**Updated Use Case Description:**

BY:HALEEMA SADIA

**UC: Register Project**

**Primary Actor:**

* Supervisor

**Pre-conditions:**

* The supervisor must be logged into the system.
* The project registration period must be open.

**Post-conditions:**

* The project is registered in the system.
* The project is now available for student selection.

**Input:**

* Project Title
* Project Description
* Programming Languages
* Academic Requirements
* Minimum Required CGPA
* Required Credit Hours

**Output:**

* Project stored in the database.
* Confirmation message displayed to the supervisor.

**Main Success Scenario:**

1. The supervisor logs into the system and navigates to the "Register Project" section.
2. The system displays the project registration form.
3. The supervisor enters project details (title, description, required skills, CGPA).
4. The supervisor submits the project registration form.
5. The system validates the data and checks for duplicate project titles and invalid CGPA.
6. The system stores the project in the database.
7. The system confirms successful registration.

**Alternative Scenarios:**

**Missing Fields:**

* If the supervisor does not complete all required fields, the system prompts them to enter missing details.

**Duplicate Project Name:**

* If the project title already exists, the system asks the supervisor to provide a different name.

**System Failure:**

* If the system crashes, project details are saved as a draft for later submission.

**UC: Approve Project**

**Primary Actor:**

* Supervisor

**Pre-conditions:**

* The supervisor must be logged into the system.
* The project request must be in a "Pending Approval" state (submitted by a student).

**Post-conditions:**

* The project is either approved or rejected
* The student receives a notification of the decision.

**Input:**

* Project details
* Student justification
* Supervisor’s decision (Approve, Reject)
* Supervisor comments

**Output:**

* Updated project status
* Student receives approval/rejection notification

**Main Success Scenario:**

* The supervisor logs into the system and navigates to the "Pending Project Approvals" section.
* The system displays a list of registered projects.
* The supervisor selects a project to review.
* The system displays the student’s details, selected project, and justification.
* The supervisor evaluates the student’s capability and project suitability.
* The supervisor selects one of the following options:

**Approve:** If the student is suitable for the project.

**Reject:** If the project is not suitable for the student.

* The system updates the project status accordingly.
* The system notifies the student of the supervisor’s decision.

**Alternative Scenarios:**

**Student Not Qualified:**

* If the student lacks the required skills, the supervisor rejects the request with feedback.

**Project Already Assigned:**

* If another student has already been approved for the same project, the supervisor must reassign the student.

**System Failure:** If the system crashes, the supervisor’s progress is saved.

**UC: Receive Feedback**

**Primary Actor:**

* Student

**Pre-Conditions:**

* The student is logged into the system.
* The student has already submitted work product for review. Word product may have many versions.
* The supervisor has provided feedback on the submitted work.

**Post-Conditions:**

* The student views the feedback provided by the supervisor.
* The feedback is stored in the system for future reference.

**Input:**

* None

**Output:**

* Displayed feedback text or attached review file.

**Main Success Scenario:**

1. The student logs into the system.
2. The student select the version of work product
3. The student navigates to the "View Feedback"section.
4. The system retrieves the feedback stored for the student.
5. The system displays the feedback provided by the supervisor
6. The student reviews the feedback and can use it for project improvements.

**Alternative Scenario:**

**No Feedback Available**

* If the supervisor has not provided feedback, then system displays: *“*Please check back later.*”*

**System Failure While Loading Feedback**

* If the feedback fails to load, the student receives an error message: “An error occurred. Please try again later.”

**Sequence Diagram:**

By: Haleema Sadia

**UC: Register Project:**

**UC: Approve Project**



**UC: Receive Feedback**



import javax.swing.\*;

import javax.swing.border.EmptyBorder;

import javax.swing.table.DefaultTableModel;

import java.awt.\*;

import java.util.ArrayList;

import java.util.List;

public class ApproveProjectScreen extends JFrame {

    static class Project {

        String title;

        String description;

        String studentJustification;

        String studentRegNo;

        Status status;

        enum Status { PENDING, APPROVED, REJECTED }

        Project(String title, String desc, String justification, String regNo, Status status) {

            this.title = title;

            this.description = desc;

            this.studentJustification = justification;

            this.studentRegNo = regNo;

            this.status = status;

