****

**Lab\_Assignment\_4 (Code)**

|  |  |
| --- | --- |
| **Name :** | **Roll No** |
| **JAVAIRIA LATEEF** | **SP24-BSE-050** |
| **MARYAM ABID** | **SP24-BSE-058** |
| **AREEJ ABRAR** | **SP24-BSE-141** |

**UNIVERSITY MANAGEMENT STYSETEM**

**package com.example.demo;  
  
import javafx.application.Application;  
import javafx.scene.Scene;  
import javafx.scene.layout.BorderPane;  
import javafx.scene.layout.StackPane;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class MainApp extends Application {** private StackPane root;  
 private Scene scene;  
  
 @Override  
 public void start(Stage primaryStage) throws IOException {  
 root = new StackPane();  
 MainLayout mainLayout = new MainLayout(this); // Pass MainApp instance  
 root.getChildren().add(mainLayout.getPane());  
  
 scene = new Scene(root, 800, 600);  
 primaryStage.setScene(scene);  
 primaryStage.setTitle("University System");  
 primaryStage.show();  
 }  
  
 public void switchLayout(BorderPane newLayout) {  
 root.getChildren().clear();  
 root.getChildren().add(newLayout);  
 }  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
**}**

**package com.example.demo;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.\*;  
import javafx.scene.shape.Circle;  
import javafx.scene.shape.Rectangle;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontPosture;  
import javafx.scene.text.FontWeight;  
  
public class MainLayout {  
 private BorderPane pane;  
 private MainApp app; // Reference to MainApp  
  
 // Pass MainApp to MainLayout  
 public MainLayout(MainApp app) {  
 this.app = app; // Assign MainApp instance  
 pane = new BorderPane();  
 pane.setStyle("-fx-background-color:#C8A97D");  
  
 VBox centrepart = main();  
 pane.setCenter(centrepart);  
 }  
  
 private VBox main() {  
 Image sideimage=new Image(this.getClass().getResource("/oop images.jpg").toExternalForm());  
 ImageView imageView = new ImageView(sideimage);  
 imageView.setFitWidth(600);  
 imageView.setFitHeight(750);  
  
 // Create curves for the image  
 Rectangle clip = new Rectangle(600, 750);  
 clip.setArcHeight(30);  
 clip.setArcWidth(30);  
  
 imageView.setClip(clip);  
 VBox imagebox = new VBox();  
 imagebox.getChildren().addAll(imageView);  
 imagebox.setPadding(new Insets(20, 0, 0, 20));  
 pane.setLeft(imagebox);  
  
 // Create form components  
 Label type = new Label("Type");  
 type.setFont(Font.*font*("Arial", FontWeight.*BOLD*, FontPosture.*ITALIC*, 15));  
 type.setPadding(new Insets(5, 0, 0, 0));  
  
 ComboBox<String> comboBox = new ComboBox<>();  
 comboBox.getItems().addAll("Student", "Faculty", "Admin");  
  
 HBox typebox = new HBox();  
 typebox.getChildren().addAll(type, comboBox);  
 typebox.setSpacing(10);  
  
 Label Name = new Label("USERNAME");  
 Name.setFont(Font.*font*("Arial", FontWeight.*BOLD*, FontPosture.*ITALIC*, 15));  
 TextField textField = new TextField();  
  
  
 Label Pasword = new Label("PASSWORD");  
 Pasword.setFont(Font.*font*("Arial", FontWeight.*BOLD*, FontPosture.*ITALIC*, 15));  
 PasswordField passwordField = new PasswordField();  
  
 textField.setPrefHeight(25); // Fixed width for the TextField  
 passwordField.setPrefHeight(25);  
  
 Button login = new Button("LOGIN");  
 login.setPadding(new Insets(10, 10, 10, 10));  
 login.setFont(Font.*font*("Arial", FontPosture.*ITALIC*, 15));  
  
 Button register = new Button("REGISTER");  
 register.setFont(Font.*font*("Arial", FontPosture.*ITALIC*, 15));  
 register.setPadding(new Insets(10, 10, 10, 10));  
  
 HBox button = new HBox();  
 button.getChildren().addAll(login, register);  
 button.setSpacing(20);  
  
 // Set the Login button's action  
 login.setOnAction(e -> {  
 String selectedType = comboBox.getValue();  
 if ("Admin".equals(selectedType)) {  
 AdminDashboard adminPortal=new AdminDashboard();  
 if (app != null) {  
 app.switchLayout(adminPortal.getPane());  
 } else {  
 System.*err*.println("MainApp reference is null");  
 }  
  
 }  
 if ("Faculty".equalsIgnoreCase(selectedType)) {  
 FacultyDashboard FacultyPortal=new FacultyDashboard();  
 if (app != null) {  
 app.switchLayout(FacultyPortal.getPane());  
 } else {  
 System.*err*.println("MainApp reference is null");  
 }  
  
 }  
 if ("Student".equalsIgnoreCase(selectedType)) {  
 StudentPortal studentportal=new StudentPortal();  
 if (app != null) {  
 app.switchLayout(studentportal.getPane());  
 } else {  
 System.*err*.println("MainApp reference is null");  
 }  
  
 }  
 else{  
 System.*out*.println("no type selected");  
 }  
 });  
  
 VBox centrepart = new VBox();  
 centrepart.getChildren().addAll(typebox, Name, textField, Pasword, passwordField, button);  
 centrepart.setSpacing(30);  
 centrepart.setAlignment(Pos.*CENTER*);  
 centrepart.setPadding(new Insets(0, 0, 0, 200));  
  
 VBox combinedBox = new VBox();  
 combinedBox.getChildren().addAll(logo(), centrepart);  
 combinedBox.setSpacing(100);  
 combinedBox.setPadding(new Insets(20));  
 centrepart.setMaxWidth(450);  
  
 return combinedBox;  
 }  
  
 public BorderPane getPane() {  
 return pane;  
 }  
  
 private HBox logo() {  
 Image logo=new Image(this.getClass().getResource("/481px-COMSATS\_new\_logo.jpg").toExternalForm());  
  
 ImageView logoview = new ImageView(logo);  
 logoview.setFitHeight(50);  
 logoview.setFitWidth(50);  
 logoview.setPreserveRatio(true);  
  
 Circle clip = new Circle(25, 25, 25);  
 logoview.setClip(clip);  
  
 Label uni = new Label("CUI PORTAL");  
 uni.setFont(Font.*font*("Brush Script MT", FontWeight.*BOLD*, FontPosture.*ITALIC*, 36));  
 uni.setPadding(new Insets(5, 0, 0, 0));  
  
