1. Aims of micro project

The aim of this micro project is to create a Hostel Leave Application Management System using Java Swing for the interface and Oracle Database for storing data. The system allows students to submit leave applications, which are then managed by department admins and reviewed by the principal. It includes input validations, secure admin access, and displays leave records in a clear and organized way. The project helps simplify the leave approval process and improves record-keeping in hostels.

2. Proposed Methodology

1. Class Structure

The system is built using Java Swing for the user interface and connects to an Oracle database.

The main class manages different panels: student form, admin panel, and principal panel.

Leave application details like name, roll number, department, room, dates, and reason are handled here.

2. Data Initialization

When the system starts, it shows a main window with three tabs: Student Leave, Admin Panel, and Principal Panel.

Admin department names and passwords are stored in a map for verification.

3. Student Leave Application

Students can fill in a form to apply for leave.

Fields like name, roll number, room number, and dates must follow specific formats.

If valid, the data is saved into the Oracle database.

4. Admin Panel Access

Each department has a button in the Admin Panel.

When clicked, it asks for the department password.

If the password is correct, it shows the leave applications of that department.

5. Principal Panel Access

The principal can enter a date and view all student applications submitted for that date.

A table is shown with student details and their leave info from all departments.

6. User Interaction

The application runs with a graphical interface.

Users can switch between tabs (Student, Admin, Principal) and perform actions.

Buttons are used to submit forms and view records.

3. Action Plane

Sr. No.	Detail of activity	Plan Start Date	Plan Finish Date
1.	Gather requirements and understand manual leave process	01/04/2025	02/04/2025
2.	Design UI layout using Java Swing	03/04/2025	04/04/2025
3.	Create student form with input validations	05/04/2025	06/04/2025
4.	Connect application to Oracle database	07/04/2025	08/04/2025
5.	Build Admin and Principal panels for application management	09/04/2025	15/04/2025
6.	Test form submission, password check, and data display	11/04/2025	12/04/2025
7.	Write project report with screenshots and code explanation	13/04/2025	14/04/2025
8.	After confirmation the report was printed	15/04/2025	15/04/2025

4. Brief Description of Micro project

The Hostel Leave Application System is a simple Java Swing project that helps students submit leave applications online. Instead of using paper forms, students can fill in their details like name, roll number, room number, leave dates, and reason in the software.

The system has three main parts: Student Form, Admin Panel, and Principal Panel. Each department's admin can log in with a password to see the leave requests of their students. The principal can check all leave applications for a particular date.

The system connects to an Oracle database, where all the student data is saved. It also checks that the entered information is correct, like making sure names have only letters and roll numbers have exactly 4 digits.

This project makes the leave process faster, easier, and more organized for students, admins, and the principal.

4.1 Used Concepts

The Hostel Leave Application System is built using Java Swing for the user interface and Oracle Database for storing student leave data. It follows the object-oriented programming approach and uses GUI components, event handling, and database connectivity to manage leave applications effectively.

4.1.1 Classes and Objects

The system is built using object-oriented principles. Two main classes are:

1. StudentForm: Handles student leave form data input and validation.
2. AdminPanel / PrincipalPanel: Manage and display leave requests.

Each class has its own attributes and methods, which helps keep the code modular and easy to maintain.

4.1.2 Data Members and Member Functions

Data Members store form data:

studentName, rollNumber, roomNumber, leaveFrom, leaveTo, reason, department, password.

Member Functions handle logic:

1. validateForm(): Ensures input format is correct.
2. submitToDatabase(): Saves the form to Oracle DB.
3. viewLeaveApplications(): Fetches and displays saved applications.

4.1.3 Constructor Overloading

The system may use multiple constructors to create objects in different ways.

Example: Student() for empty object, and Student(String name, int rollNo) to directly assign values.

This improves flexibility in object creation and improves code readability.

4.1.4 Input Validation

Validation is done before data submission:

1. Name: Only letters allowed
2. Roll number: Exactly 4 digits
3. Room number: Exactly 2 digits

This ensures only clean and accurate data enters the system.

4.1.5 GUI using Java Swing

Java Swing is used to build the user interface.

Components like JTextField, JButton, JLabel, and JPanel help create forms and display data.

Event handling (e.g., ActionListener) captures button clicks to trigger actions.

