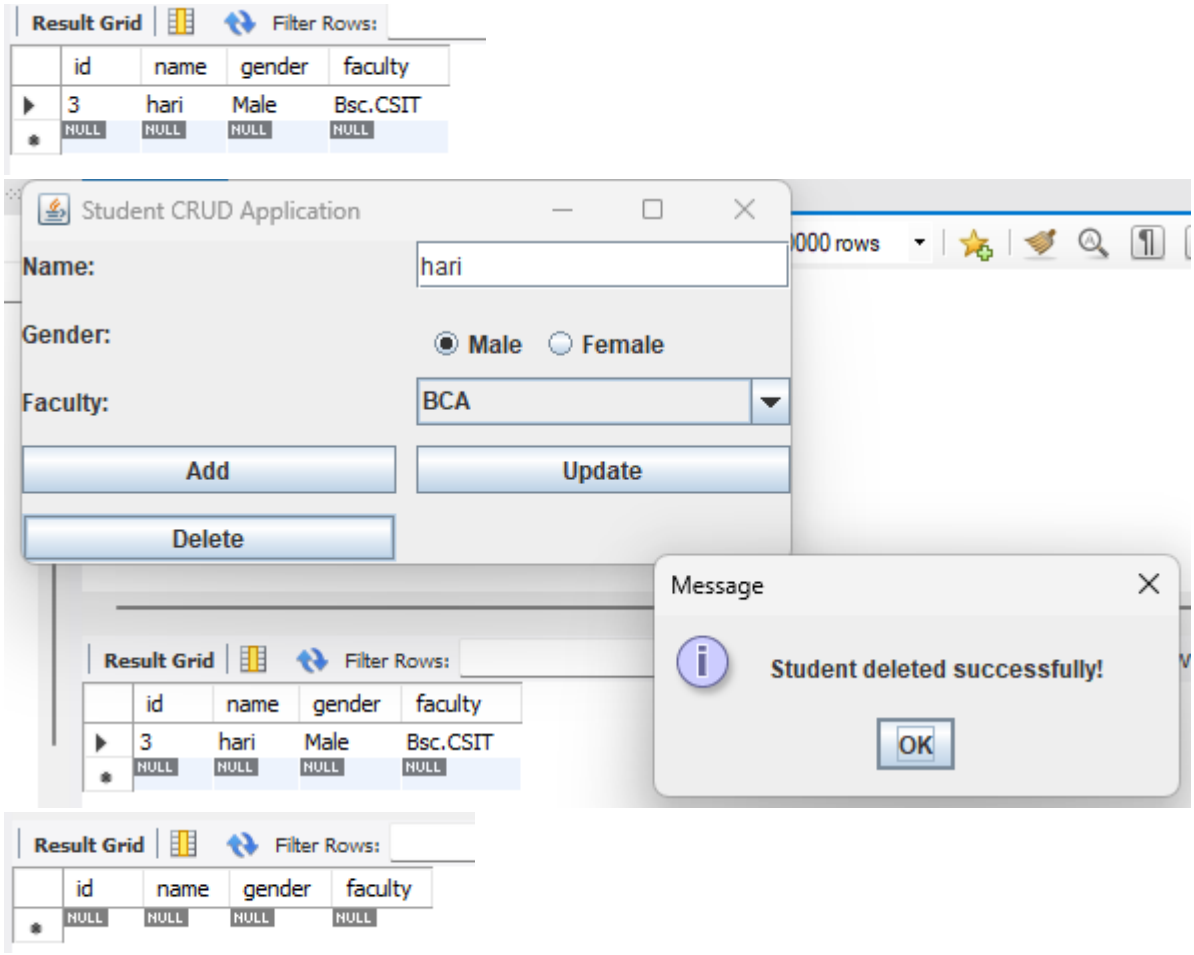
```
            int choice = JOptionPane.showConfirmDialog(null, "Are you sure you want to update
this student?");
            if (choice == JOptionPane.YES_OPTION) {
                student.setGender(gender);
                student.setFaculty(faculty);
                studentDAO.updateStudent(student);
                JOptionPane.showMessageDialog(null, "Student updated successfully!");
                clearForm();
            }
        } else {
            JOptionPane.showMessageDialog(null, "Student not found.");
        }
    }

    private void deleteStudent() {
        String name = nameField.getText();

        if (name.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Please enter a name.");
            return;
        }

        Student student = studentDAO.getStudentByName(name);
        if (student != null) {
            int choice = JOptionPane.showConfirmDialog(null, "Are you sure you want to delete
this student?");
            if (choice == JOptionPane.YES_OPTION) {
                studentDAO.deleteStudent(student.getId());
                JOptionPane.showMessageDialog(null, "Student deleted successfully!");
                clearForm();
            }
        } else {
            JOptionPane.showMessageDialog(null, "Student not found.");
        }
    }

    private void clearForm() {
        nameField.setText("");
        maleRadio.setSelected(true);
        facultyCombo.setSelectedIndex(0);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {
            @Override
            public void run() {
                new StudentCRUDAppUI();
            }
```

```
        });
    }
}
```