 HBox topbox = new HBox();  
 topbox.getChildren().addAll(logoview, uni);  
 topbox.setSpacing(10);  
 topbox.setPadding(new Insets(50, 0, 0, 50));  
  
 return topbox;  
}  
}**

**package com.example.demo;  
  
  
import javafx.beans.property.SimpleStringProperty;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableArray;  
import javafx.collections.ObservableList;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Group;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.control.cell.PropertyValueFactory;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.\*;  
import javafx.scene.shape.Circle;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontPosture;  
import javafx.scene.text.FontWeight;  
import javafx.scene.text.Text;  
import javafx.stage.Stage;  
import org.w3c.dom.ls.LSOutput;  
  
import java.io.IOException;  
import java.net.ConnectException;  
import java.sql.\*;  
  
public class AdminDashboard {  
  
 private BorderPane pane;  
 private Button selectedButton = null;  
  
  
 public AdminDashboard() {  
  
 pane = new BorderPane();  
  
 pane.setStyle("-fx-background-color:#C8A97D");  
  
 VBox navigationBar = dashboardmain();  
 pane.setLeft(navigationBar);  
 pane.setCenter(upperpart());  
  
  
 }  
  
  
 private VBox dashboardmain() {  
 // vbox to add in centre  
 VBox vBox = new VBox();  
 vBox.setSpacing(20);  
 vBox.setStyle("-fx-background-color:#6F4E37");  
  
  
 vBox.setPrefWidth(90);  
  
  
 Image logoImage = new Image(getClass().getResource("/menu-burger.png").toExternalForm()); // Replace with your logo image  
 ImageView logoView = new ImageView(logoImage);  
 logoView.setFitWidth(50);  
 logoView.setFitHeight(50);  
 logoView.setClip(new Circle(25, 25, 25)); // Circular clip for logo  
  
 // Create buttons with icons  
 Button homeButton = createNavButton("/home.png");  
 Button settingsButton = createNavButton("/settings.png");  
 Button userButton = createNavButton("/user.png");  
 Button toolsButton = createNavButton("/globe.png");  
 Button editButton = createNavButton("/edit.png");  
 Button appsButton = createNavButton("/user.png");  
 Button deleteuserbutton = createNavButton("/trash-xmark.png");  
 Button deletefacultybutton = createNavButton("/cross-circle.png");  
  
  
 // A bottom section button  
 Button exitButton = createNavButton("/play.png");  
 exitButton.setStyle("-fx-background-color:#6F4E37 ;");  
  
  
 // Add buttons to the navigation bar  
 vBox.getChildren().addAll(logoView, homeButton, settingsButton, userButton, toolsButton, editButton, appsButton, deleteuserbutton, deletefacultybutton, exitButton);  
 VBox.*setMargin*(exitButton, new Insets(18, 0, 0, 0));  
 vBox.setPadding(new Insets(40, 0, 0, 25));  
 vBox.setSpacing(15);  
  
 userButton.setOnAction(actionEvent -> pane.setCenter(user\_part()));  
 homeButton.setOnAction(actionEvent -> pane.setCenter(upperpart()));  
 editButton.setOnAction(actionEvent -> pane.setCenter(course\_part()));  
 toolsButton.setOnAction(actionEvent -> pane.setCenter(course\_info\_part()));  
 appsButton.setOnAction(actionEvent -> pane.setCenter(user\_info\_display()));  
  
 return vBox;  
  
 }  
  
  
 private VBox upperpart() {  
 // Vbox for dashboard centre part  
 VBox dashcentre = new VBox();  
 dashcentre.getChildren().addAll(logo(), dashboardcentre());  
 dashcentre.*setMargin*(logo(), new Insets(50, 0, 0, 0));  
 dashcentre.setSpacing(100);  
 return dashcentre;  
  
 }  
  
  
 private VBox course\_part() {  
 VBox addcourse = new VBox();  
 addcourse.getChildren().addAll(logo(), Add\_Course());  
 addcourse.setSpacing(30);  
  
 return addcourse;  
 }  
  
 public BorderPane getPane() {  
 return pane;  
 }  
  
 private VBox user\_part() {  
 VBox user = new VBox();  
 user.getChildren().addAll(logo(), Add\_User());  
 user.setSpacing(30);  
 return user;  
 }  
 private VBox course\_info\_part() {  
 VBox vbox = new VBox();  
 vbox.getChildren().addAll(logo(), course\_info());  
 vbox.setSpacing(30);  
 return vbox;  
 }  
  
  
 private Button createNavButton(String iconPath) {  
 // Create a button  
 Button button = new Button();  
 button.setStyle("-fx-background-color:#6F4E37 ;"); // Match nav background color  
 button.setPadding(new Insets(10));  
 button.setPrefSize(50, 50); // Square buttons  
  
 // Add an icon to the button  
 Image icon = new Image(getClass().getResource(iconPath).toExternalForm());  
 ImageView iconView = new ImageView(icon);  
 iconView.setFitWidth(30);  
 iconView.setFitHeight(30);  
 button.setGraphic(iconView);  
  
  
 // Change button style on hover  
 // Change button style on hover  
 button.setOnMouseClicked(e -> {  
 if (selectedButton != null) {  
 // Reset the previously selected button  
 selectedButton.setStyle("-fx-background-color:#6F4E37;");  
 }  
  
 // Set the current button as selected and change its color  
 button.setStyle("-fx-background-color: #A0522D;");  
 selectedButton = button; // Update the currently selected button  
 });  
  
 return button;  
 }  
  
 private HBox logo() {  
 Image logo=new Image(this.getClass().getResource("/481px-COMSATS\_new\_logo.jpg").toExternalForm());  
 ImageView logoview = new ImageView(logo);  
 logoview.setFitHeight(50);  
 logoview.setFitWidth(50);  
 logoview.setPreserveRatio(true);  
  
 //shaping of logo  
 Circle Clip = new Circle(25, 25, 25); // (centerX, centerY, radius)  
 logoview.setClip(Clip);  
  
  
 // uni name  
 // uni name  
 Label uni = new Label("CUI PORTAL");  
  
 uni.setFont(Font.*font*("Brush Script MT", FontWeight.*BOLD*, FontPosture.*ITALIC*, 36));  
 uni.setPadding(new Insets(5, 0, 0, 0));  
 HBox topbox = new HBox();  
 topbox.getChildren().addAll(logoview, uni);  
 topbox.setSpacing(10);  
 topbox.setPadding(new Insets(50, 0, 0, 50));  
  
 return topbox;  
  