4.1.6 Database Connectivity (JDBC)

Uses Java Database Connectivity (JDBC) to connect to Oracle DB.

Performs operations like:

1. INSERT for saving leave requests
2. SELECT for fetching leave records

Connection handled using Connection, PreparedStatement, and ResultSet.

4.1.7 Secure Access with Passwords

Admin and Principal panels are password protected.

Verifies credentials before showing sensitive data like leave requests.

4.2 Programs or code of relevant concepts and output

```
import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.sql.*;
import java.util.HashMap;
import java.util.Map;
public class StudentLeaveApplications
{
   private static final Map<String, String> departmentPasswords = new HashMap<>();
   static {
      departmentPasswords.put("Computer Science", "cs123");
      departmentPasswords.put("Electronics", "elec123");
      departmentPasswords.put("Civil", "civil123");
      departmentPasswords.put("Electrical", "elec456");
      departmentPasswords.put("Information Technology", "it123");
      departmentPasswords.put("Mechanical", "mech123");
   }
   public static void main(String[] args) {
      SwingUtilities.invokeLater(StudentLeaveApplications::createMainGUI); }

   private static void createMainGUI() {
      JFrame frame = new JFrame("Hostel Leave Application");
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      frame.setSize(1200, 700);
```

```
        frame.setLayout(new BorderLayout());


        JPanel headerPanel = new JPanel(new GridLayout(2, 1));
        headerPanel.setBackground(new Color(0, 102, 204));
        JLabel titleLabel = new JLabel("Shri Siddheshwar Women's Polytechnic, Solapur", SwingConstants.CENTER);
        titleLabel.setFont(new Font("Times New Roman", Font.BOLD, 26));
        titleLabel.setForeground(Color.WHITE);


        JLabel subHeadingLabel = new JLabel("Hostel Leave Application", SwingConstants.CENTER);
        subHeadingLabel.setFont(new Font("Times New Roman", Font.PLAIN, 20));
        subHeadingLabel.setForeground(Color.WHITE);


        headerPanel.add(titleLabel);
        headerPanel.add(subHeadingLabel);
       JTabbedPane tabbedPane = new JTabbedPane();
        tabbedPane.addTab("Student Leave", createStudentFormPanel());
        tabbedPane.addTab("Admin Panel", createAdminPanel());
        tabbedPane.addTab("Principal Panel", createPrincipalPanel());


        frame.add(headerPanel, BorderLayout.NORTH);
        frame.add(tabbedPane, BorderLayout.CENTER);
        frame.setVisible(true);
    }
    private static JPanel createStudentFormPanel()     {
        JPanel panel = new JPanel(new GridLayout(9, 2, 10, 10));
        panel.setBackground(new Color(204, 229, 255));
        String[] departments = {"Computer Science", "Electronics", "Civil", "Electrical", "Information Technology", "Mechanical"};
```

```
JComboBox  deptComboBox = new JComboBox(departments);
JTextField nameField = new JTextField();
JTextField rollField = new JTextField();
JTextField roomField = new JTextField();
JTextField leaveDateField = new JTextField();
JTextField reportDateField = new JTextField();
JTextArea reasonArea = new JTextArea(3, 20);
JButton submitButton = new JButton("Submit");
Font labelFont = new Font("Times New Roman", Font.BOLD, 22);
panel.add(createBoldLabel("Department:", labelFont));
panel.add(deptComboBox);
panel.add(createBoldLabel("Name:", labelFont));
panel.add(nameField);
panel.add(createBoldLabel("Roll Number:", labelFont));
panel.add(rollField);
panel.add(createBoldLabel("Room Number:", labelFont));
panel.add(roomField);
panel.add(createBoldLabel("Leave Date (DD-MM-YYYY):", labelFont));
panel.add(leaveDateField);
panel.add(createBoldLabel("Report Date (DD-MM-YYYY):", labelFont));
panel.add(reportDateField);
panel.add(createBoldLabel("Reason:", labelFont));
panel.add(new JScrollPane(reasonArea));
panel.add(new JLabel());
panel.add(submitButton);
submitButton.addActionListener(e -> {
    String dept = (String) deptComboBox.getSelectedItem();
    String name = nameField.getText().trim();
    String roll = rollField.getText().trim();
```

```
        String room = roomField.getText().trim();
        String leaveDate = leaveDateField.getText().trim();
        String reportDate = reportDateField.getText().trim();
        String reason = reasonArea.getText().trim();
        if (!name.matches("[a-zA-Z ]+")){
           JOptionPane.showMessageDialog(null, "Name must contain only letters!");
        } else if (!roll.matches("\\d{4}"))
           JOptionPane.showMessageDialog(null, "Roll Number must be exactly 4 digits!");
        } else if (!room.matches("\\d{2}"))
        {
           JOptionPane.showMessageDialog(null, "Room Number must be exactly 2 digits!")
        } else
        {
          try {
              insertIntoDatabase(dept, name, roll, room, leaveDate, reportDate, reason);
              JOptionPane.showMessageDialog(null, "Leave Application Submitted
Successfully!");
                 } catch (Exception ex){
              JOptionPane.showMessageDialog(null, "Error: " + ex.getMessage());}
        }
    });return panel;  }
  private static JPanel createAdminPanel() {
    JPanel panel = new JPanel(new GridLayout(3, 2, 10, 10));
    panel.setBackground(new Color(204, 229, 255));
    for (String dept : departmentPasswords.keySet()) {
       JButton deptButton = new JButton(dept);
       deptButton.addActionListener(e -> verifyAndShowDepartmentData(dept));
       panel.add(deptButton);
    } return panel;
```

```
    }
    private static JPanel createPrincipalPanel()
            {
        JPanel panel = new JPanel(new BorderLayout());
        panel.setBackground(new Color(204, 229, 255));
        JTextField dateField = new JTextField(20);
        JButton viewButton = new JButton("View Applications");
        DefaultTableModel model = new DefaultTableModel(new String[]{"Name", "Roll", "Room", "Dept", "Leave Date", "Report Date", "Reason"}, 0);
        JTable table = new JTable(model);
        viewButton.addActionListener(e -> fetchApplicationsByDate(dateField.getText(), model));
        JPanel inputPanel = new JPanel();
        inputPanel.setBackground(new Color(204, 229, 255));
        inputPanel.add(new JLabel("Enter Date (DD-MM-YYYY):"));
        inputPanel.add(dateField);
        inputPanel.add(viewButton);
        panel.add(inputPanel, BorderLayout.NORTH);
        panel.add(new JScrollPane(table), BorderLayout.CENTER);
        return panel;
    }
    private static void insertIntoDatabase(String department, String name, String roll, String room, String leaveDate, String reportDate, String reason) throws SQLException
            {
 try (Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system", "vaishnavi"))
                    {
            String sql = "INSERT INTO application (department, s_name, roll_number, room_number, leave_date, report_date, reason) VALUES (?, ?, ?, ?, TO_DATE(?, 'DD-MM-YYYY'), TO_DATE(?, 'DD-MM-YYYY'), ?)";
            PreparedStatement stmt = conn.prepareStatement(sql);
```