        }

    }

    private List<Project> projects = new ArrayList<>();

    private JTable pendingTable, approvedTable;

    private DefaultTableModel pendingModel, approvedModel;

    private JTextField titleField, regNoField;

    private JTextArea descriptionArea, studentJustificationArea, supervisorCommentsArea;

    private JRadioButton approveRadio, rejectRadio;

    private JButton submitBtn, cancelBtn;

    private Project selectedProject;

    public ApproveProjectScreen() {

        setTitle("Approve Projects");

        setSize(1000, 600);

        setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

        setLocationRelativeTo(null);

        applyDarkTheme();

        loadSampleProjects();

        JTabbedPane tabbedPane = new JTabbedPane();

        JPanel pendingPanel = createPendingProjectsPanel();

        tabbedPane.addTab("Pending Projects", pendingPanel);

        JPanel approvedPanel = createApprovedProjectsPanel();

        tabbedPane.addTab("Approved Projects", approvedPanel);

        add(tabbedPane, BorderLayout.CENTER);

        add(createDetailsPanel(), BorderLayout.EAST);

        setVisible(true);

    }

    private void applyDarkTheme() {

        UIManager.put("control", new Color(40, 40, 40));

        UIManager.put("info", new Color(40, 40, 40));

        UIManager.put("nimbusBase", new Color(18, 30, 49));

        UIManager.put("nimbusAlertYellow", new Color(248, 187, 0));

        UIManager.put("nimbusDisabledText", new Color(128, 128, 128));

        UIManager.put("nimbusFocus", new Color(115, 164, 209));

        UIManager.put("nimbusGreen", new Color(176, 179, 50));

        UIManager.put("nimbusInfoBlue", new Color(66, 139, 221));

        UIManager.put("nimbusLightBackground", new Color(30, 30, 30));

        UIManager.put("nimbusOrange", new Color(191, 98, 4));

        UIManager.put("nimbusRed", new Color(169, 46, 34));

        UIManager.put("nimbusSelectedText", Color.WHITE);

        UIManager.put("nimbusSelectionBackground", new Color(104, 93, 156));

        UIManager.put("text", Color.WHITE);

        try {

            for (UIManager.LookAndFeelInfo info : UIManager.getInstalledLookAndFeels()) {

                if ("Nimbus".equals(info.getName())) {

                    UIManager.setLookAndFeel(info.getClassName());

                    SwingUtilities.updateComponentTreeUI(this);

                    break;

                }

            }

        } catch (Exception e) {

            System.err.println("Failed to apply dark theme.");

        }

    }

    private void loadSampleProjects() {

        projects.add(new Project("Smart Campus App", "Mobile app for campus navigation",

                "I want to work on mobile apps.", "04072312010", Project.Status.PENDING));

        projects.add(new Project("AI Chatbot", "Chatbot for university help desk",

                "Interested in AI projects.", "04072312011", Project.Status.PENDING));

        projects.add(new Project("Website Revamp", "Revamping university website",

                "I have web development skills.", "04072312012", Project.Status.APPROVED));

    }

    private JPanel createPendingProjectsPanel() {

        JPanel panel = new JPanel(new BorderLayout());

        panel.setBackground(new Color(40, 40, 40));

        String[] columns = {"Project Title", "Student Reg No", "Status"};

        pendingModel = new DefaultTableModel(columns, 0) {

            public boolean isCellEditable(int row, int column) { return false; }

        };

        pendingTable = new JTable(pendingModel);

        pendingTable.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);

        pendingTable.setBackground(Color.DARK\_GRAY);

        pendingTable.setForeground(Color.WHITE);

        pendingTable.setGridColor(Color.GRAY);

        loadPendingProjects();

        pendingTable.getSelectionModel().addListSelectionListener(e -> {

            if (!e.getValueIsAdjusting()) {

                int row = pendingTable.getSelectedRow();

                if (row >= 0) {

                    String title = (String) pendingModel.getValueAt(row, 0);

                    String regNo = (String) pendingModel.getValueAt(row, 1);

                    selectedProject = findProjectByTitleAndRegNo(title, regNo, Project.Status.PENDING);

                    updateProjectDetails(selectedProject);

                }

            }

        });

        panel.add(new JScrollPane(pendingTable), BorderLayout.CENTER);

        return panel;