 }  
  
 private Button createButton(String iconPath, String text) {  
 Button button = new Button();  
 button.setStyle("-fx-background-color:#6F4E37; -fx-text-fill: white; -fx-font-size: 20;");  
  
 try {  
 // Load the icon  
 Image icon = new Image(getClass().getResource(iconPath).toExternalForm());  
 ImageView iconView = new ImageView(icon);  
 iconView.setFitHeight(20);  
 iconView.setFitWidth(20);  
  
 button.setGraphic(iconView);  
 button.setGraphicTextGap(10);  
 button.setText(text);  
 button.setContentDisplay(ContentDisplay.*LEFT*);  
 button.setPadding(new Insets(10));  
 button.setMaxSize(300, 300);  
 }  
 catch(Exception e)  
  
 {  
 System.*err*.println("Could not load icon for: " + iconPath);  
 }  
  
 return button;  
 }  
  
 private GridPane dashboardcentre() {  
  
 GridPane gridPane = new GridPane();  
 gridPane.setPadding(new Insets(20));  
 gridPane.setHgap(30);  
 gridPane.setVgap(30);  
 gridPane.setAlignment(Pos.*CENTER*);  
  
  
 // Buttons  
 Button home = createButton("/home.png", "Home");  
 Button addCourseButton = createButton("/edit.png", "Add Course");  
 Button addUserButton = createButton("/user.png", "Add User");  
 Button coursesButton = createButton("/globe.png", "Courses");  
 Button usersButton = createButton("/user.png", "Users");  
 Button updateFeeButton = createButton("/settings.png", "Update Fee");  
 Button logoutButton = createButton("/play.png", "Logout");  
 Button deleteuser = createButton("/trash-xmark.png", "Delete User");  
 Button deletefaculty = createButton("/cross-circle.png", "Delete Faculty");  
  
 coursesButton.setTranslateY(15);  
 usersButton.setTranslateY(15);  
 updateFeeButton.setTranslateY(15);  
 deleteuser.setTranslateY(25);  
 deletefaculty.setTranslateY(25);  
  
 home.setTranslateX(-80);  
 addCourseButton.setTranslateX(-30);  
 addUserButton.setTranslateX(20);  
 coursesButton.setTranslateX(-80);  
 usersButton.setTranslateX(-30);  
 updateFeeButton.setTranslateX(20);  
 logoutButton.setTranslateX(450);  
 logoutButton.setTranslateY(100);  
 deletefaculty.setTranslateX(-30);  
 deleteuser.setTranslateX(-80);  
  
  
 home.setPadding(new Insets(30, 50, 30, 50));  
 addCourseButton.setPadding(new Insets(30, 50, 30, 50));  
 addUserButton.setPadding(new Insets(30, 50, 30, 50));  
 coursesButton.setPadding(new Insets(30, 50, 30, 50));  
 usersButton.setPadding(new Insets(30, 50, 30, 50));  
 updateFeeButton.setPadding(new Insets(30, 50, 30, 50));  
  
 deletefaculty.setPadding(new Insets(30, 50, 30, 50));  
 deleteuser.setPadding(new Insets(30, 50, 30, 50));  
  
  
 home.setOnAction(actionEvent -> pane.setCenter(upperpart()));  
 addCourseButton.setOnAction(actionEvent -> pane.setCenter(course\_part()));  
 addUserButton.setOnAction(actionEvent -> pane.setCenter(user\_part()));  
  
  
 // Add buttons to the GridPane in the required layout  
  
 // Add buttons to the GridPane in the required layout  
  
 gridPane.add(home, 0, 0);  
 gridPane.add(addCourseButton, 1, 0);  
 gridPane.add(addUserButton, 2, 0);  
 gridPane.add(coursesButton, 0, 1);  
 gridPane.add(usersButton, 1, 1);  
 gridPane.add(updateFeeButton, 2, 1);  
 gridPane.add(logoutButton, 1, 2);  
 gridPane.add(deleteuser, 0, 2);  
 gridPane.add(deletefaculty, 1, 2);  
  
 return gridPane;  
  
  
 }  
 //Connection with database  
 String url="jdbc:mysql://localhost:3306/university\_system";  
 String username="root";  
 String password\_sql="0348jav.";  
  
 VBox Add\_Course() {  
  
 Text txt = new Text("COURSE REGISTRATION");  
 txt.setStyle("-fx-font-weight:bold;-fx-font-size:40;-fx-font-style:italic;-fx-font-family:Arial");  
  
  
 Label course\_name = new Label("COURSE NAME ");  
 TextField name = new TextField();  
 HBox hbox1 = new HBox(20);  
 hbox1.getChildren().addAll(course\_name, name);  
  
 Label course\_code = new Label("COURSE CODE ");  
 TextField code = new TextField();  
 HBox hbox2 = new HBox(20);  
 hbox2.getChildren().addAll(course\_code, code);  
  
 Label credit\_Hrs = new Label("CREDIT\_Hrs ");  
 ChoiceBox<Integer> hrs = new ChoiceBox();  
 ObservableList<Integer> hrs\_list = FXCollections.*observableArrayList*(1, 2, 3);  
 hrs.setItems(hrs\_list);  
 HBox hbox3 = new HBox(20);  
 hbox3.getChildren().addAll(credit\_Hrs, hrs);  
  
 Label semester = new Label("SEMESTER ");  
 ChoiceBox<Integer> sem = new ChoiceBox<Integer>();  
 ObservableList<Integer> sem\_list = FXCollections.*observableArrayList*(1, 2, 3, 4, 5, 6, 7, 8);  
 sem.setItems(sem\_list);  
 HBox hbox4 = new HBox(20);  
 hbox4.getChildren().addAll(semester, sem);  
  
 Label course\_type = new Label("TYPE ");  
 ChoiceBox<String> type = new ChoiceBox<>();  
 type.getItems().addAll("CORE", "ELECTIVE");  
 HBox hbox5 = new HBox(20);  
 hbox5.getChildren().addAll(course\_type, type);  
  
 Label instructor\_name = new Label("INSTRUCTOR NAME");  
 TextField instructor = new TextField();  
 HBox hbox6 = new HBox(20);  
 hbox6.getChildren().addAll(instructor\_name, instructor);  
  
  
 Label dept\_name = new Label("DEPARTMENT");  
 TextField dept = new TextField();  
 HBox hbox7 = new HBox(20);  
 hbox7.getChildren().addAll(dept\_name, dept);  
  