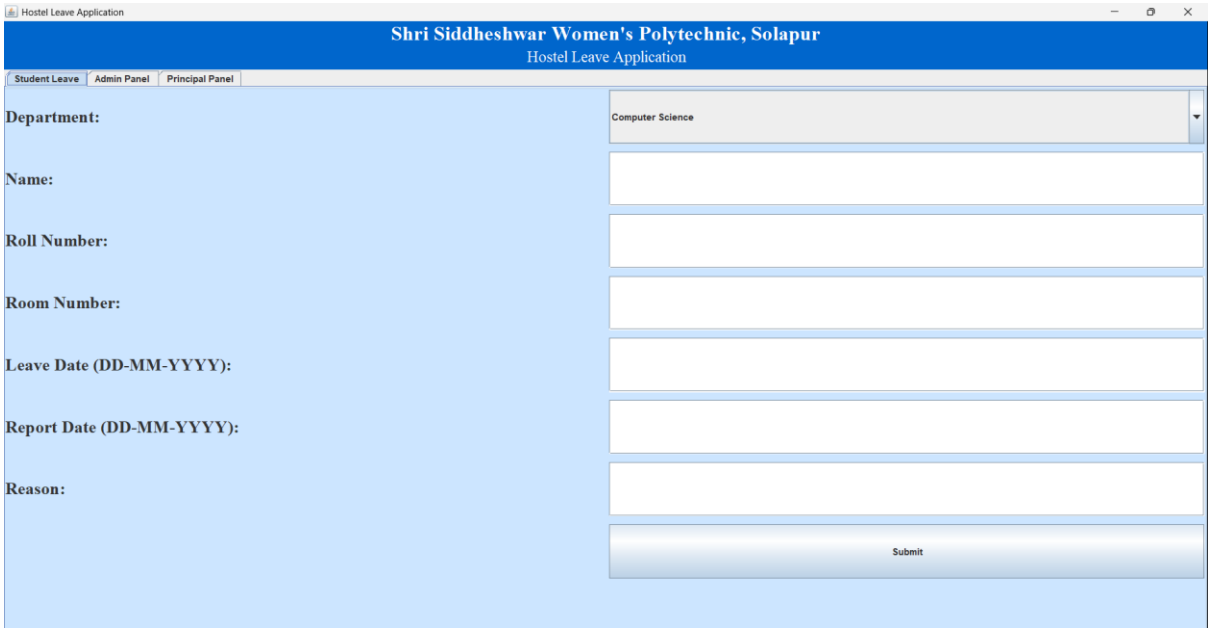
```
            stmt.setString(1, department);
            stmt.setString(2, name);
            stmt.setString(3, roll);
            stmt.setString(4, room);
            stmt.setString(5, leaveDate);
            stmt.setString(6, reportDate);
            stmt.setString(7, reason);
            stmt.executeUpdate();
        }
    }
    private static void verifyAndShowDepartmentData(String department) {
        String password = JOptionPane.showInputDialog("Enter password for " + department +
" department:");
        if (password != null && password.equals(departmentPasswords.get(department))) {
            showDepartmentData(department);
        } else
            {
            JOptionPane.showMessageDialog(null, "Incorrect password!");
        }
    }
    private static void showDepartmentData(String department) {
        JFrame frame = new JFrame(department + " Applications");
        frame.setSize(800, 400);
        DefaultTableModel model = new DefaultTableModel();
        JTable table = new JTable(model);
        model.addColumn("Name");
        model.addColumn("Roll");
        model.addColumn("Room");
        model.addColumn("Leave Date");
```

