DIGITAL NOTICE

AND REGISTRATION SYSTEM

Group-13 members:

Aiswarya M (FIT24CS021)
Amrita Murthi (FIT24CS033)
Ananya Nair (FIT24CS036)
Angeleena Ann Shibu (FIT24CS039)
Angelina Johnson (FIT24CS040)
Antreena Babu (FIT24CS053)

Guided by : Dr. Paul P Mathai
[Dept of Computer Sci. & Engg.]

INTRODUCTION

Problem Statement and Motivation

In contemporary academic institutions, the efficient and reliable dissemination of official notices and event information is paramount for fostering student engagement and ensuring administrative compliance.Traditional methods of disseminating crucial institutional information, such as reliance on physical notice boards and peer-to-peer messaging applications (e.g., WhatsApp), present significant limitations in a modern academic environment. Physical notices suffer from low visibility and are prone to being overlooked or damaged, while high-volume chat groups result in a phenomenon of information saturation, causing crucial updates to be easily buried or lost, thereby compromising information accessibility and trackability. This fractured communication environment leads directly to missed opportunities for student participation and presents significant challenges for reliable record-keeping.

Proposed Solution: Digital Notice and Event Registration System

To address these pronounced shortcomings, this project introduces the development of a Digital Notice and Event Registration System. The Digital Notice and Event Registration System is designed as a centralized, secure, and authenticated digital platform implemented in Java, intended to serve as the definitive source for all official institutional announcements and event management.

The core functionalities of the DNBEMS are designed to provide:

-Centralized Access: A single, secure platform where all official notices and event details are published.

-Student Engagement: A dedicated portal allowing students to login, view notices, acknowledge receipt of important updates, and register for events seamlessly.

-Advanced Filtering: Students can efficiently search and filter notices, utilizing key criteria such as department name to quickly locate relevant information.

-Administrative Control: An Admin Dashboard empowers authorized personnel to easily add, modify, or remove notices and event listings, ensuring real-time accuracy.

Ultimately, this project delivers a unified, secure digital hub for college communication. The system ensures that all students have equitable, traceable access to notices and events, thereby solving the persistent problem of missed information and establishing a foundation for reliable, real-time record-keeping

OBJECTIVES

The primary aim of the Digital Notice and Event Registration System (DNERS) is to transition the college from traditional, scattered communication methods to a centralized, efficient, and interactive digital platform.

Functional Objectives (User-Centric Goals)

-*Centralization and Accessibility*: To develop a single, secure, and easily accessible digital platform for publishing all official college notices and event information, eliminating reliance on disparate physical and digital communication methods.
-*Effective Information Retrieval*: To implement robust search and filter functionality allowing students to efficiently locate relevant notices using keywords, notably department name, enhancing the usability of the notice board.
-*Notice Acknowledgment and Accountability:* To provide a mechanism for students to digitally acknowledge the receipt and viewing of critical notices, thereby offering administrators a verifiable record of compliance and readership.
-*Streamlined Event Registration:* To allow students to browse event details and complete secure, direct online registration via the platform, replacing manual sign-up processes.

Technical Objectives (System-Centric Goals)

-*Secure Database Integration:* To design and implement a structured database schema for the reliable storage of all entities: student profiles, notices, event details, registration records, and acknowledgement logs, facilitating easy and fast data retrieval.
-*Role-Based Access Control:* To secure the system with authenticated dual-role login (Student and Administrator), ensuring that content management (CRUD operations) is restricted to authorized administrative personnel.
-*Maintainable Architecture (Java-Based):* To build the system using a modular Java architecture, promoting code reusability, easy maintenance, and scalability for future enhancements.

EXISTING SYSTEMS AND DISADVANTAGES

The development of the Digital Notice and Event Registration System is necessitated by the significant shortcomings present in the institution's existing, decentralized methods of communication. The following analysis identifies the critical limitations associated with traditional tools, highlighting the environment of information fragmentation and administrative inefficiency that the proposed system seeks to resolve

1. Physical Notice Boards

-Description : The traditional, paper-based method for posting announcements in designated physical locations.

-Disadvantages :

Limited Reach & Visibility: Information is constrained by location and time; notices are easily missed or expire unnoticed.

Zero Tracking: Provides no audit trail or tracking mechanism to confirm student readership or acknowledgment.

2. Email Systems

-Description : Official communication distributed via institutional email accounts.

-Disadvantages :

Low Engagement: Emails are frequently ignored or filtered to spam, leading to critical announcements being overlooked.

Lack of Personalization: Mass emails lack efficient filtering, resulting in information overload for the recipient.

3.WhatsApp Groups

-Description : Informal messaging platforms used for quick announcements and group discussions.

-Disadvantages :

Information Overload: Notices get quickly buried amidst general conversation and excessive message volume.

Poor Organization & Record-Keeping: Lacks search functionality, formal archival, and integration with official college records.

4. Learning Management Systems (LMS)

-Description : Academic platforms (like Moodle or Google Classroom) focused on course content and assignments.

-Disadvantages :

Limited Scope: Primarily academic-focused, often possessing rudimentary event management features and lacking robust, dedicated noticeboard functionality.

Siloed Information: Notices are tied to course structures rather than centralized campus communication.

5. Google Forms (or similar)

-Description : External, separate tools used solely for collecting event registration data.

-Disadvantages :

Lack of Integration: Acts as a siloed tool detached from the communication and notice platform.

No Comprehensive Management: Cannot manage the entire event lifecycle (notice, registration, tracking, participation history) in one place.

In conclusion, the combined drawbacks—including limited reach, information overload, lack of acknowledgment tracking, and poor integration—demonstrate that current methods are insufficient for modern academic communication. This analysis unequivocally justifies the need for the Digital notice and registration system as a unified, digital solution to establish centralized, reliable, and traceable information exchange.

PROPOSED SYSTEM AND ADVANTAGES

Proposed system:

The Digital Notice and Event Registration System is designed as a secure, centralized, and role-based platform that directly addresses the shortcomings of current communication methods. The system's architecture focuses on efficiency, real-time updates, and verifiable record-keeping. The system is built around a few critical components that define its functionality and user experience:

- User Authentication and Access Control:
  The system mandates secure user login via a unique username and password .This ensures that only authorized students can access personalized notices and register for events, and restricts content management functions to verified administrators.The design implements role-based access control (RBAC) to differentiate between student and admin privileges.
- Centralized Digital Platform:
  The Digital notice and registration system serves as a single-point access hub for all official campus communication. It features an intuitive and user-friendly interface (UI) designed for easy navigation and quick information retrieval, solving the problem of notices being scattered or hard to find.
- Student-Facing Functionality:
  The Student Portal is designed for maximum accessibility and personalized information retrieval. After performing a secure, authenticated login, students are granted real-time access to the centralized platform where they can view all official notices and upcoming events. A core feature is the robust search and filter functionality, which allows students to efficiently narrow down announcements using key criteria, such as the department name, to overcome information overload. Furthermore, the portal enables seamless event registration directly within the system, eliminating the need for external tools and scattered links. For accountability, students can digitally acknowledge critical notices. Finally, a personalized dashboard gives each student an organized view of their current registered events and participation history, facilitating better planning and engagement.