    }

    private JPanel createApprovedProjectsPanel() {

        JPanel panel = new JPanel(new BorderLayout());

        panel.setBackground(new Color(40, 40, 40));

        String[] columns = {"Project Title", "Student Reg No", "Status"};

        approvedModel = new DefaultTableModel(columns, 0) {

            public boolean isCellEditable(int row, int column) { return false; }

        };

        approvedTable = new JTable(approvedModel);

        approvedTable.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);

        approvedTable.setBackground(Color.DARK\_GRAY);

        approvedTable.setForeground(Color.WHITE);

        approvedTable.setGridColor(Color.GRAY);

        loadApprovedProjects();

        approvedTable.getSelectionModel().addListSelectionListener(e -> {

            if (!e.getValueIsAdjusting()) {

                int row = approvedTable.getSelectedRow();

                if (row >= 0) {

                    String title = (String) approvedModel.getValueAt(row, 0);

                    String regNo = (String) approvedModel.getValueAt(row, 1);

                    selectedProject = findProjectByTitleAndRegNo(title, regNo, Project.Status.APPROVED);

                    updateProjectDetails(selectedProject);

                }

            }

        });

        panel.add(new JScrollPane(approvedTable), BorderLayout.CENTER);

        return panel;

    }

    private void loadPendingProjects() {

        pendingModel.setRowCount(0);

        for (Project p : projects) {

            if (p.status == Project.Status.PENDING) {

                pendingModel.addRow(new Object[]{p.title, p.studentRegNo, p.status.name()});

            }

        }

    }

    private void loadApprovedProjects() {

        approvedModel.setRowCount(0);

        for (Project p : projects) {

            if (p.status == Project.Status.APPROVED) {

                approvedModel.addRow(new Object[]{p.title, p.studentRegNo, p.status.name()});

            }

        }

    }

    private Project findProjectByTitleAndRegNo(String title, String regNo, Project.Status status) {

        for (Project p : projects) {

            if (p.title.equals(title) && p.studentRegNo.equals(regNo) && p.status == status) {

                return p;

            }

        }

        return null;

    }

    private JPanel createDetailsPanel() {

        JPanel panel = new JPanel(new BorderLayout());

        panel.setPreferredSize(new Dimension(400, 0));

        panel.setBorder(new EmptyBorder(10, 10, 10, 10));

        panel.setBackground(new Color(40, 40, 40));

        JPanel fieldsPanel = new JPanel(new GridBagLayout());

        fieldsPanel.setBackground(new Color(40, 40, 40));

        GridBagConstraints gbc = new GridBagConstraints();

        gbc.insets = new Insets(5,5,5,5);

        gbc.anchor = GridBagConstraints.WEST;

        gbc.fill = GridBagConstraints.HORIZONTAL;

        int row = 0;

        gbc.gridx = 0; gbc.gridy = row; fieldsPanel.add(createLabel("Project Title:"), gbc);

        titleField = new JTextField(); titleField.setEditable(false); titleField.setBackground(Color.DARK\_GRAY); titleField.setForeground(Color.WHITE);

        gbc.gridx = 1; fieldsPanel.add(titleField, gbc); row++;

        gbc.gridx = 0; gbc.gridy = row; fieldsPanel.add(createLabel("Student Reg No:"), gbc);

        regNoField = new JTextField(); regNoField.setEditable(false); regNoField.setBackground(Color.DARK\_GRAY); regNoField.setForeground(Color.WHITE);

        gbc.gridx = 1; fieldsPanel.add(regNoField, gbc); row++;

        gbc.gridx = 0; gbc.gridy = row; fieldsPanel.add(createLabel("Description:"), gbc);

        descriptionArea = new JTextArea(3, 20); descriptionArea.setEditable(false); descriptionArea.setBackground(Color.DARK\_GRAY); descriptionArea.setForeground(Color.WHITE);

        gbc.gridx = 1; fieldsPanel.add(new JScrollPane(descriptionArea), gbc); row++;

        gbc.gridx = 0; gbc.gridy = row; fieldsPanel.add(createLabel("Student Justification:"), gbc);

        studentJustificationArea = new JTextArea(3, 20); studentJustificationArea.setEditable(false); studentJustificationArea.setBackground(Color.DARK\_GRAY); studentJustificationArea.setForeground(Color.WHITE);

        gbc.gridx = 1; fieldsPanel.add(new JScrollPane(studentJustificationArea), gbc);

        panel.add(fieldsPanel, BorderLayout.NORTH);