 VBox vbox = new VBox(30);  
 vbox.setTranslateX(300);  
 Region space = new Region();  
 Button button = new Button("Add Course");  
 button.setFont(Font.*font*("Arial", FontWeight.*BOLD*, FontPosture.*ITALIC*, 10));  
 button.setTranslateX(270);  
 button.setPrefHeight(40);  
 button.setPrefWidth(80);  
 button.setOnAction(e->{  
 //Using Databases for saving information  
 try{  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection= DriverManager.*getConnection*(url,username,password\_sql);  
 connection.setAutoCommit(false);  
 String sql="insert into Course values(?,?,?,?,?,?,?)";  
 PreparedStatement statement=connection.prepareStatement(sql);  
 statement.setString(1,code.getText());  
 statement.setString(2,name.getText());  
 statement.setInt(3,hrs.getValue());  
 statement.setInt(4,sem.getValue());  
 statement.setString(5,type.getValue());  
 statement.setString(6,instructor.getText());  
 statement.setString(7,dept.getText());  
 statement.executeUpdate();  
 connection.commit();  
 statement.close();  
 connection.close();  
 }catch(Exception exception){  
 System.*out*.println(exception.getMessage());  
 }  
 Alert alert=new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setTitle("Course Information");  
 alert.setContentText("Course has been saved successfully");  
 alert.show();  
 });  
  
 Button button3 = new Button(" Remove Course");  
 button3.setFont(Font.*font*("Arial", FontWeight.*BOLD*, FontPosture.*ITALIC*, 10));  
 button3.setTranslateX(270);  
 button3.setPrefHeight(40);  
 button3.setPrefWidth(80);  
 button3.setOnAction(e->{  
 try{  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection= DriverManager.*getConnection*(url,username,password\_sql);  
 connection.setAutoCommit(false);  
 String sql="delete from Course where Course\_code=?";  
 PreparedStatement statement=connection.prepareStatement(sql);  
 statement.setString(1,code.getText());  
 statement.executeUpdate();  
 connection.commit();  
 statement.close();  
 connection.close();  
 }catch(Exception exception){  
 System.*out*.println(exception.getMessage());  
 }  
 Alert alert=new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setTitle("Course Information");  
 alert.setContentText("Course has been removed successfully");  
 alert.show();  
  
 });  
 Button button2=new Button("CLEAR INFO");  
 vbox.getChildren().addAll(txt, space, hbox1, hbox2, hbox3, hbox4, hbox5, hbox6, hbox7, button,button2,button3);  
 FontStyling(vbox);  
 button2.setOnAction(e->{  
 for(var hbox: vbox.getChildren()){  
 if(hbox instanceof HBox){  
 for(var fields: ((HBox)hbox).getChildren()){  
 if(fields instanceof TextField){  
 ((TextField)fields).clear();  
 }  
 }  
 }  
 }  
 });  
 return vbox;  
 }  
  
 public void FontStyling(VBox vbox) {  
 for (var hbox : vbox.getChildren()) {  
 if (hbox instanceof HBox) {  
 for (var h : ((HBox) hbox).getChildren()) {  
 if (h instanceof Label) {  
 if (!((Label) h).getText().equalsIgnoreCase("CUI PORTAL")) {  
 ((Label) h).setFont(Font.*font*("Arial", FontWeight.*BOLD*, FontPosture.*ITALIC*, 15));  
 ((Label) h).setPrefWidth(150);  
 }  
 }  
 if (h instanceof TextField || h instanceof ChoiceBox) {  
 ((Control) h).setPrefWidth(190);  
  
 }  
 }  
 }  
  
 }  
 }  
 private VBox Add\_User() {  
 Text txt = new Text("REGISTER USER ");  
 txt.setStyle("-fx-font-weight:bold;-fx-font-size:30;-fx-font-style:italic;-fx-font-family:Arial");  
  
 Label first\_name = new Label("FIRST NAME ");  
 Label last\_name = new Label("LAST NAME");  
 HBox HBox1 = new HBox(90, first\_name, last\_name);  
  
  
 TextField first = new TextField();  
 TextField last = new TextField();  
 HBox HBox2 = new HBox(50, first, last);  
  
 Label Reg\_number = new Label("ID/REGISTRATION NO");  
 Label password = new Label("PASSWORD");  
 HBox HBox3 = new HBox(95, Reg\_number, password);  
  
 TextField reg = new TextField();  
 TextField pass = new TextField();  
 HBox HBox4 = new HBox(50, reg, pass);  
  
 Label date = new Label("DateOfBirth");  
 Label type = new Label("TYPE");  
 HBox HBox5 = new HBox(85, date, type);  
  
 DatePicker dob = new DatePicker();  
 ChoiceBox<String> types = new ChoiceBox<>();  
 types.getItems().addAll("ADMIN","FACULTY","STUDENT");  
 HBox HBox6=new HBox(60,dob,types);  
  
  
 Label department = new Label("DEPARTMENT ");  
 Label semester = new Label("SEMESTER ");  
 HBox hbox7 = new HBox(85,department,semester);  
  
 TextField dept =new TextField();  
 ChoiceBox<Integer> sem = new ChoiceBox<Integer>();  
 sem.getItems().addAll(1,2,3,4,5,6,7,8);  
 Text text=new Text("Select it if Student/Teacher");  
 HBox hbox8 = new HBox(85,dept,sem,text);  
  
 Button button=new Button("REGISTER");  
 button.setFont(Font.*font*("Arial",FontWeight.*BOLD*,FontPosture.*ITALIC*,10));  
 button.setPrefHeight(40);  
 button.setPrefWidth(80);  
  
 Button button2 =new Button("CLEAR INFO");  
 button2.setFont(Font.*font*("Arial",FontWeight.*BOLD*,FontPosture.*ITALIC*,10));  
 button2.setPrefHeight(40);  
 button2.setPrefWidth(80);  
  
 Button button3 =new Button("Remove User");  
 button3.setFont(Font.*font*("Arial",FontWeight.*BOLD*,FontPosture.*ITALIC*,10));  
 button3.setPrefHeight(40);  
 button3.setPrefWidth(80);  
  
 HBox hbox9=new HBox(40,button,button2,button3);  
  
 Region space=new Region();  
 space.setPrefWidth(20);  
  
 //saving data inot database  
 button.setOnAction(e->{  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection = DriverManager.*getConnection*(url, username, password\_sql);  
 connection.setAutoCommit(false);  
 String sql = "insert into all\_users values(?,?,?,?,?,?,?)";  
 PreparedStatement statement = connection.prepareStatement(sql);  
 statement.setString(1, types.getValue());  
 statement.setString(2, reg.getText());  
 statement.setString(3, first.getText() + " " + last.getText());  
 statement.setString(4, pass.getText());  
 statement.setString(5, dob.getValue().toString());  
 statement.setInt(6, sem.getValue());  
 statement.setString(7,dept.getText());  
  
 statement.executeUpdate();  
 connection.commit();  
 statement.close();  
 connection.close();  
  