- Administrative Features

The Admin Dashboard provides authorized staff and faculty with secure, role-based access to actively manage and monitor institutional communication. Administrators can efficiently post new official notices (including rich text, links, and attachments) and create detailed event listings by defining all necessary parameters such as the event name, date, time, venue, and sponsoring department or club. Crucially, the system offers integrated monitoring and reporting tools. These tools allow admins to track the effectiveness of communication by viewing which students have submitted a digital acknowledgement for specific notices, and to easily manage participant logistics by viewing and exporting comprehensive registration lists for any college event. This ensures efficient oversight, reliable record-keeping, and improved accountability across all campus activities.

Advantages of the proposed system

The Digital Notice and Event Registration System offers significant functional and strategic advantages over the previous, fragmented communication methods (physical boards, emails, and informal groups). These advantages directly contribute to improved administrative efficiency and heightened student engagement.

Operational and Administrative Advantages

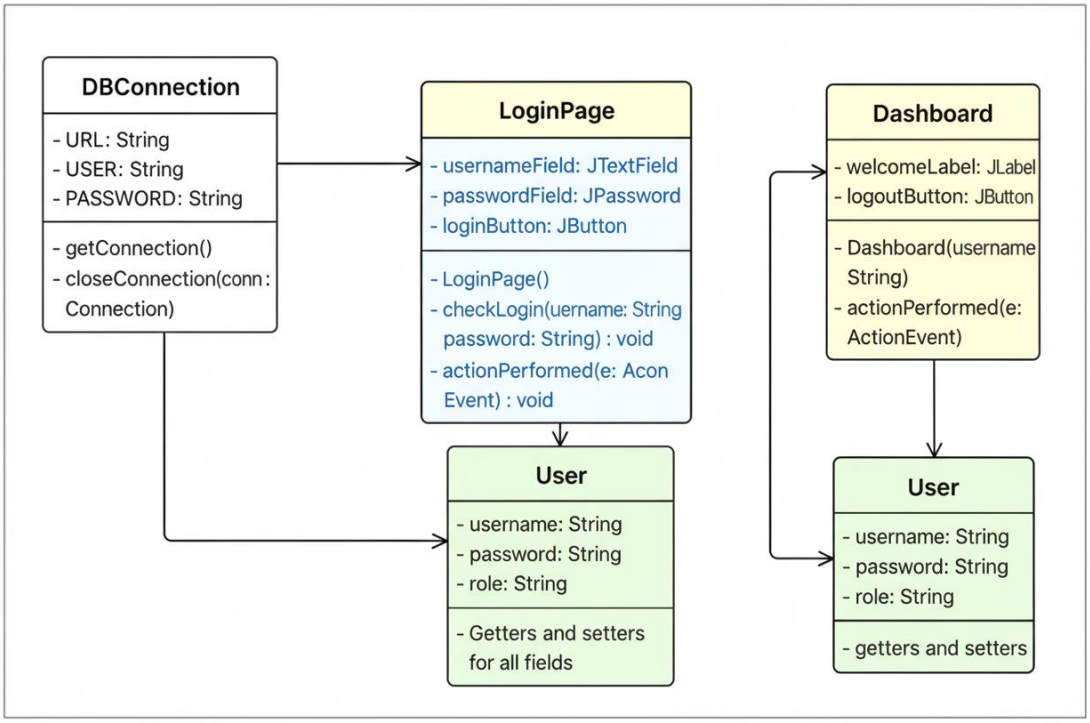
- Centralized Control and Data Integrity:
  The system establishes a single, authoritative source for all information, eliminating version conflicts and ensuring data consistency.
- Accountability
  The Notice Acknowledgement feature creates a verifiable audit trail, allowing administrators to confirm precisely which students have read critical announcements. This is impossible with physical boards or general emails.
- Streamlined Event Management:
  Event creation, publication, and participant tracking are unified in one platform. Administrators can easily export registration lists directly from the system, drastically reducing manual effort and potential errors associated with separate tools like Google Forms.
- Improved Administrative Efficiency:
  Reduces the time and resources spent on printing, manually posting, and tracking notices across physical locations and multiple digital platforms.

Student-Centric Advantages

- Guaranteed Access and Visibility:
  Students gain access to all official information from any location, eliminating the geographical constraint and time lag of physical notice boards. The centralized nature ensures no information is 'buried' or lost.
- Targeted Information Retrieval:
  The powerful search and filter function, keyed by attributes like Department Name, prevents information overload. Students can efficiently filter the high volume of campus announcements to see only what is relevant to them.
- Enhanced Engagement and Planning:
  The Personalized Student Portal allows students to view their registered events and participation history, helping them better track and plan their involvement in college activities.
- Simplified Interaction:
  The integrated Event Registration process is seamless and intuitive, encouraging higher participation rates compared to navigating external links and tools.