        JPanel decisionPanel = new JPanel(new GridBagLayout());

        decisionPanel.setBorder(BorderFactory.createTitledBorder("Supervisor Decision"));

        decisionPanel.setBackground(new Color(40, 40, 40));

        decisionPanel.setForeground(Color.WHITE);

        ButtonGroup decisionGroup = new ButtonGroup();

        approveRadio = new JRadioButton("Approve"); approveRadio.setBackground(new Color(40, 40, 40)); approveRadio.setForeground(Color.WHITE);

        rejectRadio = new JRadioButton("Reject"); rejectRadio.setBackground(new Color(40, 40, 40)); rejectRadio.setForeground(Color.WHITE);

        decisionGroup.add(approveRadio);

        decisionGroup.add(rejectRadio);

        gbc.gridx = 0; gbc.gridy = 0; decisionPanel.add(approveRadio, gbc);

        gbc.gridx = 1; decisionPanel.add(rejectRadio, gbc);

        gbc.gridx = 0; gbc.gridy = 1; gbc.gridwidth = 2;

        decisionPanel.add(createLabel("Supervisor Comments (required):"), gbc);

        supervisorCommentsArea = new JTextArea(4, 30); supervisorCommentsArea.setBackground(Color.DARK\_GRAY); supervisorCommentsArea.setForeground(Color.WHITE);

        JScrollPane commentsScroll = new JScrollPane(supervisorCommentsArea);

        gbc.gridy = 2;

        decisionPanel.add(commentsScroll, gbc);

        JPanel buttonsPanel = new JPanel();

        buttonsPanel.setBackground(new Color(40, 40, 40));

        submitBtn = new JButton("Submit");

        cancelBtn = new JButton("Cancel");

        submitBtn.addActionListener(e -> handleSubmit());

        cancelBtn.addActionListener(e -> handleCancel());

        buttonsPanel.add(submitBtn);

        buttonsPanel.add(cancelBtn);

        JPanel bottomPanel = new JPanel(new BorderLayout());

        bottomPanel.setBackground(new Color(40, 40, 40));

        bottomPanel.add(decisionPanel, BorderLayout.CENTER);

        bottomPanel.add(buttonsPanel, BorderLayout.SOUTH);

        panel.add(bottomPanel, BorderLayout.SOUTH);

        return panel;

    }

    private JLabel createLabel(String text) {

        JLabel label = new JLabel(text);

        label.setForeground(Color.WHITE);

        return label;

    }

    private void updateProjectDetails(Project project) {

        if (project == null) {

            clearDetails();

            return;

        }

        titleField.setText(project.title);

        regNoField.setText(project.studentRegNo);

        descriptionArea.setText(project.description);

        studentJustificationArea.setText(project.studentJustification);

        supervisorCommentsArea.setText("");

        approveRadio.setSelected(false);

        rejectRadio.setSelected(false);

    }

    private void clearDetails() {

        titleField.setText("");

        regNoField.setText("");

        descriptionArea.setText("");

        studentJustificationArea.setText("");

        supervisorCommentsArea.setText("");

        approveRadio.setSelected(false);

        rejectRadio.setSelected(false);

    }

    private void handleSubmit() {

        if (selectedProject == null) {

            JOptionPane.showMessageDialog(this, "Please select a project first.");

            return;

        }

        if (!approveRadio.isSelected() && !rejectRadio.isSelected()) {

            JOptionPane.showMessageDialog(this, "Please select Approve or Reject.");

            return;

        }

        String comments = supervisorCommentsArea.getText().trim();

        if (comments.isEmpty()) {

            JOptionPane.showMessageDialog(this, "Supervisor comments are required.");

            return;

        }

        selectedProject.status = approveRadio.isSelected() ? Project.Status.APPROVED : Project.Status.REJECTED;

        loadPendingProjects();

        loadApprovedProjects();

        clearDetails();

        JOptionPane.showMessageDialog(this, "Decision submitted successfully.");

    }

    private void handleCancel() {

        clearDetails();

        pendingTable.clearSelection();

        approvedTable.clearSelection();

    }

    public static void main(String[] args) {

        SwingUtilities.invokeLater(ApproveProjectScreen::new);

    }

}