 }catch (Exception exception){  
  
 }  
  
 });  
 button3.setOnAction(e->{  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection = DriverManager.*getConnection*(url, username, password\_sql);  
 connection.setAutoCommit(false);  
 String sql="delete from all\_users where id=?and password=?";  
 PreparedStatement statement=connection.prepareStatement(sql);  
 statement.setString(1,reg.getText());  
 statement.setString(2,pass.getText());  
  
 statement.executeUpdate();  
 connection.commit();  
 statement.close();  
 connection.close();  
  
 }catch(Exception exception) {  
 System.*out*.println(exception.getMessage());  
 }  
 });  
  
 VBox vbox=new VBox(30);  
 vbox.setTranslateX(300);  
 vbox.setTranslateY(30);  
 vbox.getChildren().addAll(txt,space,HBox1,HBox2,HBox3,HBox4,HBox5,HBox6,hbox7,hbox8,hbox9);  
 FontStyling(vbox);  
  
 button2.setOnAction(e->{  
 for(var hbox: vbox.getChildren()){  
 if(hbox instanceof HBox){  
 for(var fields: ((HBox)hbox).getChildren()){  
 if(fields instanceof TextField){  
 ((TextField)fields).clear();  
 }  
 }  
 }  
 }  
 });  
 return vbox;  
}  
  
Group course\_info(){  
  
 TableView course\_table=new TableView();  
 TableColumn col1=new TableColumn("COURSE CODE");  
 TableColumn col2=new TableColumn("COURSE NAME");  
 TableColumn col3=new TableColumn("CREDIT Hrs");  
 TableColumn col4=new TableColumn("TEACHER Assigned");  
 TableColumn col5=new TableColumn("TYPE");  
 TableColumn col6=new TableColumn("SEMESTER");  
 TableColumn col7=new TableColumn("DEPARTMENT");  
  
 ObservableList<Course> course\_list=FXCollections.*observableArrayList*();  
 Button button =new Button("Remove Course");  
  
 //database  
 try{  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection = DriverManager.*getConnection*(url, username, password\_sql);  
 connection.setAutoCommit(false);  
 Statement statement=connection.createStatement();  
 ResultSet resultset=statement.executeQuery("Select \*from Course");  
 while(resultset.next()){  
 course\_list.add(  
 new Course(  
 resultset.getString(1),  
 resultset.getString(2),  
 resultset.getInt(3),  
 resultset.getString(6),  
 resultset.getString(5),  
 resultset.getInt(4),  
 resultset.getString(7)));  
  
 }  
 //adding values  
 col1.setCellValueFactory(new PropertyValueFactory<Course,String>("course\_code"));  
 col2.setCellValueFactory(new PropertyValueFactory<Course,String>("name"));  
 col3.setCellValueFactory(new PropertyValueFactory<Course,Integer>("hrs"));  
 col4.setCellValueFactory(new PropertyValueFactory<Course,String>("instructor"));  
 col5.setCellValueFactory(new PropertyValueFactory<Course,String>("type"));  
 col6.setCellValueFactory(new PropertyValueFactory<Course,Integer>("semester"));  
 col7.setCellValueFactory(new PropertyValueFactory<Course,String>("dept"));  
  
 //data in the table  
 course\_table.setItems(course\_list);  
 resultset.close();  
 statement.close();  
 connection.close();  
  
  
 button.setOnAction(e->{  
 Course selected=(Course)course\_table.getSelectionModel().getSelectedItem();  
 course\_list.remove(selected);  
  
 //Action on button  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection2 = DriverManager.*getConnection*(url, username, password\_sql);  
 connection2.setAutoCommit(false);  
 String sql = "delete from Course where Course\_code=?";  
 PreparedStatement statement2 = connection2.prepareStatement(sql);  
 statement2.setString(1, selected.getCourse\_code());  
 statement2.executeUpdate();  
 connection2.commit();  
 statement2.close();  
 connection2.close();  
 }catch(Exception ex){  
  
 }  
 });  
  
 }catch(Exception e){  
 e.getStackTrace();  
 }  
 col1.setPrefWidth(150);  
 col2.setPrefWidth(150);  
 col3.setPrefWidth(150);  
 col4.setPrefWidth(150);  
 col5.setPrefWidth(150);  
 col6.setPrefWidth(150);  
 col7.setPrefWidth(150);  
  
 course\_table.getColumns().addAll(col1,col2,col3,col4,col5,col6,col7);  
 // for(var col: course\_table.getColumns()){  
 // }  
 course\_table.setPrefSize(1000,400);  
 course\_table.setTranslateX(100);  
 course\_table.setTranslateY(70);  
  
 // VBox vbox=new VBox(20,course\_table);  
 Group grp=new Group();  
 grp.getChildren().addAll(course\_table,button);  
 return grp;  
}  
  
//Show User Information  
Group user\_info\_display(){  
  
 TableView user\_table=new TableView();  
  
 TableColumn col1=new TableColumn("TYPE");  
 TableColumn col2=new TableColumn("ID");  
 TableColumn col3=new TableColumn("NAME");  
 TableColumn col4=new TableColumn("Password");  
 TableColumn col5=new TableColumn("DEPARTMENT");  
 TableColumn col6=new TableColumn("SEMESTER");  
 TableColumn col7=new TableColumn("DateOfBirth");  
  
 ObservableList<User> user\_list=FXCollections.*observableArrayList*();  
 Button button =new Button("Remove User");  
 //database  
 try{  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection=DriverManager.*getConnection*(url,username,password\_sql);  
 connection.setAutoCommit(false);  
 Statement statement=connection.createStatement();  
 ResultSet resultset=statement.executeQuery("Select \*from all\_users");  
  
 while(resultset.next()) {  
 user\_list.add(  
 new User(resultset.getString(1),  
 resultset.getString(2),  
 resultset.getString(3),  
 resultset.getString(4),  
 resultset.getString(7),  
 resultset.getInt(6),  
 resultset.getString(5))  
 );  
 }  
 col1.setCellValueFactory(new PropertyValueFactory<User,String>("type"));  
 col2.setCellValueFactory(new PropertyValueFactory<User,String>("id"));  
 col3.setCellValueFactory(new PropertyValueFactory<User,String>("name"));  
 col4.setCellValueFactory(new PropertyValueFactory<User,String>("password"));  
 col5.setCellValueFactory(new PropertyValueFactory<User,String>("department"));  
 col6.setCellValueFactory(new PropertyValueFactory<User,Integer>("semester"));  
 col7.setCellValueFactory(new PropertyValueFactory<User,String>("dob"));  
  
 user\_table.setItems(user\_list);  
  
 statement.close();  
 resultset.close();  
 connection.close();  
  
 button.setOnAction(e->{  
 User selected=(User)user\_table.getSelectionModel().getSelectedItem();  
 user\_list.remove(selected);  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection2 = DriverManager.*getConnection*(url, username, password\_sql);  
 connection2.setAutoCommit(false);  
 String sql="delete from all\_users where id=?";  
 PreparedStatement statement2=connection2.prepareStatement(sql);  
 statement2.setString(1,selected.getId());  
 System.*out*.println(selected.getId());  
 statement2.executeUpdate();  
  
 connection2.commit();  
 statement2.close();  
 connection2.close();  
  