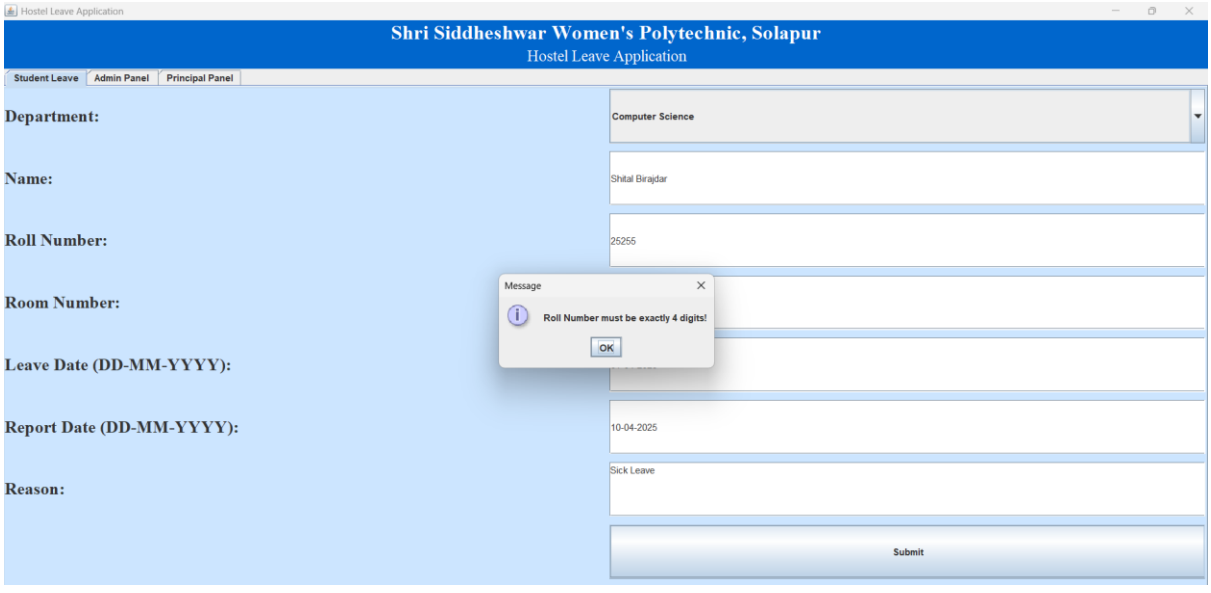
```
        model.addColumn("Report Date");
        model.addColumn("Reason");
        fetchDataForDepartment(model, department);
        frame.add(new JScrollPane(table));
        frame.setVisible(true);
    }
    private static void fetchApplicationsByDate(String date, DefaultTableModel model) {
try (Connection con=DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe",
"system", "vaishnavi"))
{
 String sql = "SELECT s_name, roll_number, room_number, department,
TO_CHAR(leave_date, 'DD-MM-YYYY'), TO_CHAR(report_date, 'DD-MM-YYYY'),
reason FROM application WHERE leave_date = TO_DATE(?, 'DD-MM-YYYY')";
            PreparedStatement stmt = conn.prepareStatement(sql);
            stmt.setString(1, date);
            ResultSet rs = stmt.executeQuery();
            model.setRowCount(0);
            while (rs.next()) {
                model.addRow(new Object[]{
                        rs.getString(1),
                        rs.getString(2),
                        rs.getString(3),
                        rs.getString(4),
                        rs.getString(5),
                        rs.getString(6),
                        rs.getString(7)
                });
            }
        } catch (Exception ex)
            {
```