Student CRUD Application
Name:
hari
Gender:
Male
Female
Faculty:
Bsc.CSIT
Add
Update
Delete
Message
Student added successfully!
OK
Result Grid
Filter Rows:
id
name
gender
faculty
NULL


Result Grid
Filter Rows:
id
name
gender
faculty
3
hari
Male
Bsc.CSIT
NULL


Student CRUD Application
Name:
hari
Gender:
Male
Female
Faculty:
Bsc.CSIT
Add
Update
Delete
Message
Student updated successfully!
OK
Result Grid
Filter Rows:
id
name
gender
faculty
3
hari
Male
Bsc.CSIT
NULL



6.Write a program using javabeans to show the usage of propertyDescriptor and methodDescriptor.

```
package sumir.labreport;

import java.beans.BeanInfo;
import java.beans.IntrospectionException;
import java.beans.Introspector;
import java.beans.MethodDescriptor;
import java.beans.PropertyDescriptor;

public class Six {

    public static void main(String[] args) {
        try {
            // Get the BeanInfo for the Person class
            BeanInfo beanInfo = Introspector.getBeanInfo(Person.class);

            // Get the property descriptors
```

```
            PropertyDescriptor[] propertyDescriptors = beanInfo.getPropertyDescriptors();
            System.out.println("Property Descriptors:");
            for (PropertyDescriptor pd : propertyDescriptors) {
                System.out.println("Property Name: " + pd.getName());
                System.out.println("Property Type: " + pd.getPropertyType());
                System.out.println("Read Method: " + pd.getReadMethod());
                System.out.println("Write Method: " + pd.getWriteMethod());
                System.out.println();
            }

            // Get the method descriptors
            MethodDescriptor[] methodDescriptors = beanInfo.getMethodDescriptors();
            System.out.println("Method Descriptors:");
            for (MethodDescriptor md : methodDescriptors) {
                System.out.println("Method Name: " + md.getName());
                System.out.println("Method Parameters: " +
md.getMethod().getParameterCount());
                System.out.println();
            }
        } catch (IntrospectionException e) {
            e.printStackTrace();
        }
    }
}

class Person {

    private String name;
    private int age;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public void sayHello() {
        System.out.println("Hello, I'm " + name + " and I'm " + age + " years old.");
```

```
    }
}
```

```
Property Descriptors:
Property Name: age
Property Type: int
Read Method: public int sumir.labreport.Person.getAge()
Write Method: public void sumir.labreport.Person.setAge(int)

Property Name: class
Property Type: class java.lang.Class
Read Method: public final native java.lang.Class java.lang.Object.getClass()
Write Method: null

Property Name: name
Property Type: class java.lang.String
Read Method: public java.lang.String sumir.labreport.Person.getName()
Write Method: public void sumir.labreport.Person.setName(java.lang.String)

Method Descriptors:
Method Name: getClass
Method Parameters: 0

Method Name: getName
Method Parameters: 0

Method Name: sayHello
Method Parameters: 0

Method Name: setAge
Method Parameters: 1

Method Name: setName
Method Parameters: 1
```

8. Write a program to demonstrate session creation, setting and retrieving values form session and destruction of session in servlet. (Write the contents of html and web.xml files)

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
        version="4.0">
    <session-config>
        <session-timeout>
            30
```

```
        </session-timeout>
    </session-config>
    <servlet>
        <servlet-name>SessionServlet</servlet-name>
        <servlet-class>sumir.labreportservlet.SessionExampleServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>SessionServlet</servlet-name>
        <url-pattern>/sessionServlet</url-pattern>
    </servlet-mapping>
</web-app>
```

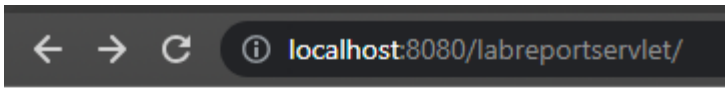
index.html

```
<!DOCTYPE html>
<html>
    <head>
        <title>Session Example</title>
    </head>
    <body>
        <h2>Session Example</h2>
        <form action="sessionServlet" method="POST">
            <label for="name">Name:</label>
            <input type="text" name="name" id="name" required><br><br>
            <label for="age">Age:</label>
            <input type="number" name="age" id="age" required><br><br>
            <input type="submit" value="Submit">
        </form>
    </body>
</html>
```

SessionExampleServlet.java

```
package sumir.labreportservlet;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

public class SessionExampleServlet extends HttpServlet {

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        String name = request.getParameter("name");
        int age = Integer.parseInt(request.getParameter("age"));
```

```
        HttpSession session = request.getSession();

        session.setAttribute("name", name);
        session.setAttribute("age", age);

