```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.model.CourseManagement;
import com.example.studentmanagementsystem.model.Student;
import com.example.studentmanagementsystem.model.StudentManagement;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.event.Event;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Node;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;
import javafx.stage.Modality;
import javafx.stage.Stage;
import com.example.studentmanagementsystem.model.Course;
import javafx.fxml.FXML;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;

/**
 * The type Main controller.
 */
public class MainController {

    @FXML
    private TableView<Student> studentTable;
    @FXML
    private TableColumn<Student, String> nameColumn;
    @FXML
    private TableColumn<Student, String> idColumn;
    @FXML
```

```
36      private TableColumn<Student, Integer> ageColumn
    ;
37      @FXML
38      private TableColumn<Student, String>
    gradeColumn;
39
40      private ObservableList<Student> studentData =
    FXCollections.observableArrayList();
41      private Stage primaryStage;
42      @FXML
43      private TableView<Course> courseTable;
44      @FXML
45      private TableColumn<Course, String>
    courseNameColumn;
46      @FXML
47      private TableColumn<Course, String>
    courseIDColumn;
48
49      private ObservableList<Course> courseData =
    FXCollections.observableArrayList();
50
51
52      @FXML
53      private void initialize() {
54          // Initialize the student table with the
    four columns.
55          nameColumn.setCellValueFactory(cellData ->
    cellData.getValue().nameProperty());
56          idColumn.setCellValueFactory(cellData ->
    cellData.getValue().idProperty());
57          ageColumn.setCellValueFactory(cellData ->
    cellData.getValue().ageProperty().asObject());
58          gradeColumn.setCellValueFactory(cellData
     -> cellData.getValue().gradeProperty());
59
60          // Load initial data
61          studentData.addAll(StudentManagement.
    getStudents());
62          studentTable.setItems(studentData);
63
64      }
```

```
65
66      @FXML
67      private void handleAddStudentButtonAction(
    ActionEvent event) {
68          try {
69              // Load the FXML file
70              FXMLLoader loader = new FXMLLoader(
    getClass().getResource("/com/example/
    studentmanagementsystem/AddStudent.fxml"));
71
72              Parent root = loader.load();
73
74              AddStudentController controller =
    loader.getController();
75              controller.setMainController(this);
76
77
78              Stage dialogStage = new Stage();
79              dialogStage.setTitle("Add Student");
80              dialogStage.initModality(Modality.
    WINDOW_MODAL);  // This line makes the dialog
    modal
81              dialogStage.initOwner(primaryStage);
82              dialogStage.setScene(new Scene(root));
83
84  // Set the dialog stage in the controller
85              controller.setDialogStage(dialogStage
    );
86
87              // Show the dialog
88              dialogStage.showAndWait();
89
90          } catch (Exception e) {
91              e.printStackTrace();
92          }
93
94
95      }
96
97      /**
98       * Add student.
```

```
 99      *
100      * @param student the student
101      */
102     public void addStudent(Student student) {
103         studentData.add(student);
104     }
105 
106     /**
107      * Update student.
108      *
109      * @param student the student
110      */
111     public void updateStudent(Student student) {
112         // Assuming studentData is your
    ObservableList
113         int index = studentData.indexOf(student);
114         if (index != -1) {
115             studentData.set(index, student); //
    Update student in the list
116             studentTable.refresh();         //
    Refresh the table
117         }
118     }
119 
120 
121     /**
122      * Sets primary stage.
123      *
124      * @param primaryStage the primary stage
125      */
126     public void setPrimaryStage(Stage primaryStage
    ) {
127         this.primaryStage = primaryStage;
128     }
129 
130 
131     @FXML
132     private void handleAddCourseAction(ActionEvent
     event) {
133         try {
134             FXMLLoader loader = new FXMLLoader(
```

```
134 getClass().getResource("/com/example/
    studentmanagementsystem/AddCourse.fxml"));
135             Parent root = loader.load();
136 
137             Stage stage = new Stage();
138             stage.setTitle("Add Course");
139             stage.setScene(new Scene(root));
140             stage.show();
141 
142         } catch (Exception e) {
143             e.printStackTrace();
144         }
145     }
146 
147 
148     @FXML
149     private void handleEnrollStudentAction(
    ActionEvent event) {
150         try {
151             FXMLLoader loader = new FXMLLoader(
    getClass().getResource("/com/example/
    studentmanagementsystem/EnrollStudent.fxml"));
152             Parent root = loader.load();
153 
154             Stage stage = new Stage();
155             stage.setTitle("Enroll Student"); //
    Fixed the missing quotation mark
156             stage.setScene(new Scene(root));
157             stage.show();
158 
159         } catch (Exception e) {
160             e.printStackTrace();
161         }
162     }
163 
164     @FXML
165     private void handleAssignGradeAction(
    ActionEvent event) {
166         try {
167             FXMLLoader loader = new FXMLLoader(
    getClass().getResource("/com/example/
```

```
167 studentmanagementsystem/AssignGrade.fxml"));
168             Parent root = loader.load();
169 
170             Stage stage = new Stage();
171             stage.setTitle("Assign Grade"); //
    Fixed the title
172             stage.setScene(new Scene(root));
173             stage.show();
174 
175         } catch (Exception e) {
176             e.printStackTrace();