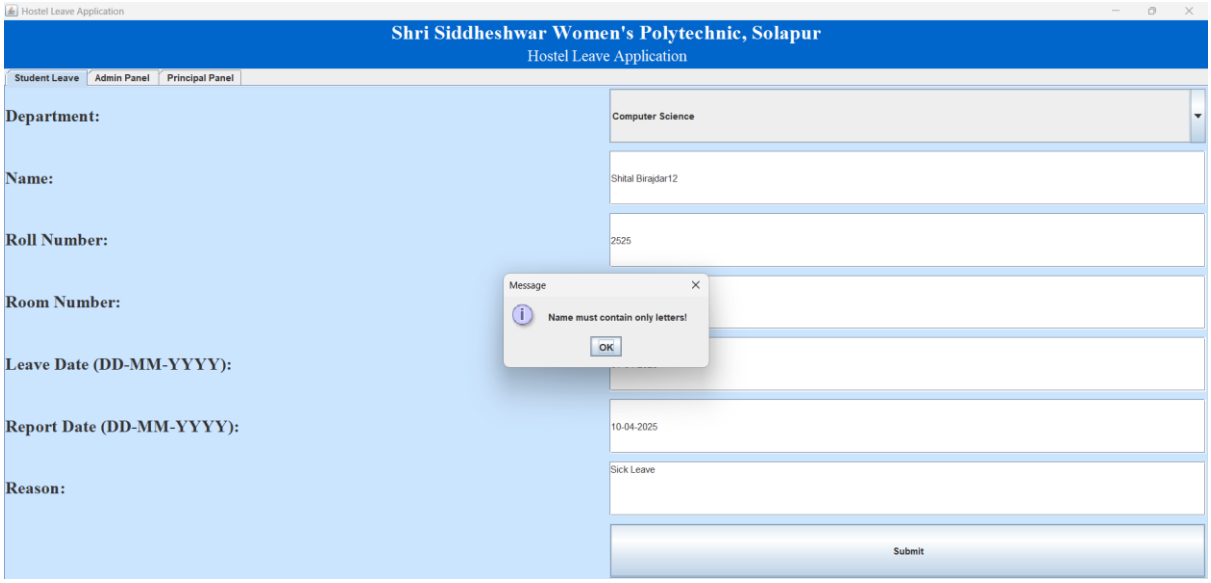
```
        JOptionPane.showMessageDialog(null, "Error fetching data: " + ex.getMessage());
      }
   }
   private static void fetchDataForDepartment(DefaultTableModel model, String department){
      try (Connection conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "system", "vaishnavi"))
 {
        String sql = "SELECT s_name, roll_number, room_number, TO_CHAR(leave_date, 'DD-MM-YYYY'), TO_CHAR(report_date, 'DD-MM-YYYY'), reason FROM application WHERE department = ?";
        PreparedStatement stmt = conn.prepareStatement(sql);
        stmt.setString(1, department);
        ResultSet rs = stmt.executeQuery();
        model.setRowCount(0);
        while (rs.next())
        {
           model.addRow(new Object[]{
                rs.getString(1),
                rs.getString(2),
                rs.getString(3),
                rs.getString(4),
                rs.getString(5),
                rs.getString(6)
           });
        }
      }
        catch (Exception ex)
        {
        JOptionPane.showMessageDialog(null, "Error fetching data: " + ex.getMessage());
      }
```