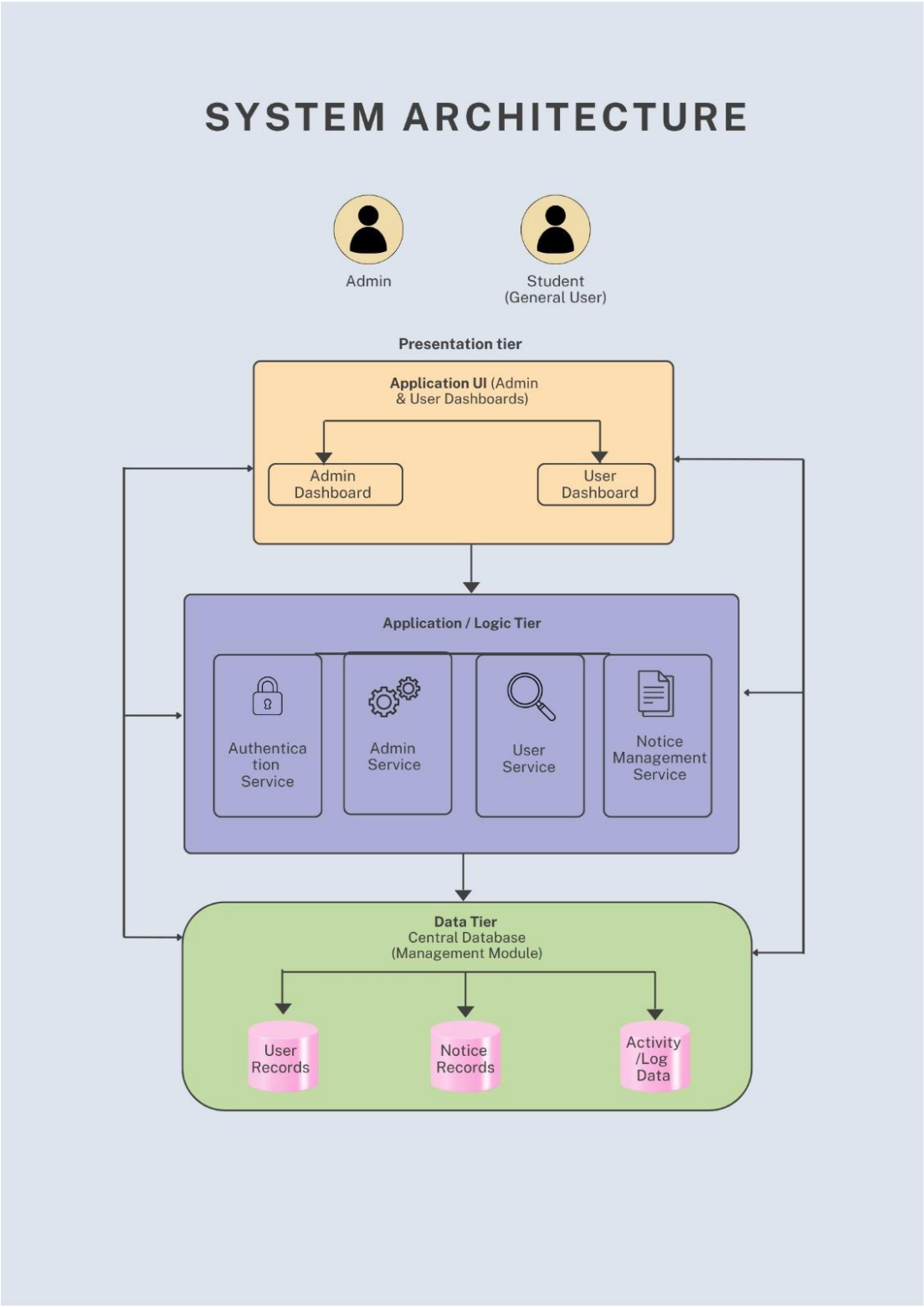
SYSTEM DESIGN

UML Class Diagram :



The UML Class Diagram outlines the static structure of the Digital Notice and Event Registration System by defining four core Java classes and their relationships. The system's foundational component is the **DBConnection** class, which centralizes database connection handling through attributes like URL, USER, and PASSWORD, and methods like getConnection(). The user's interaction starts at the **LoginPage**, which handles secure user input fields, checks credentials using the checkLogin() method, and depends on DBConnection to access stored data. User credentials and roles are modeled in the **User** class, which holds the username, password, and role, enabling Role-Based Access Control. Upon successful authentication, the LoginPage navigates the session to the **Dashboard**. This Dashboard is the central user interface, initialized with the User's details to load the appropriate Student Portal or Admin Dashboard view, maintaining a logical and secure flow from database access to personalized session management.

System architecture



The Digital Notice and Event Registration System (DNERS) employs a Three-Tier Architecture—Presentation, Application, and Data—to ensure a clear separation of concerns, robust security, and simplified maintenance.

The *Presentation Tier* constitutes the client-facing Graphical User Interface (GUI), which is built using Java components. This tier includes the LoginPage and the various Dashboard views for both students and administrators. Its primary function is to handle user input and display information received from the application logic.

The *Application Tier* (or Business Logic Tier) is the core of the system, written in Java. This tier manages all processing, calculations, and decision-making, such as authenticating user credentials, applying the search/filter logic (keyed by department name), processing event registrations, and generating admin reports. All communication between the front end and the database is channeled securely through this layer, specifically utilizing the DBConnection class to manage connectivity.

Finally, the *Data Tier* is the backend storage repository, typically a relational database like MySQL. This tier is responsible for persistent data storage and secure retrieval. It centrally stores all essential system information, including User Records (containing student/admin roles for access control), Notice Records (the announcements published by administrators), Event Records (all details required for event listings), and Registration Records (linking students to the events they signed up for). This structure ensures data integrity and forms the foundation for all system operations.

IMPLEMENTATION DETAILS

This chapter details the specific tools, languages, and methodologies used to translate the system design into the working Digital Notice and Event Registration System.

- ***Technology Stack***

  Programming Language-Java
  Used for all application logic, backend processing, and database connectivity. Java was chosen for its platform independence and strong object-oriented features.
  User Interface (GUI)-Swing
  Used to develop the native desktop application interface, including the Login Page, Dashboard, and various data viewing panels.
  Database-MySQL
  The relational database management system used for persistent storage of all system data.
  Connectivity-JDBC (Java Database Connectivity)
  The standard Java API used to connect the Java Application Tier to the MySQL Data Tier.
  Development Environment-VS Code
  The Integrated Development Environment (IDE) used for coding, compilation, and debugging.

- ***Module Implementation***

  The implementation process followed a modular approach, focusing on the functionality defined in the System Design. Key implementation aspects include:

  User Authentication and Role-Based Access
  The LoginPage handles secure login. Upon user input, the checkLogin() method queries the User table to validate the credentials. The critical role attribute is retrieved during this process, which determines the subsequent access level:
  -If role is 'Student': The system loads the Student Portal view of the Dashboard.
  -If role is 'Admin': The system loads the Admin Dashboard view.

Notice and Event Publication (Admin)

The administrative module implements the content creation functionality. This interface allows the Admin to input all required fields (e.g., event name, date, department) and insert the data directly into the NoticeRecords or EventRecords tables via JDBC queries. This mechanism ensures that the published content is immediately reflected in the centralized database.

Information Retrieval and Filtering (Student)

The Student Dashboard features the primary display for notices and events. This module implements the Search and Filter logic. SQL queries are dynamically constructed in the Java layer to filter data based on user input, specifically the department name, and the results are then displayed on the student's screen in a structured table or list format.

Event Registration and Acknowledgment

Student interaction is managed by two key functions:

1.Event Registration:

When a student clicks the register button for an event, the system executes an insert query into the RegistrationRecords table, linking the student's ID with the event's ID.

2.Notice Acknowledgment:

When a student acknowledges a notice, an appropriate record or flag is updated in the database, providing the verifiable audit trail required by the objectives.

In short, the implementation successfully leveraged Java, Swing, and JDBC to deliver a modular, data-driven solution that fully translates the design specifications into a functional and secure Digital Notice and Event Registration System.

DATABASE DESIGN

Table User:

```
mysql> select*from user;
+---------+---------------+----------+--------------------+-------------------+---------+-----------------------------+
| user_id | username      | password | full_name          | email             | role    | department                  |
+---------+---------------+----------+--------------------+-------------------+---------+-----------------------------+
|    1120 | amritamurthi  | 36424    | Amrita Murthi      | murti123@gmail.com | admin   | computer science            |
|    1121 | aiswaryam     | 11824    | Aiswarya M         | aishh56@gmail.com  | student | Civil                       |
|    1122 | ananyanair    | 46224    | Ananya Nair        | anna@gmail.com     | admin   | Electronics and Communication |
|    1123 | antreenababu  | 45724    | Antreena Babu      | antreena@gmail.com | student | Computer Science            |
|    1124 | angelinajohnson | 07624  | Angelina Johnson   | angel@gmail.com    | student | Mechanical                  |
|    1125 | angeleenaann  | 06024    | Angeleena Ann Shibu | angeann@gmail.com | student | Electrical                  |
+---------+---------------+----------+--------------------+-------------------+---------+-----------------------------+
```

Table Notices:

```
mysql> select*from notices;
+-----------+----------------------+-----------------------------------------------------------------+-----------+-----------------+
| notice_id | title                | description                                                     | file_name | department      |
+-----------+----------------------+-----------------------------------------------------------------+-----------+-----------------+
|         6 | Volunteer Call       | A volunteer call for the Inter College Hackathon                | volunteer | coumputer science |
|         7 | ElevateX             | A freshers day will be held on 14 august 2025 for mechanical students. | elevatex  | Mechanical      |
|         8 | Inter College Hackathon | An Inter College Hackathon will be conducted on 24th Deacember 2026. | Hackathon | coumputer science |
+-----------+----------------------+-----------------------------------------------------------------+-----------+-----------------+
```

The database consists of two main tables — user and notices — which together manage user information and departmental announcements within the system.