177         }
178     }
179 
180     @FXML
181     private void handleUpdateStudentButtonAction
    () {
182         Student selectedStudent = studentTable.
    getSelectionModel().getSelectedItem();
183         if (selectedStudent != null) {
184             try {
185                 FXMLLoader loader = new FXMLLoader
    (getClass().getResource("/com/example/
    studentmanagementsystem/UpdateStudent.fxml"));
186                 Parent root = loader.load();
187 
188                 UpdateStudentController controller
     = loader.getController();
189                 controller.setMainController(this
    );
190                 controller.setSelectedStudent(
    selectedStudent);
191 
192                 Stage stage = new Stage();
193                 stage.setTitle("Update Student");
194                 stage.setScene(new Scene(root));
195                 stage.show();
196             } catch (Exception e) {
197                 e.printStackTrace();
198             }
199         } else {
```

```
200 // No student selected, display a warning dialog
201             // ... you can use an Alert for this
    ...
202         }
203     }
204 
205     @FXML
206     private void handleDeleteStudentAction() {
207         int selectedIndex = studentTable.
    getSelectionModel().getSelectedIndex();
208         if (selectedIndex >= 0) {
209             studentTable.getItems().remove(
    selectedIndex);
210         } else {
211             // No student selected, display a
    warning dialog
212             // ... you can use an Alert for this
    ...
213         }
214     }
215     @FXML
216     private void
    handleViewStudentDetailsButtonAction() {
217         // Code to handle viewing student details
218     }
219 
220     /**
221      * Handle view courses action.
222      *
223      * @param event the event
224      */
225     public void handleViewCoursesAction(
    ActionEvent event) {
226         try {
227             FXMLLoader loader = new FXMLLoader(
    getClass().getResource("/com/example/
    studentmanagementsystem/ViewCourses.fxml"));
228             Parent root = loader.load();
229 
230             Stage stage = new Stage();
231             stage.setTitle("View Courses");
```

```
232                 stage.setScene(new Scene(root));
233                 stage.show();
234
235             } catch (Exception e) {
236                 e.printStackTrace();
237             }
238         }
239 }
240
```

```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.model.Student;
import com.example.studentmanagementsystem.model.StudentManagement;
import javafx.event.ActionEvent;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.stage.Stage;
import javafx.stage.Modality;
import javafx.scene.Scene;
import java.io.IOException;


/**
 * The type Student controller.
 */
public class StudentController {

    /**
     * Handle add student.
     *
     * @param name  the name
     * @param ID    the id
     * @param age   the age
     * @param grade the grade
     */
// Method to handle adding a student
    public void handleAddStudent(String name, String ID, int age, String grade) {
        StudentManagement.addStudent(name, ID, age, grade);
        return;
    }

    /**
     * Handle update student.
     *
     * @param ID       the id
```

```
37       * @param newName  the new name
38       * @param newAge   the new age
39       * @param newGrade the new grade
40       */
41  // Method to handle updating a student
42      public void handleUpdateStudent(String ID,
    String newName, int newAge, String newGrade) {
43          StudentManagement.updateStudent(ID, newName
    , newAge, newGrade);
44          return;
45      }
46
47      /**
48       * Handle view student student.
49       *
50       * @param ID the id
51       * @return the student
52       */
53  // Method to handle viewing a student
54      public Student handleViewStudent(String ID) {
55          return StudentManagement.getStudentDetails(
    ID);
56      }
57
58      /**
59       * Handle add student button action.
60       */
61      public void handleAddStudentButtonAction() {
62          try {
63              FXMLLoader loader = new FXMLLoader();
64              loader.setLocation(getClass().
    getResource("/AddStudent.fxml"));
65              Parent dialogRoot = loader.load();
66              Stage dialogStage = new Stage();
67              dialogStage.setTitle("Add Student");
68              dialogStage.initModality(Modality.
    WINDOW_MODAL);
69              Scene dialogScene = new Scene(
    dialogRoot);
70              dialogStage.setScene(dialogScene);
71              dialogStage.showAndWait();
```

```
72          } catch (IOException e) {
73              e.printStackTrace();
74              // Add error handling here (e.g., show
    an error dialog)
75          }
76      }
77
78 }
79
```

```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.model.Course;
import com.example.studentmanagementsystem.model.CourseManagement;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.TextField;
import javafx.stage.Stage;

/**
 * The type Add course controller.
 */
public class AddCourseController {

    @FXML
    private TextField courseNameField;

    @FXML
    private TextField courseIDField;

    private Stage dialogStage;

    /**
     * Sets dialog stage.