        String sessionName = (String) session.getAttribute("name");
        int sessionAge = (int) session.getAttribute("age");

        response.setContentType("text/html");
        response.getWriter().println("<h2>Session Example</h2>");
        response.getWriter().println("<p>Name: " + sessionName + "</p>");
        response.getWriter().println("<p>Age: " + sessionAge + "</p>");

        session.invalidate();
    }
}
```

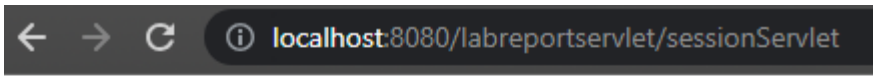


Session Example

Name: sad

Age: 22

Submit



Session Example

Name: sad

Age: 22

9. WAP that takes principal, rate and time values from a form and displays the simple interest value using JSP.

Web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
        xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
        version="4.0">
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <servlet>
        <servlet-name>CalculateServlet</servlet-name>
        <servlet-class>sumir.labreportservlet.CalculateServlet</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>CalculateServlet</servlet-name>
        <url-pattern>/calculate</url-pattern>
    </servlet-mapping>
</web-app>
```

interest.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <title>Simple Interest Calculator</title>
    </head>
    <body>
        <h1>Simple Interest Calculator</h1>
        <form action="calculate" method="post">
            <label for="principal">Principal:</label>
            <input type="text" id="principal" name="principal"><br><br>

            <label for="rate">Rate (%):</label>
            <input type="text" id="rate" name="rate"><br><br>

            <label for="time">Time (years):</label>
            <input type="text" id="time" name="time"><br><br>

            <input type="submit" value="Calculate">
        </form>
    </body>
</html>
```

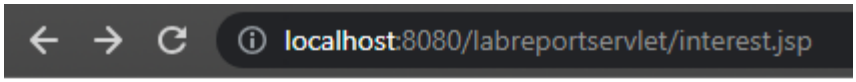
CalculateServlet.java

```
import java.io.IOException;
import javax.servlet.ServletException;
```

```
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class CalculateServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        double principal = Double.parseDouble(request.getParameter("principal"));
        double rate = Double.parseDouble(request.getParameter("rate"));
        double time = Double.parseDouble(request.getParameter("time"));

        double interest = (principal * rate * time) / 100;

        request.setAttribute("principal", principal);
        request.setAttribute("rate", rate);
        request.setAttribute("time", time);
        request.setAttribute("interest", interest);

        request.getRequestDispatcher("/result.jsp").forward(request, response);
    }
}
```

result.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
    <head>
        <title>Simple Interest Result</title>
    </head>
    <body>
        <h1>Simple Interest Result</h1>
        <p>Principal: ${principal}</p>
        <p>Rate: ${rate}</p>
        <p>Time: ${time}</p>
        <p>Simple Interest: ${interest}</p>
    </body>
</html>
```



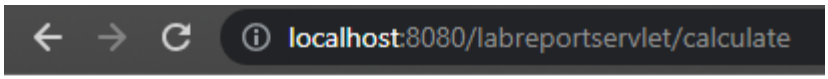
Simple Interest Calculator

Principal: 2000

Rate (%): 10

Time (years): 5

Calculate



Simple Interest Result

Principal: 2000.0

Rate: 10.0

Time: 5.0

Simple Interest: 1000.0

10. WAP to showcase the client server application using RMI.

ExampleRemoteInterface.java

```
import java.rmi.Remote;
import java.rmi.RemoteException;

public interface ExampleRemoteInterface extends Remote {

    String sayHello() throws RemoteException;
}
```

ExampleRemoteImplementation .java

```
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

public class ExampleRemoteImplementation extends UnicastRemoteObject implements
ExampleRemoteInterface {

    public ExampleRemoteImplementation() throws RemoteException {
```

```
        super();
    }

    @Override
    public String sayHello() throws RemoteException {
        return "Hello from the server!";
    }
}
```

Server.java

```
import java.rmi.Naming;
import java.rmi.registry.LocateRegistry;

public class Server {

    public static void main(String[] args) {
        try {
            ExampleRemoteImplementation remoteObj = new ExampleRemoteImplementation();

            // Create and start the RMI registry on port 1099
            LocateRegistry.createRegistry(1099);

            // Bind the remote object to the registry with a specific name
            Naming.rebind("ExampleRemote", remoteObj);

            System.out.println("Server is running...");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Client.java

```
import java.rmi.Naming;

public class Client {

    public static void main(String[] args) {
        try {
            // Look up the remote object by its name
            ExampleRemoteInterface remoteObj = (ExampleRemoteInterface)
Naming.lookup("rmi://localhost/ExampleRemote");

            // Invoke the remote method
            String result = remoteObj.sayHello();

            System.out.println("Server response: " + result);
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

```
PS E:\BCA\6th sem\np\LabReport\src\main\java\rmi> javac *.java
PS E:\BCA\6th sem\np\LabReport\src\main\java\rmi> java Server
Server is running...
```

```
PS E:\BCA\6th sem\np\LabReport\src\main\java\rmi> java Client
Server response: Hello from the server!
PS E:\BCA\6th sem\np\LabReport\src\main\java\rmi>
```