The user table stores details of all users, including administrators and students. It contains fields such as user ID, username, password, full name, email, role, and department. This table helps in identifying users and controlling access based on their roles.

The notices table maintains records of various departmental announcements and events. It includes fields such as notice ID, title, description, file name, and department. Each notice is associated with a specific department and includes event information.

Overall, this database structure supports efficient management of user data and communication of important notices across different departments.

PROGRAM CODE

DBConnection.java:

```
import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DBConnection {

    private static final String URL = "jdbc:mysql://localhost:3306/noticeboard";

    private static final String USER = "root";

    private static final String PASSWORD = "byee";

  public static Connection getConnection() throws SQLException {

        Connection conn=null;

        try {

        Class.forName("com.mysql.cj.jdbc.Driver");
conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/noticeboard",
"root", "byee");

        } catch (ClassNotFoundException e) {

            System.out.println("MySQL JDBC Driver not found.");

            e.printStackTrace();

        }

        return conn;

    }

}
```

AdminPanel.java:

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.*;

import java.io.File;

import java.sql.Connection;

import java.sql.PreparedStatement;

class AdminPanel extends JFrame implements ActionListener {

    JTextField titleField;

    JTextField departmentField;

    JTextArea descArea;

    JButton uploadBtn, postBtn;

    JLabel fileLabel;

    File selectedFile;

    AdminPanel() {

      setTitle("Admin Panel - Post Notice");

        setSize(650, 550);

      setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

      setLocationRelativeTo(null);

      setLayout(new BorderLayout());

      JPanel background = new JPanel() {

                protected void paintComponent(Graphics g) {

                super.paintComponent(g);

                Graphics2D g2d = (Graphics2D) g;

                GradientPaint gp = new GradientPaint(

                      0, 0, new Color(210, 225, 255),

               0, getHeight(), new Color(235, 245, 255)
```

```
                );
                g2d.setPaint(gp);
           g2d.fillRect(0, 0, getWidth(), getHeight());
            }
        };
        background.setLayout(new GridBagLayout());
        JPanel card = new JPanel();
        card.setBackground(Color.WHITE);
        card.setPreferredSize(new Dimension(500, 450));
        card.setLayout(null);
              card.setBorder(BorderFactory.createLineBorder(new Color(220, 220,
220), 1, true))


    JLabel title = new JLabel("Admin Panel",     SwingConstants.LEFT);
        title.setFont(new Font("Segoe UI", Font.BOLD, 22));
        title.setForeground(new Color(0, 51, 153));
        title.setBounds(180, 20, 300, 40);
        card.add(title);
      JLabel noticeLabel = new JLabel("Notice Title");
       noticeLabel.setFont(new Font("Segoe UI", Font.PLAIN, 14));
          noticeLabel.setForeground(new Color(20, 40, 90));
          noticeLabel.setBounds(60, 70, 200, 25);
          card.add(noticeLabel);
          titleField = new JTextField();
          titleField.setBounds(60, 100, 370, 35);
   titleField.setFont(new Font("Segoe UI", Font.PLAIN, 14));
                       titleField.setBorder(BorderFactory.createEmptyBorder(5,
5, 5, 5));
        titleField.setBackground(new Color(245, 247, 250));
        card.add(titleField);
```

```
         JLabel deptLabel = new JLabel("Department");
    deptLabel.setFont(new Font("Segoe UI", Font.PLAIN, 14));
        deptLabel.setForeground(new Color(20, 40, 90));
           deptLabel.setBounds(60, 145, 200, 25);
           card.add(deptLabel);

         departmentField = new JTextField();
         departmentField.setBounds(60, 175, 370, 35);
        departmentField.setFont(new Font("Segoe UI", Font.PLAIN, 14));
                departmentField.setBorder(BorderFactory.createEmptyBorder(5,
5, 5, 5));
        departmentField.setBackground(new Color(245, 247, 250));
         card.add(departmentField);

        JLabel descLabel = new JLabel("Description");
    descLabel.setFont(new Font("Segoe UI", Font.PLAIN, 14));
        descLabel.setForeground(new Color(20, 40, 90));
           descLabel.setBounds(60, 230, 200, 25);
           card.add(descLabel);



        descArea = new JTextArea("Enter notice details here...");
             descArea.setBounds(60, 260, 370, 80);
        descArea.setFont(new Font("Segoe UI", Font.PLAIN, 13));
               descArea.setForeground(Color.GRAY);
        descArea.setBorder(BorderFactory.createEmptyBorder(8, 8, 8, 8));
             descArea.setBackground(new Color(245, 247, 250));
                descArea.setLineWrap(true);
```

```
                descArea.setWrapStyleWord(true);
         descArea.addFocusListener(new FocusAdapter() {
            public void focusGained(FocusEvent e) {
           if (descArea.getText().equals("Enter notice details here...")) {
                    descArea.setText("");
descArea.setForeground(Color.BLACK);
                }
            }
            public void focusLost(FocusEvent e) {
                 if (descArea.getText().isEmpty()) {
                      descArea.setText("Enter notice details here...");
descArea.setForeground(Color.GRAY);
                }
            }
        });
        card.add(descArea);
        uploadBtn = new JButton("Upload File");
        uploadBtn.setBounds(60, 355, 150, 30);
     uploadBtn.setBackground(new Color(70, 115, 255));
        uploadBtn.setForeground(Color.WHITE);
        uploadBtn.setFont(new Font("Segoe UI", Font.BOLD, 13));
        uploadBtn.setFocusPainted(false);
        uploadBtn.setBorder(BorderFactory.createEmptyBorder());
        uploadBtn.addActionListener(this);
        card.add(uploadBtn);

        fileLabel = new JLabel("", SwingConstants.LEFT);
        fileLabel.setBounds(220, 360, 200, 25);
```

```
        fileLabel.setFont(new Font("Segoe UI", Font.PLAIN, 12));
        card.add(fileLabel);

        postBtn = new JButton("Post Notice");
        postBtn.setBounds(60, 400, 370, 40);
        postBtn.setBackground(new Color(0, 80, 255));
        postBtn.setForeground(Color.WHITE);
        postBtn.setFont(new Font("Segoe UI", Font.BOLD, 15));
        postBtn.setFocusPainted(false);
        postBtn.setBorder(BorderFactory.createEmptyBorder());
        postBtn.addActionListener(this);
        card.add(postBtn);

        background.add(card);
        add(background);
        setVisible(true);
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getSource() == uploadBtn) {
            JFileChooser chooser = new JFileChooser();
            int result = chooser.showOpenDialog(this);
            if (result == JFileChooser.APPROVE_OPTION) {
                selectedFile = chooser.getSelectedFile();
                fileLabel.setText(selectedFile.getName());
            }
        } else if (e.getSource() == postBtn) {
            String title = titleField.getText();
            String department = departmentField.getText();
```