 }catch(Exception ex){  
 ex.getStackTrace();  
 }  
 });  
  
 }catch(Exception exception){  
  
 }  
  
 //button action  
  
 col1.setPrefWidth(150);  
 col2.setPrefWidth(150);  
 col3.setPrefWidth(150);  
 col4.setPrefWidth(150);  
 col5.setPrefWidth(150);  
 col6.setPrefWidth(150);  
 user\_table.getColumns().addAll(col1,col2,col3,col4,col5,col6,col7);  
  
 user\_table.setPrefSize(1000,400);  
 user\_table.setTranslateX(100);  
 user\_table.setTranslateY(70);  
 Group grp=new Group();  
 grp.getChildren().addAll(user\_table,button);  
 return grp;  
}  
  
}**

**package com.example.demo;  
  
import javafx.application.Application;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Group;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.control.cell.PropertyValueFactory;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.BorderPane;  
import javafx.scene.layout.GridPane;  
import javafx.scene.layout.HBox;  
import javafx.scene.layout.VBox;  
import javafx.scene.shape.Circle;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontPosture;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
  
import javax.swing.\*;  
import java.io.IOException;  
import java.sql.\*;  
  
public class FacultyDashboard {  
  
 private BorderPane pane;  
 private Button selectedButton = null;  
  
 public FacultyDashboard() {  
  
 pane = new BorderPane();  
  
 pane.setStyle("-fx-background-color:#C8A97D");  
  
 VBox navigation = dashboardmain();  
 pane.setLeft(navigation);  
 pane.setCenter(upperpart());  
  
 }  
  
 private VBox dashboardmain() {  
 // vbox to add in centre  
 VBox vBox = new VBox();  
 vBox.setSpacing(20);  
 vBox.setStyle("-fx-background-color:#6F4E37");  
  
 vBox.setPrefWidth(90);  
  
  
 Image logoImage = new Image(this.getClass().getResource("/oop images.jpg").toExternalForm());  
 ImageView logoView = new ImageView(logoImage);  
 logoView.setFitWidth(50);  
 logoView.setFitHeight(50);  
 logoView.setClip(new Circle(25, 25, 25)); // Circular clip for logo  
  
 // Create buttons with icons  
 Button homeButton = createNavButton("/home.png");  
 Button settingsButton = createNavButton("/graduation-cap.png");  
 Button userButton = createNavButton("/bookmark.png");  
 Button attendButton = createNavButton("/apps.png");  
 Button editButton = createNavButton("/settings.png");  
  
  
 // A bottom section button  
 Button exitButton = createNavButton("/play.png");  
 exitButton.setStyle("-fx-background-color:#6F4E37 ;");  
  
  
 // Add buttons to the navigation bar  
 vBox.getChildren().addAll(logoView, homeButton, settingsButton, userButton, attendButton, editButton, exitButton);  
 VBox.*setMargin*(exitButton, new Insets(60, 0, 0, 0));  
 vBox.setPadding(new Insets(40, 0, 0, 25));  
 vBox.setSpacing(15);  
  
  
 homeButton.setOnAction(actionEvent -> pane.setCenter(upperpart()));  
 attendButton.setOnAction(actionEvent -> pane.setCenter(Attendance()));  
  
 return vBox;  
  
 }  
  
 private VBox upperpart() {  
 // Vbox for dashboard centre part  
 VBox dashcentre = new VBox();  
 dashcentre.getChildren().addAll(logo(), dashboardcentre());  
 dashcentre.*setMargin*(logo(), new Insets(50, 0, 0, 0));  
 dashcentre.setSpacing(100);  
 return dashcentre;  
  
 }  
  
 public BorderPane getPane() {  
 return pane;  
 }  
  
  
 private Button createNavButton(String iconPath) {  
 // Create a button  
 Button button = new Button();  
 button.setStyle("-fx-background-color:#6F4E37 ;"); // Match nav background color  
 button.setPadding(new Insets(10));  
 button.setPrefSize(50, 50); // Square buttons  
  
 // Add an icon to the button  
 Image icon = new Image(getClass().getResource(iconPath).toExternalForm());  
 ImageView iconView = new ImageView(icon);  
 iconView.setFitWidth(30);  
 iconView.setFitHeight(30);  
 button.setGraphic(iconView);  
  
  
 // Change button style on hover  
 button.setOnMouseClicked(e -> {  
 if (selectedButton != null) {  
 // Reset the previously selected button  
 selectedButton.setStyle("-fx-background-color:#6F4E37;");  
 }  
  
 // Set the current button as selected and change its color  
 button.setStyle("-fx-background-color: #A0522D;");  
 selectedButton = button; // Update the currently selected button  
 });  
  
 return button;  
 }  
  
 private HBox logo() {  
 Image logo = new Image(this.getClass().getResource("/481px-COMSATS\_new\_logo.jpg").toExternalForm());  
 ImageView logoview = new ImageView(logo);  
 logoview.setFitHeight(50);  
 logoview.setFitWidth(50);  
 logoview.setPreserveRatio(true);  
  
 //shaping of logo  
 Circle Clip = new Circle(25, 25, 25); // (centerX, centerY, radius)  
 logoview.setClip(Clip);  
  
 // uni name  
 Label uni = new Label("CUI PORTAL");  
  
 uni.setFont(Font.*font*("Brush Script MT", FontWeight.*BOLD*, FontPosture.*ITALIC*, 36));  
 uni.setPadding(new Insets(5, 0, 0, 0));  
 HBox topbox = new HBox();  
 topbox.getChildren().addAll(logoview, uni);  
 topbox.setSpacing(10);  
 topbox.setPadding(new Insets(50, 0, 0, 50));  
  
 return topbox;  
  