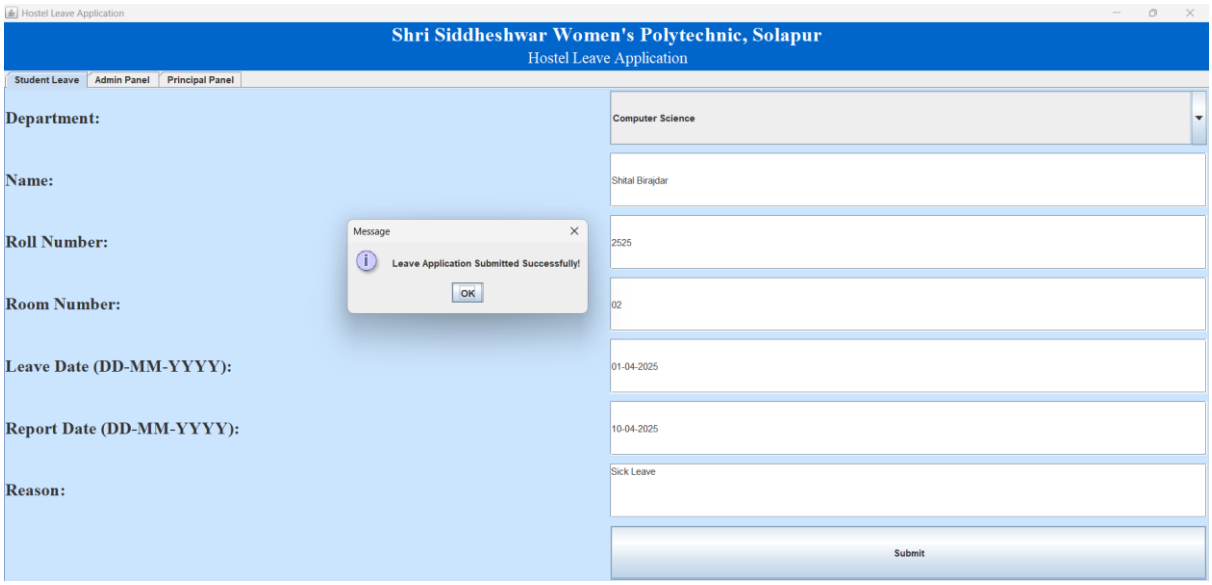
```
    }
    private static JLabel createBoldLabel(String text, Font font) {
        JLabel label = new JLabel(text);
        label.setFont(font);
        return label;
    }
}
```