```
            String desc = descArea.getText();

            String fileName = (selectedFile != null) ? selectedFile.getName() :
null;



            if (title.isEmpty() || department.isEmpty() || desc.equals("Enter
notice details here...") || desc.isEmpty()) {

                JOptionPane.showMessageDialog(this,  "Please  fill  in  all
fields.", "Warning", JOptionPane.WARNING_MESSAGE);

                return;

            }



            try (Connection conn = DBConnection.getConnection()) {

                if (conn != null) {

                    String  sql  =  "INSERT  INTO  notices  (title,  department,
description, file_name) VALUES (?, ?, ?, ?)";

                    PreparedStatement stmt = conn.prepareStatement(sql);

                    stmt.setString(1, title);

                    stmt.setString(2, department);

                    stmt.setString(3, desc);

                    stmt.setString(4, fileName);

                    stmt.executeUpdate();



                    JOptionPane.showMessageDialog(this,     "Notice     Posted
Successfully!", "Success", JOptionPane.INFORMATION_MESSAGE);



                    titleField.setText("");

                    departmentField.setText("");

                    descArea.setText("Enter notice details here...");

                    descArea.setForeground(Color.GRAY);

                    fileLabel.setText("");

                    selectedFile = null;

                } else {
```

```
                    JOptionPane.showMessageDialog(this,  "Database  connection
failed.", "Error", JOptionPane.ERROR_MESSAGE);

                }

            } catch (Exception ex) {

                ex.printStackTrace();

                JOptionPane.showMessageDialog(this,      "Error:      "      +
ex.getMessage(), "Database Error", JOptionPane.ERROR_MESSAGE);

            }

        }

    }


    public static void main(String[] args) {

        SwingUtilities.invokeLater(AdminPanel::new);

    }

}
```

StudentPanel.java:

```
import javax.swing.*;

import java.awt.*;

import java.awt.event.*;

import java.sql.*;

import java.util.ArrayList;

import java.util.List;


class StudentPanel extends JFrame implements ActionListener {


    static class Event {

        String title;

        String description;

        Event(String title, String description) {
```

```
            this.title = title;
            this.description = description;


    }
    }
    private String username;
    private String fullName;
    private JTextField searchField;
    private JPanel cardsPanel;


    StudentPanel(String username) {
        this.username = username;
        this.fullName = fetchFullName(username);
        setTitle("Student Dashboard");
        setSize(1000, 700);
     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setLayout(new BorderLayout());
        JPanel background = createGradientPanel();
        background.setLayout(new BorderLayout());
        JPanel contentArea = createContentArea();
        background.add(contentArea, BorderLayout.CENTER);
        add(background);
        setVisible(true);
    }
    private String fetchFullName(String username) {
        String name = username;
        try (Connection conn = DBConnection.getConnection()) {
            if (conn != null) {
```

```
                String sql = "SELECT full_name FROM user WHERE username = ?";
                PreparedStatement stmt = conn.prepareStatement(sql);
                stmt.setString(1, username);
                ResultSet rs = stmt.executeQuery();
                if (rs.next()) {
                    name = rs.getString("full_name");
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return name;
    }
    private List<Event> fetchNotices(String department) {
        List<Event> events = new ArrayList<>();
        try (Connection conn = DBConnection.getConnection()) {
            if (conn != null) {
                String sql = "SELECT title, description FROM notices WHERE
department LIKE ?";
                PreparedStatement stmt = conn.prepareStatement(sql);
                stmt.setString(1, "%" + department + "%");
                ResultSet rs = stmt.executeQuery();
                while (rs.next()) {

 events.add(new Event(rs.getString("title"), rs.getString("description")));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
        return events;
    }

        private JPanel createGradientPanel() {
        return new JPanel() {
            protected void paintComponent(Graphics g) {
                super.paintComponent(g);
                Graphics2D g2d = (Graphics2D) g;
                GradientPaint gp = new GradientPaint(
                    0, 0, new Color(240, 248, 255),
                    0, getHeight(), new Color(218, 227, 245)
                );
                g2d.setPaint(gp);
                g2d.fillRect(0, 0, getWidth(), getHeight());
            }
        };
    }
    private JPanel createContentArea() {
        JPanel contentArea = new JPanel(new BorderLayout(10, 10));
        contentArea.setOpaque(false);
        contentArea.setBorder(BorderFactory.createEmptyBorder(30, 30, 30, 30));
        JPanel headerPanel = new JPanel();
        headerPanel.setLayout(new BoxLayout(headerPanel, BoxLayout.Y_AXIS));
        headerPanel.setOpaque(false);

        JLabel welcomeLabel = new JLabel("Welcome, " + fullName + "!");
        welcomeLabel.setFont(new Font("Segoe UI", Font.BOLD, 30));
        welcomeLabel.setForeground(new Color(20, 35, 75));
        welcomeLabel.setAlignmentX(Component.CENTER_ALIGNMENT);
```

```
        JPanel searchPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, 10,
10));
        searchPanel.setOpaque(false);
        searchField = new JTextField("Search by department...", 30);
        searchField.setFont(new Font("Segoe UI", Font.PLAIN, 14));
        searchField.setForeground(Color.GRAY);

        searchField.addFocusListener(new FocusAdapter() {
            public void focusGained(FocusEvent e) {
                if (searchField.getText().equals("Search by department...")) {
                    searchField.setText("");
                    searchField.setForeground(Color.BLACK);
                }
            }
            public void focusLost(FocusEvent e) {
                if (searchField.getText().isEmpty()) {
                    searchField.setText("Search by department...");
                    searchField.setForeground(Color.GRAY);
                }
            }
        });

        JButton searchButton = new JButton("Search Department");
        searchButton.setBackground(new Color(70, 115, 255));
        searchButton.setForeground(Color.WHITE);
        searchButton.setFont(new Font("Segoe UI", Font.BOLD, 13));
        searchButton.setFocusPainted(false);
        searchButton.addActionListener(this);
```

```
        searchPanel.add(searchField);
        searchPanel.add(searchButton);

        headerPanel.add(welcomeLabel);
        headerPanel.add(Box.createVerticalStrut(15));
        headerPanel.add(searchPanel);
        headerPanel.add(Box.createVerticalStrut(20));

        contentArea.add(headerPanel, BorderLayout.NORTH);

        JPanel eventsPanel = new JPanel();
        eventsPanel.setLayout(new BoxLayout(eventsPanel, BoxLayout.Y_AXIS));
        eventsPanel.setOpaque(false);