 }  
  
 private Button createButton(String iconPath, String text) {  
 Button button = new Button();  
 button.setStyle("-fx-background-color:#6F4E37; -fx-text-fill: white; -fx-font-size: 20;");  
  
 try {  
 // Load the icon  
 Image icon = new Image(getClass().getResource(iconPath).toExternalForm());  
 ImageView iconView = new ImageView(icon);  
 iconView.setFitHeight(20);  
 iconView.setFitWidth(20);  
  
 button.setGraphic(iconView);  
 button.setGraphicTextGap(10);  
 button.setText(text);  
 button.setContentDisplay(ContentDisplay.*LEFT*);  
 button.setPadding(new Insets(10));  
 button.setMaxSize(300, 300);  
 } catch (Exception e) {  
 System.*err*.println("Could not load icon for: " + iconPath);  
 }  
  
 return button;  
 }  
  
 private GridPane dashboardcentre() {  
  
 GridPane gridPane = new GridPane();  
 gridPane.setPadding(new Insets(20));  
 gridPane.setHgap(30);  
 gridPane.setVgap(30);  
 gridPane.setAlignment(Pos.*CENTER*);  
  
  
 // Buttons  
 Button home = createButton("/home.png", "Home");  
 Button addCourseButton = createButton("/graduation-cap.png", "Study Plan");  
 Button addUserButton = createButton("/bookmark.png", "Marks");  
 Button attendanceButton = createButton("/apps.png", "Attendence");  
 Button usersButton = createButton("/settings.png", "Transcript");  
 Button logoutButton = createButton("/play.png", "Logout");  
  
  
 attendanceButton.setTranslateY(15);  
 usersButton.setTranslateY(15);  
  
  
 home.setTranslateX(-80);  
 addCourseButton.setTranslateX(-30);  
 addUserButton.setTranslateX(20);  
 attendanceButton.setTranslateX(-80);  
 usersButton.setTranslateX(-30);  
  
 logoutButton.setTranslateX(450);  
 logoutButton.setTranslateY(100);  
  
  
 home.setPadding(new Insets(30, 50, 30, 50));  
 addCourseButton.setPadding(new Insets(30, 50, 30, 50));  
 addUserButton.setPadding(new Insets(30, 50, 30, 50));  
 attendanceButton.setPadding(new Insets(30, 50, 30, 50));  
 usersButton.setPadding(new Insets(30, 50, 30, 50));  
  
 home.setOnAction(actionEvent -> pane.setCenter(upperpart()));  
  
 attendanceButton.setOnAction(actionEvent -> pane.setCenter(Attendance()));  
  
  
 // Add buttons to the GridPane in the required layout  
  
 gridPane.add(home, 0, 0);  
 gridPane.add(addCourseButton, 1, 0);  
 gridPane.add(addUserButton, 2, 0);  
 gridPane.add(attendanceButton, 0, 1);  
 gridPane.add(usersButton, 1, 1);  
 gridPane.add(logoutButton, 1, 2);  
  
  
 return gridPane;  
  
 }  
 String url="jdbc:mysql://localhost:3306/university\_system";  
 String username="root";  
 String password\_sql="0348jav.";  
  
 VBox Attendance(){  
  
 Label course\_label=new Label("Course\_Code");  
 TextField course\_field=new TextField();  
 DatePicker date\_attendance=new DatePicker();  
 HBox hbox=new HBox(20,course\_label,course\_field,date\_attendance);  
 Button attend=new Button("Show Attendance Sheet");  
 Button upload=new Button("UPLOAD");  
 HBox hbox2=new HBox(30,attend,upload);  
  
 VBox vbox1=new VBox(20,hbox,hbox2);  
  
  
 TableView<Student> table=new TableView();  
 // three colums to be added  
 TableColumn<Student,String> c1=new TableColumn("STUDENT Name");  
 TableColumn <Student,String>c2=new TableColumn("Registration no");  
 TableColumn<Student,Void> c3=new TableColumn("Attendance");  
  
 ObservableList<Student> student\_list = FXCollections.*observableArrayList*();  
  
 attend.setOnAction(e->{  
 //connection with database  
 try{  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection= DriverManager.*getConnection*(url,username,password\_sql);  
 connection.setAutoCommit(false);  
 Statement statement=connection.createStatement();  
 ResultSet resultSet=statement.executeQuery("Select \*from course inner join all\_users " +  
 "on course.Semester=all\_users.semester and course.dept=all\_users.department and all\_users.type='STUDENT'");  
  
 int count=0;  
 while(resultSet.next()) {  
 if (course\_field.getText().equalsIgnoreCase(resultSet.getString("Course\_code"))) {  
 student\_list.add(new Student(  
 resultSet.getString("user\_name"),  
 resultSet.getString("id")  
 // saved\_st.setString(1,resultSet.getString("user\_name"));  
 ));  
 count++;  
 }  
 }  
 connection.commit();  
 statement.close();  
 resultSet.close();  
 connection.close();  
  
 }catch(Exception exception){  
 exception.getStackTrace();  
 }  
 c1.setCellValueFactory(new PropertyValueFactory("studentname"));  
 c2.setCellValueFactory(new PropertyValueFactory("registration"));  
 c3.setCellFactory( col -> new RadioCell());  
  
 table.setItems(student\_list);  
  
 Group grp=new Group();  
 grp.prefWidth(40);  
 grp.getChildren().addAll(table);  
 VBox tab\_vbox=new VBox(vbox1,grp);  
 pane.setCenter(tab\_vbox);  
  
 });  
  
 upload.setOnAction(actionEvent->{  
 try {  
 Class.*forName*("com.mysql.cj.jdbc.Driver");  
 Connection connection2= DriverManager.*getConnection*(url,username,password\_sql);  
 connection2.setAutoCommit(false);  
  
 String sql="insert into attendance\_info values(?,?,?,?,?)";  
 PreparedStatement saved\_st=connection2.prepareStatement(sql);  
  
 for (Student list : student\_list) {  
  
 saved\_st.setString(1, list.getStudentname());  
 saved\_st.setString(2, list.getRegistration());  
 saved\_st.setString(3,course\_field.getText());  
 saved\_st.setString(4,list.isPresent()?"Present":"Absent");  
 saved\_st.setString(5,date\_attendance.getValue().toString());  
  
 saved\_st.addBatch();  
 }  
 saved\_st.executeBatch();  
 connection2.commit();  
 saved\_st.close();  
 connection2.close();  
 System.*out*.println("Attendance data uploaded successfully!");  
 }catch(Exception excep){  
 excep.getStackTrace();  
 }  
 });  
  
  
 table.setPrefWidth(300);  
 table.setPrefHeight(300);  
  
 c1.setPrefWidth(150);  
 c2.setPrefWidth(150);  
 c3.setPrefWidth(200);  
  
  
 table.getColumns().addAll(c1,c2,c3);  
 table.setPrefSize(800,400);  
 table.setTranslateX(100);  
 // table.setTranslateY(70);  
 // VBox vbox=new VBox(vbox1,table);  
 vbox1.setSpacing(30);  
 vbox1.setAlignment(Pos.*TOP\_LEFT*);  
  