5. Outputs

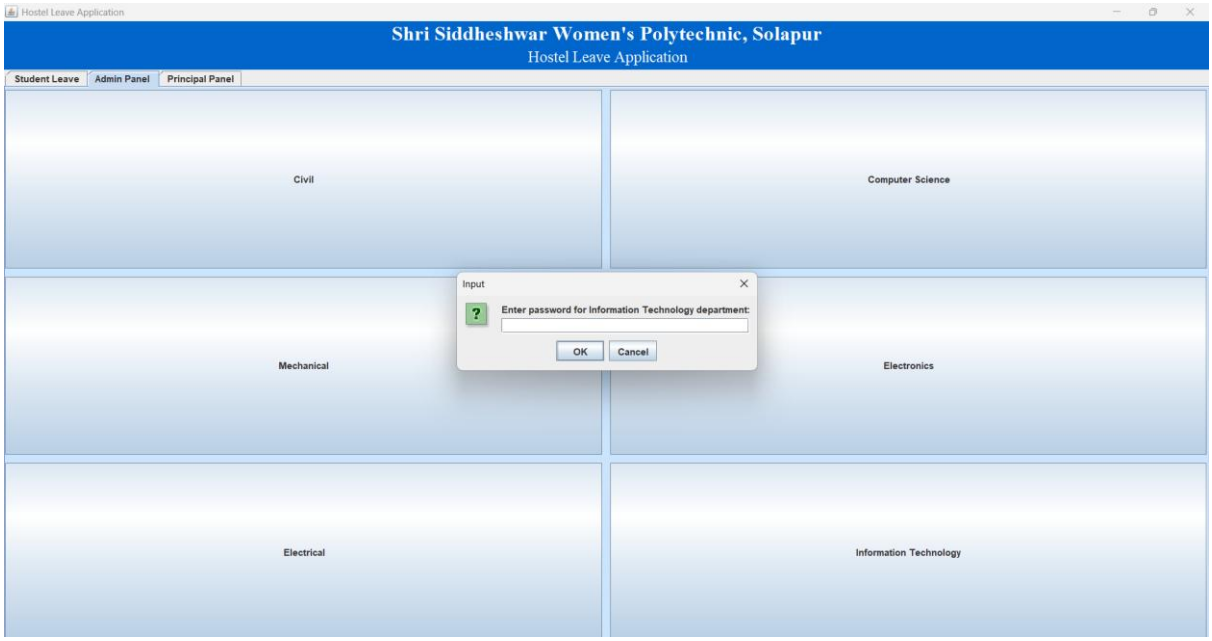





Hostel Leave Application
Shri Siddheshwar Women's Polytechnic, Solapur
Hostel Leave Application
Student Leave
Admin Panel
Principal Panel
Department:
Computer Science
Name:
Shital Birajdar12
Roll Number:
2525
Room Number:
Message
Name must contain only letters!
OK
Leave Date (DD-MM-YYYY):
Report Date (DD-MM-YYYY):
10-04-2025
Reason:
Sick Leave
Submit

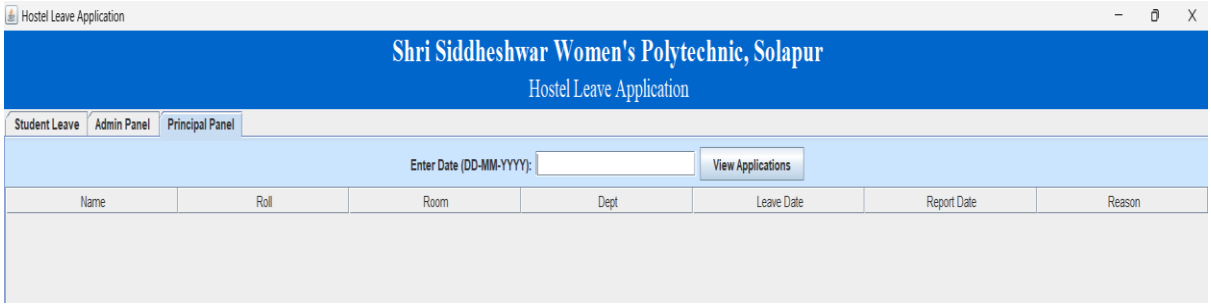

Hostel Leave Application
Shri Siddheshwar Women's Polytechnic, Solapur
Hostel Leave Application
Student Leave
Admin Panel
Principal Panel
Department:
Computer Science
Name:
Shital Birajdar
Roll Number:
2525
Message
Leave Application Submitted Successfully!
OK
Room Number:
02
Leave Date (DD-MM-YYYY):
01-04-2025
Report Date (DD-MM-YYYY):
10-04-2025
Reason:
Sick Leave
Submit