        JLabel title = new JLabel("Department Notices");
        title.setFont(new Font("Segoe UI", Font.BOLD, 26));
        title.setForeground(new Color(20, 35, 75));
        title.setAlignmentX(Component.CENTER_ALIGNMENT);

        eventsPanel.add(title);
        eventsPanel.add(Box.createVerticalStrut(20));

        cardsPanel = new JPanel(new FlowLayout(FlowLayout.CENTER, 20, 20));
        cardsPanel.setOpaque(false);

        JScrollPane scrollPane = new JScrollPane(cardsPanel);

scrollPane.setHorizontalScrollBarPolicy(JScrollPane.HORIZONTAL_SCROLLBAR_AS_NE
EDED);

scrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALWAYS);
```

```
        scrollPane.setBorder(BorderFactory.createEmptyBorder());
        scrollPane.setOpaque(false);
        scrollPane.getViewport().setOpaque(false);

        eventsPanel.add(scrollPane);

        contentArea.add(eventsPanel, BorderLayout.CENTER);

        return contentArea;
    }

    private JPanel createNoticeCard(Event event) {
        JPanel card = new JPanel();
        card.setLayout(new BoxLayout(card, BoxLayout.Y_AXIS));
        card.setPreferredSize(new Dimension(220, 160));
        card.setBackground(new Color(50, 80, 200));
        card.setBorder(BorderFactory.createEmptyBorder(15, 15, 15, 15));

        JLabel titleLabel = new JLabel(event.title);
        titleLabel.setFont(new Font("Segoe UI", Font.BOLD, 15));
        titleLabel.setForeground(Color.WHITE);
        titleLabel.setAlignmentX(Component.LEFT_ALIGNMENT);

        JButton viewButton = new JButton("View Details");
        viewButton.setFont(new Font("Segoe UI", Font.BOLD, 12));
        viewButton.setBackground(new Color(150, 180, 255));
        viewButton.setForeground(Color.BLACK);
        viewButton.setFocusPainted(false);
        viewButton.setCursor(new Cursor(Cursor.HAND_CURSOR));
```

```
        viewButton.addActionListener(ae -> {
            JOptionPane.showMessageDialog(this,
                event.description,
                "Details for: " + event.title,
                JOptionPane.INFORMATION_MESSAGE);
        });

        card.add(titleLabel);
        card.add(Box.createVerticalGlue());
        card.add(viewButton);
        return card;
    }

    private void displayEvents(List<Event> events) {
        cardsPanel.removeAll();
        if (events.isEmpty()) {
            JLabel noData = new JLabel("No notices found for this department.",
SwingConstants.CENTER);
            noData.setFont(new Font("Segoe UI", Font.ITALIC, 16));
            noData.setForeground(Color.GRAY);
            cardsPanel.add(noData);
        } else {
            for (Event e : events) {
                cardsPanel.add(createNoticeCard(e));
            }
        }
        cardsPanel.revalidate();
        cardsPanel.repaint();
    }
```

```
    @Override
    public void actionPerformed(ActionEvent e) {
        String department = searchField.getText().trim();
        if (department.isEmpty() || department.equals("Search by
department...")) {
            JOptionPane.showMessageDialog(this, "Please enter a department name
to search.", "Warning", JOptionPane.WARNING_MESSAGE);
            return;
        }
        List<Event> events = fetchNotices(department);
        displayEvents(events);
    }
        public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new StudentPanel("student"));
    }
```

LoginPage.java:

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

public class LoginPage extends JFrame implements ActionListener {
    JTextField userField;
    JPasswordField passField;
    JComboBox<String> userTypeBox;
    JButton loginButton;
    JLabel msgLabel;
```

```
LoginPage() {
    setTitle("Digital Notice Board & Event Management System");
    setSize(600, 450);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    setLocationRelativeTo(null);
    setLayout(new BorderLayout());



    JPanel backgroundPanel = new JPanel() {
        protected void paintComponent(Graphics g) {
            super.paintComponent(g);
            Graphics2D g2d = (Graphics2D) g;
            GradientPaint gp = new GradientPaint(
                0, 0, new Color(185, 213, 255),
                0, getHeight(), new Color(240, 248, 255)
            );
            g2d.setPaint(gp);
            g2d.fillRect(0, 0, getWidth(), getHeight());
        }
    };
    backgroundPanel.setLayout(new GridBagLayout());



    JPanel card = new JPanel();
    card.setBackground(Color.WHITE);
    card.setPreferredSize(new Dimension(500, 400));
    card.setLayout(null);
```

```
        card.setBorder(BorderFactory.createLineBorder(new Color(230, 230, 230),
1, true));



        JLabel title = new JLabel("<html><center>Digital Notice Board &
Event<br>Management System</center></html>", SwingConstants.CENTER);

        title.setFont(new Font("Segoe UI", Font.BOLD, 18));

        title.setForeground(new Color(0, 51, 153));

        title.setBounds(90, 20, 320, 50);

        card.add(title);


        JLabel subtitle = new JLabel("Login to your account",
SwingConstants.CENTER);

        subtitle.setFont(new Font("Segoe UI", Font.PLAIN, 13));

        subtitle.setForeground(new Color(100, 100, 100));

        subtitle.setBounds(50, 70, 400, 25);

        card.add(subtitle);



        userField = new JTextField();

        userField.setBounds(120, 110, 250, 40);

        userField.setBorder(BorderFactory.createTitledBorder("Username"));

        card.add(userField);


        passField = new JPasswordField();

        passField.setBounds(120, 160, 250, 40);

        passField.setBorder(BorderFactory.createTitledBorder("Password"));

        card.add(passField);


        JPanel comboPanel = new JPanel(new BorderLayout());

        comboPanel.setBounds(120, 210, 250, 55);
```

```
        comboPanel.setBackground(Color.WHITE);
        comboPanel.setBorder(BorderFactory.createTitledBorder("User type"));

        userTypeBox = new JComboBox<>(new String[]{"Student", "Admin"});
        userTypeBox.setFont(new Font("Segoe UI", Font.PLAIN, 13));
        userTypeBox.setFocusable(false);
        userTypeBox.setBackground(Color.WHITE);
        userTypeBox.setBorder(BorderFactory.createEmptyBorder(5, 5, 5, 5));
        comboPanel.add(userTypeBox, BorderLayout.CENTER);
        card.add(comboPanel);



        loginButton = new JButton("Login");
        loginButton.setBounds(120, 270, 250, 35);
        loginButton.setBackground(new Color(0, 102, 255));
        loginButton.setForeground(Color.WHITE);
        loginButton.setFocusPainted(false);
        loginButton.setFont(new Font("Segoe UI", Font.BOLD, 14));
        loginButton.setBorder(BorderFactory.createEmptyBorder());
        loginButton.addActionListener(this);
        card.add(loginButton);



        msgLabel = new JLabel("", SwingConstants.CENTER);
        msgLabel.setForeground(Color.BLUE);
        msgLabel.setBounds(100, 300, 300, 25);
        card.add(msgLabel);

        backgroundPanel.add(card);
```