  
 VBox vbox=new VBox(30,vbox1);  
 return vbox;  
 }  
}**

**package com.example.demo;  
  
import javafx.beans.property.SimpleBooleanProperty;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.scene.control.RadioButton;  
import javafx.scene.control.TableCell;  
import javafx.scene.control.ToggleGroup;  
  
public class Student {  
 private SimpleStringProperty studentname;  
 private SimpleStringProperty registration;  
 SimpleBooleanProperty present = new SimpleBooleanProperty(false);  
  
 public Student(String studentname,String registration) {  
 this.studentname =new SimpleStringProperty(studentname);  
 this.registration=new SimpleStringProperty(registration);  
 }  
  
 public void setRegistration(String registration) {  
 this.registration.set(registration);  
 }  
  
 public void setStudentname(String studentname) {  
 this.studentname.set(studentname);  
 }  
  
 public String getRegistration() {  
 return registration.get();  
 }  
  
 public SimpleStringProperty registrationProperty() {  
 return registration;  
 }  
  
 public String getStudentname() {  
 return studentname.get();  
 }  
  
 public SimpleStringProperty studentnameProperty() {  
 return studentname;  
 }  
 public boolean isPresent() {  
 return present.get();  
 }  
  
 public void setPresent(boolean isPresent) {  
 this.present.set(isPresent);  
 }  
  
 public SimpleBooleanProperty presentProperty() {  
 return present;  
 }  
}**

**package com.example.demo;  
  
import javafx.scene.control.RadioButton;  
import javafx.scene.control.TableCell;  
import javafx.scene.control.ToggleGroup;  
import javafx.scene.layout.HBox;  
  
public class RadioCell extends TableCell<Student,Void> {  
 RadioButton presentRadioButton = new RadioButton("Present");  
 RadioButton absentRadioButton = new RadioButton("Absent");  
  
  
 public RadioCell(){  
 presentRadioButton.setOnAction(e -> {  
 if (presentRadioButton.isSelected()) {  
 absentRadioButton.setSelected(false);  
 Student student = getTableRow().getItem();  
 student.setPresent(true);  
 }  
 });  
  
 absentRadioButton.setOnAction(e -> {  
 if (absentRadioButton.isSelected()) {  
 presentRadioButton.setSelected(false);  
 Student student = getTableRow().getItem();  
 student.setPresent(false);  
 }  
 });  
  
 // Add both buttons to an HBox  
 HBox container = new HBox(10, presentRadioButton, absentRadioButton);  
 setGraphic(container);  
  
 }  
 @Override  
 protected void updateItem(Void item, boolean empty) {  
 super.updateItem(item, empty);  
 if (empty) {  
 setGraphic(null); // Remove content if cell is empty  
 } else {  
 // Display the RadioButton if cell is not empty  
 Student student = getTableView().getItems().get(getIndex());  
 presentRadioButton.setSelected(student.isPresent());  
 absentRadioButton.setSelected(!student.isPresent());  
 }  
 }  
 }**

**package com.example.demo;  
  
import javafx.beans.property.SimpleStringProperty;  
  
public class User {  
 SimpleStringProperty id;  
 SimpleStringProperty type;  
 SimpleStringProperty name;  
 SimpleStringProperty password;  
 SimpleStringProperty department;  
 int semester;  
 SimpleStringProperty dob;  
  
 public User(String type, String id,String name,  
 String password, String department,int semester, String dob) {  
 this.type = new SimpleStringProperty(type);  
 this.id = new SimpleStringProperty(id);  
 this.name = new SimpleStringProperty(name);  
 this.password = new SimpleStringProperty(password);  
 this.department=new SimpleStringProperty(department);  
 this.semester = semester;  
 this.dob=new SimpleStringProperty(dob);  
 }  
  
 public String getId() {  
 return id.get();  
 }  
  
 public SimpleStringProperty idProperty() {  
 return id;  
 }  
  
 public String getType() {  
 return type.get();  
 }  
  
 public SimpleStringProperty typeProperty() {  
 return type;  
 }  
  
 public String getName() {  
 return name.get();  
 }  
  
 public SimpleStringProperty nameProperty() {  
 return name;  
 }  
  
 public String getPassword() {  
 return password.get();  
 }  
  
 public SimpleStringProperty passwordProperty() {  
 return password;  
 }  
  
 public int getSemester() {  
 return semester;  
 }  
  
 public String getDob() {  
 return dob.get();  
 }  
  
 public SimpleStringProperty dobProperty() {  
 return dob;  
 }  
  
 public String getDepartment() {  
 return department.get();  
 }  
  
 public SimpleStringProperty departmentProperty() {  
 return department;  
 }  
}**

**package com.example.demo;  
  
import javafx.beans.property.SimpleStringProperty;  
  
 public class Course {  
 SimpleStringProperty course\_code;  
 SimpleStringProperty name;  
 int hrs;  
 SimpleStringProperty instructor;  
 SimpleStringProperty type;  
 int semester;  
 SimpleStringProperty dept;  
  
 public Course(String course\_code, String name, int hrs, String instructor,  
 String type, int semester, String dept) {  
 this.course\_code = new SimpleStringProperty(course\_code);  
 this.name = new SimpleStringProperty(name);  
 this.hrs = hrs;  
 this.instructor = new SimpleStringProperty(instructor);  
 this.type = new SimpleStringProperty(type);  
 this.semester = semester;  
 this.dept = new SimpleStringProperty(dept);  
 }  
  
 public String getCourse\_code() {  
 return course\_code.get();  
 }  
  
 public SimpleStringProperty course\_codeProperty() {  
 return course\_code;  
 }  
  
 public String getName() {  
 return name.get();  
 }  
  
 public SimpleStringProperty nameProperty() {  
 return name;  
 }  
  
 public int getHrs() {  
 return hrs;  
 }  
  
  
 public String getInstructor() {  
 return instructor.get();  
 }  
  
 public SimpleStringProperty instructorProperty() {  
 return instructor;  
 }  
  
 public String getType() {  
 return type.get();  
 }  
  
 public SimpleStringProperty typeProperty() {  
 return type;  
 }  
  
 public int getSemester() {  
 return semester;  
 }  
  
 public String getDept() {  
 return dept.get();  
 }  
  
 public SimpleStringProperty deptProperty() {  
 return dept;  
 }  
 }**