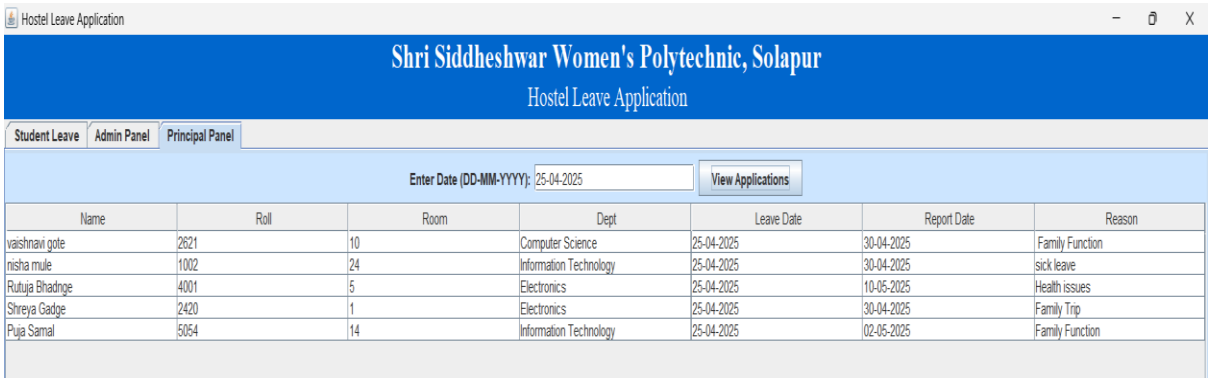

Hostel Leave Application
Shri Siddheshwar Women's Polytechnic, Solapur
Hostel Leave Application
Student Leave
Admin Panel
Principal Panel
Civil
Computer Science
Mechanical
Electronics
Electrical
Information Technology



Information Technology Applications

Name	Roll	Room	Leave Date	Report Date	Reason
Vaishnavi Gote	2621	12	10-04-2025	15-04-2025	Family Function
vaishnavi gote	2621	10	10-04-2025	25-04-2025	Family function
sakshi gote	2021	10	04-04-2025	06-04-2025	sick leave
nisha mule	1002	24	25-04-2025	30-04-2025	sick leave





6. Advantages

1. **Easy Leave Management**: Students can fill out and submit leave forms quickly. Admins and the principal can check all records easily.
2. **Correct Data Entry**: The system checks the details entered by students, like name, roll number, and room number, so there are fewer mistakes.
3. **Safe Access**: Only authorized admins and the principal can log in using passwords. This keeps student data safe.
4. **Data Stored in Database**: All leave records are saved in an Oracle database, so they can be checked anytime without losing information.
5. **No Need for Paper**: The whole process is online, so there's no need to use paper forms anymore.

7. Disadvantages

1. **Limited to One User at a Time**: The system can only be used by one student or admin at a time. It doesn't support multiple users working at once.
2. **No Auto Backup**: If the system crashes or is closed suddenly, some unsaved data might be lost unless it's already stored in the database.
3. **Basic Security**: While there is password protection, the system doesn't have advanced security like encryption or OTP verification.
4. **Fixed Fields Only**: The form has fixed input fields. If more details are needed later (like parent contact or hostel warden approval), the code must be changed.
5. **No Online Access**: The application is offline, so users must be on the same system. There's no way to apply or approve leave remotely.

8. Applications

1. **Hostel Leave Management**: Used by hostel students to submit leave requests to wardens or principals.
2. **College Attendance Monitoring**: Can be modified to track student presence during leave periods.
3. **Employee Leave System**: Useful for offices or small companies to manage staff leave requests.
4. **School Leave Requests**: Can be used in schools for students to submit sick or personal leave requests.
5. **Club or Group Activity Planning**: Members can submit unavailability through leave forms for planning meetings or events.
6. **Medical Leave Records**: Can be adapted to track medical leaves and generate reports for analysis.

10. Conclusion

The Hostel Leave Application System is a useful project made using Java. It helps students to apply for leave easily and allows admins and principals to check and manage those applications. The project shows important Java features like classes, methods, and connecting to a database. It saves time and reduces paper work by making the process digital. The system works well for small hostels, and in the future, it can be improved by adding features like email alerts and better security. Overall, it is a helpful and easy-to-use system.

11. Reference

1. J. Gosling and H. McGilton, "The Java Language Environment", 1996.
2. T. Lindholm and F. Yellin, The Java Virtual Machine Specification, Reading, Mass., 1997
3. R. Ege, "Database support for object-oriented simulation", International Journal of Systems Engineering, 1994.
4. N. K. Balcoh, M. H. Yousaf, W. Ahmad and M. I. Baig, "Algorithm for efficient attendance management: Face recognition-based approach", Int. J. Comput. Sci., vol. 9, no. 4, pp. 146, Jul. 2012.