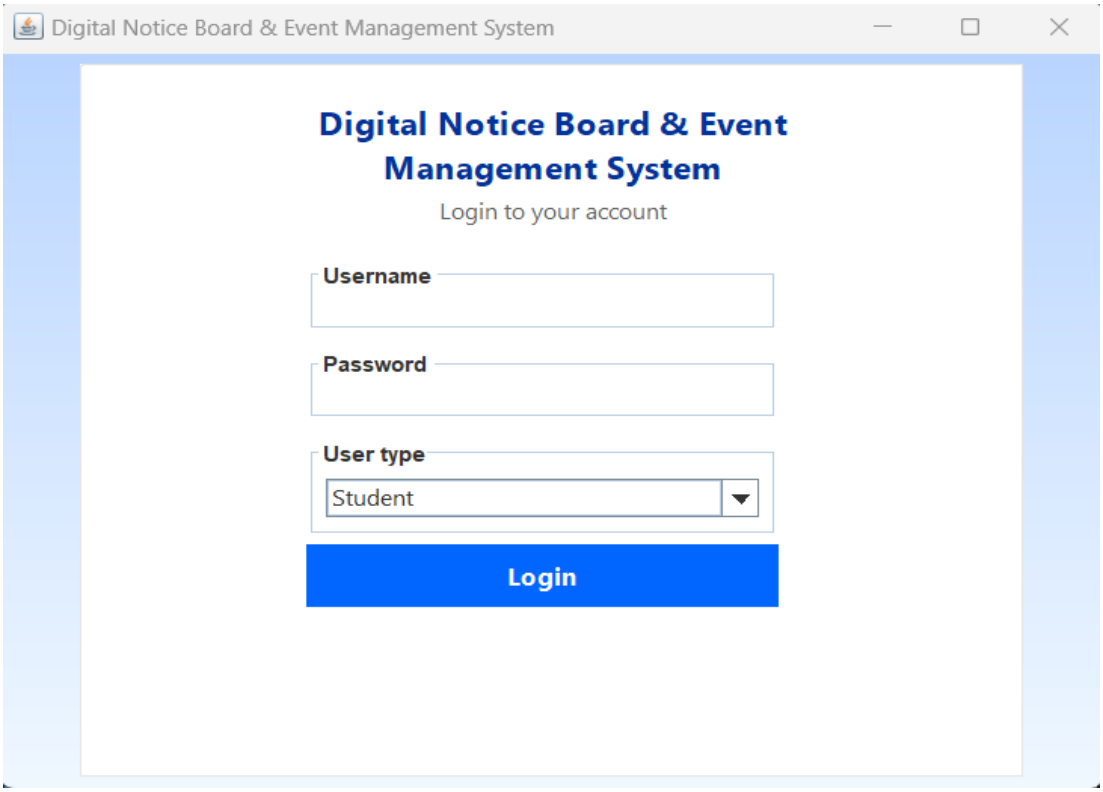
```
        add(backgroundPanel);

        setVisible(true);

    }

    private boolean checkLogin(String username, String password, String type) {

        String query = "SELECT * FROM user WHERE username = ? AND password = ?
AND role = ?";


        try (Connection conn = DBConnection.getConnection();

             PreparedStatement stmt = conn.prepareStatement(query)) {


            stmt.setString(1, username);

            stmt.setString(2, password);

            stmt.setString(3, type.toLowerCase()); //


            ResultSet rs = stmt.executeQuery();

            return rs.next();


        } catch (SQLException e) {

            e.printStackTrace();

            msgLabel.setText("Database connection error!");

            msgLabel.setForeground(Color.RED);

        }


        return false;

    }

    public void actionPerformed(ActionEvent e) {

        String user = userField.getText().trim();

        String pass = new String(passField.getPassword());

        String type = (String) userTypeBox.getSelectedItem();
```

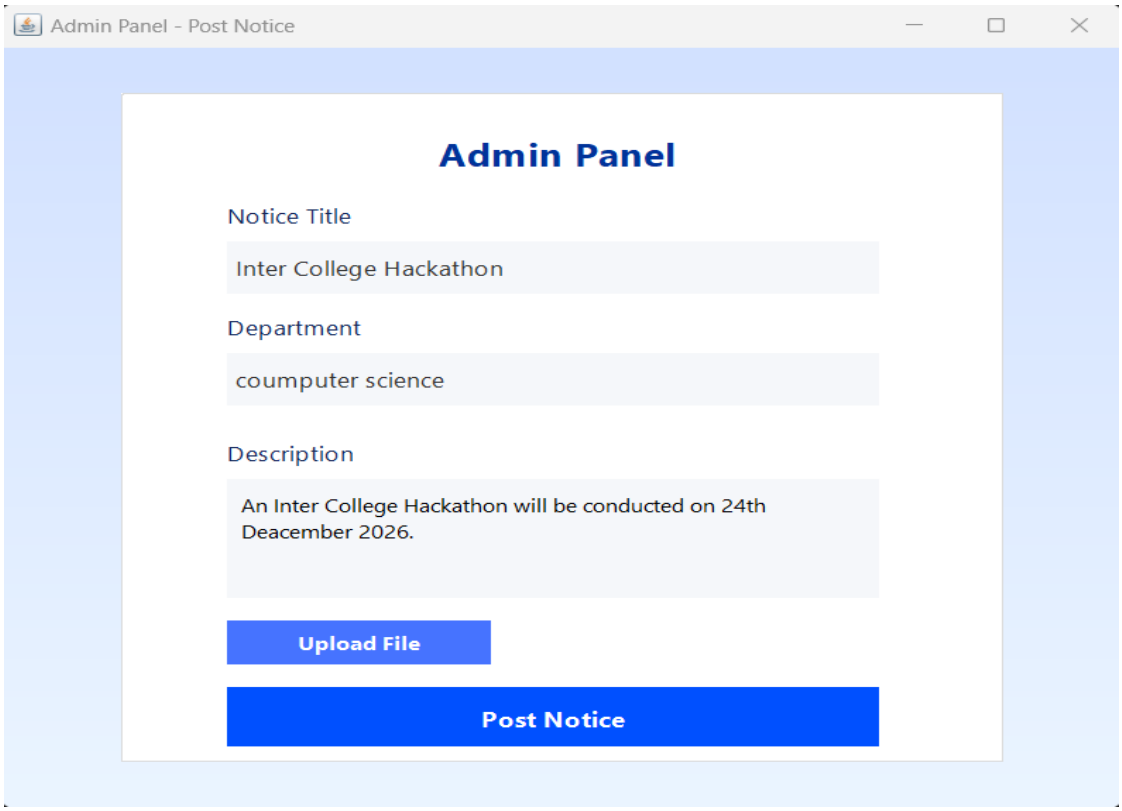
```
        if (checkLogin(user, pass, type)) {
            msgLabel.setText("Login Successful as " + type + "!");
            msgLabel.setForeground(Color.BLUE);

            if (type.equals("Admin")) {
                new AdminPanel();
                dispose();
            } else if (type.equals("Student")) {
                new StudentPanel(user);
                dispose();
            }

        } else {
            msgLabel.setText("Invalid username or password.");
            msgLabel.setForeground(Color.RED);
        }
    }
    public static void main(String[] args) {
        SwingUtilities.invokeLater(LoginPage::new);
    }
}
```

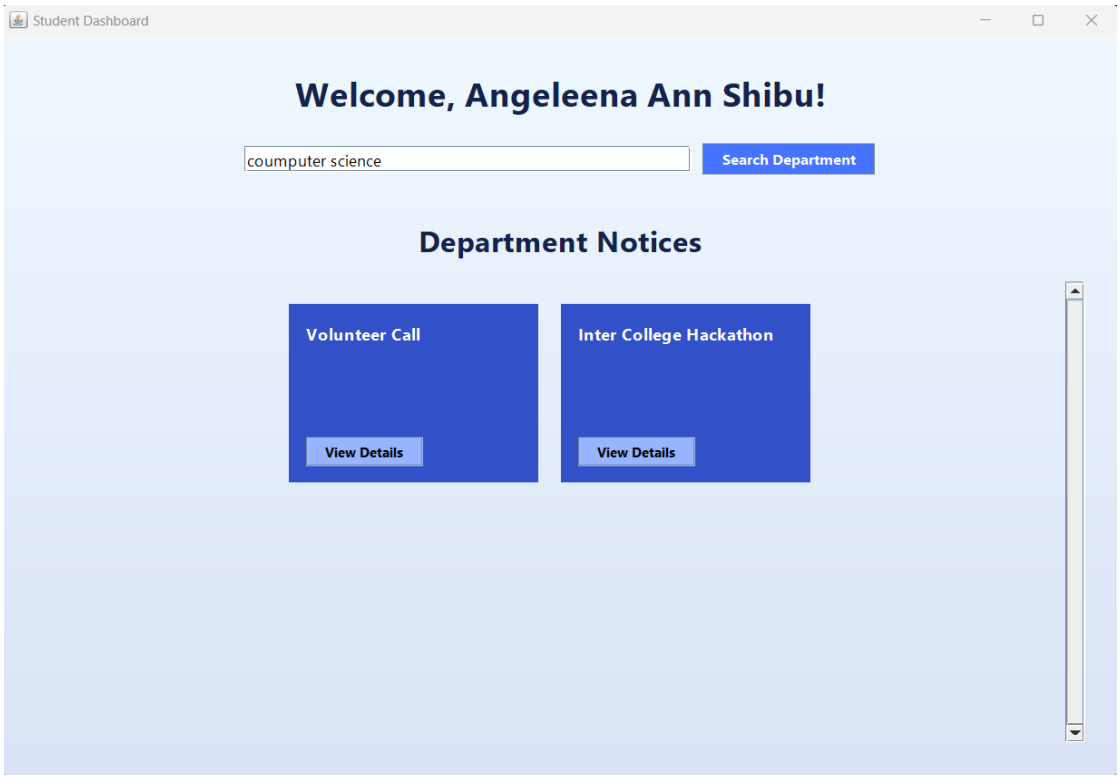
Main.java:

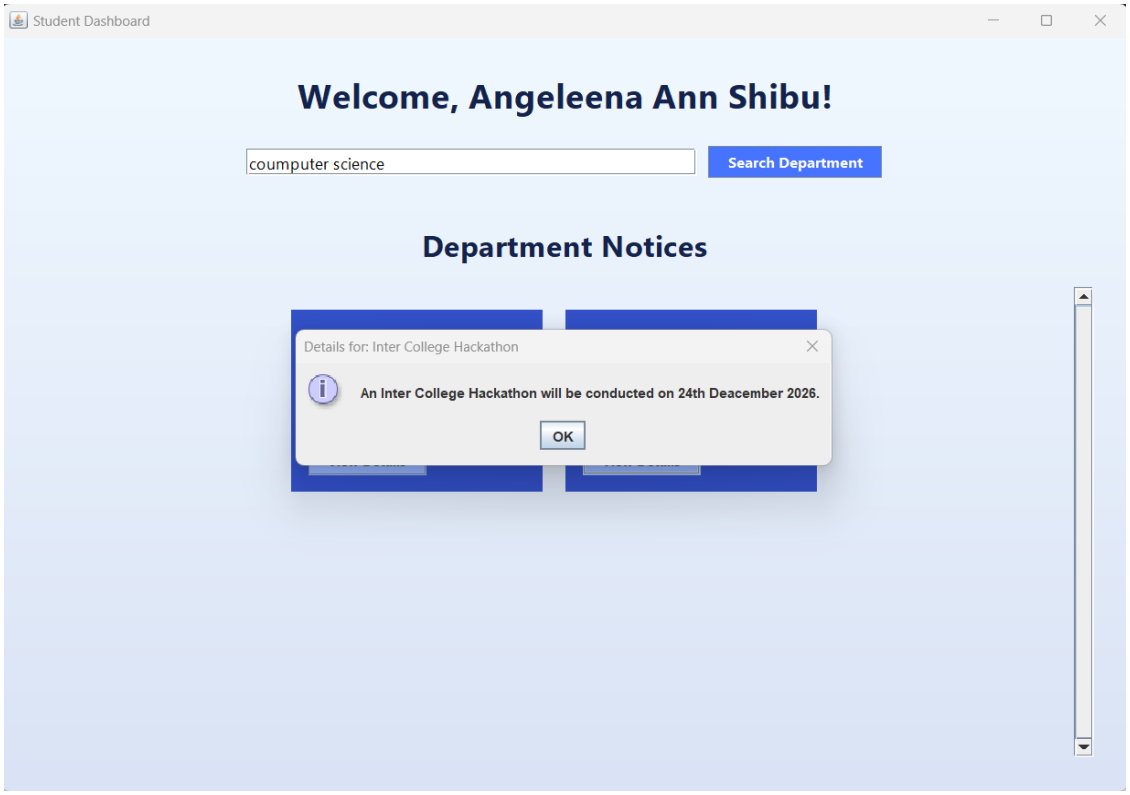
```
public class Main {
    public static void main(String[] args)
    {
        new LoginPage();
    }
}
```

PROJECT INTERFACES






Student Dashboard
Welcome, Angeleena Ann Shibu!
coumputer science
Search Department
Department Notices
Volunteer Call
View Details
Inter College Hackathon
View Details


Student Dashboard
Welcome, Angeleena Ann Shibu!
coumputer science
Search Department
Department Notices
Details for: Inter College Hackathon
An Inter College Hackathon will be conducted on 24th Deacember 2026.
OK

CONCLUSION

The Digital Notice and Event Registration System project successfully achieved its core objective: to replace fragmented, inefficient campus communication methods with a centralized, reliable, and secure digital platform. By migrating information from physical notice boards and scattered messaging groups to a unified Java-based system, the Digital Notice and Event Registration System has successfully established a single point of truth for all institutional notices and event details. The implementation delivered on all defined functional goals, including secure Role-Based Access Control, streamlined event registration, and critical information retrieval capabilities, notably the department-keyed search functionality. The development of distinct Student and Admin portals ensures enhanced administrative efficiency, guaranteed information visibility for students, and the creation of a verifiable, audit-ready record for acknowledgments and participation history.

In closing, the Digital Notice and Event Registration System is a significant and necessary technological upgrade that fundamentally transforms institutional communication from a reactive burden into a proactive, organized, and student-centric digital process.