     *
     * @param dialogStage the dialog stage
     */
    public void setDialogStage(Stage dialogStage) {
        this.dialogStage = dialogStage;
    }

    @FXML
    private void handleSave() {
        String courseName = courseNameField.getText();
        String courseID = courseIDField.getText();

        if (courseName != null && !courseName.trim
```

```
37 ().isEmpty() && courseID != null && !courseID.trim
   ().isEmpty()) {
38             // Add the course using
   CourseManagement
39             CourseManagement.addCourse(courseName,
   courseID);
40
41             // Show success alert
42             showAlert(Alert.AlertType.INFORMATION,
   "Add Course Successful", "Course has been
   successfully added.");
43
44             // Close the dialog box after addition
45             closeCurrentWindow();
46
47         } else {
48             // Show error alert
49             showAlert(Alert.AlertType.ERROR, "Add
   Course Error", "Failed to add the course. Please
   ensure all fields are correctly filled.");
50         }
51     }
52
53     @FXML
54     private void handleCancel() {
55         dialogStage.close();
56     }
57
58     private void showAlert(Alert.AlertType type,
   String title, String message) {
59         Alert alert = new Alert(type);
60         alert.setTitle(title);
61         alert.setContentText(message);
62         alert.showAndWait();  // This makes the
   alert modal
63     }
64
65     private void closeCurrentWindow() {
66         if (dialogStage != null) {
67             dialogStage.close();
68         } else {
```

```
69                 System.err.println("Dialog stage is
   not initialized.");
70             }
71         }
72 }
73
```

```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.model.Student;
import com.example.studentmanagementsystem.model.StudentManagement;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.TextField;
import javafx.stage.Stage;

/**
 * The type Add student controller.
 */
public class AddStudentController {

    @FXML
    private TextField nameField;

    @FXML
    private TextField idField;

    @FXML
    private TextField ageField;

    @FXML
    private TextField gradeField;

    private Stage dialogStage;
    private MainController mainController;

    /**
     * Sets main controller.
     *
     * @param mainController the main controller
     */
    public void setMainController(MainController mainController) {
        this.mainController = mainController;
    }
```

```
38
39      @FXML
40      private void handleSave() {
41          String name = nameField.getText();
42          String id = idField.getText();
43          String ageStr = ageField.getText();
44          String grade = gradeField.getText();
45
46          if (name != null && !name.trim().isEmpty
    () &&
47                  id != null && !id.trim().isEmpty
    () &&
48                  ageStr != null && !ageStr.trim().
    isEmpty() &&
49                  grade != null && !grade.trim().
    isEmpty()) {
50
51              int age = Integer.parseInt(ageStr); //
    Might throw an exception for non-numeric input
52              Student newStudent = StudentManagement.
    addStudent(name, id, age, grade);
53
54              if (mainController != null) {
55                  mainController.addStudent(
    newStudent);
56              }
57
58              // Show success alert
59              showAlert(Alert.AlertType.INFORMATION,
    "Add Student Successful", "Student has been
    successfully added.");
60
61              // Close the dialog box after addition
62              closeCurrentWindow();
63
64          } else {
65              // Show error alert
66              showAlert(Alert.AlertType.ERROR, "Add
    Student Error", "Failed to add the student. Please
    ensure all fields are correctly filled.");
67          }
```

```
68      }
69
70      @FXML
71      private void handleCancel() {
72          dialogStage.close();
73      }
74
75      private void showAlert(Alert.AlertType type,
    String title, String message) {
76          Alert alert = new Alert(type);
77          alert.setTitle(title);
78          alert.setContentText(message);
79          alert.showAndWait();  // This makes the
    alert modal
80      }
81
82      private void closeCurrentWindow() {
83          if (dialogStage != null) {
84              dialogStage.close();
85          } else {
86              System.err.println("Dialog stage is
    not initialized.");
87          }
88      }
89
90      /**
91       * Sets dialog stage.
92       *
93       * @param dialogStage the dialog stage
94       */
95      public void setDialogStage(Stage dialogStage
    ) {
96          this.dialogStage = dialogStage;
97      }
98  }
99
```

```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.model.Course;
import com.example.studentmanagementsystem.model.CourseManagement;
import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.fxml.FXML;
import javafx.scene.control.TableColumn;
import javafx.scene.control.TableView;

/**
 * The type Course view controller.
 */
public class CourseViewController {
    @FXML
    private TableView<Course> courseTable;
    @FXML
    private TableColumn<Course, String> courseNameColumn;
    @FXML
    private TableColumn<Course, String> courseIDColumn;

    private ObservableList<Course> courseData = FXCollections.observableArrayList();

    @FXML
    private void initialize() {
        // Initialize the course table
        courseNameColumn.setCellValueFactory(cellData -> cellData.getValue().courseNameProperty());
        courseIDColumn.setCellValueFactory(cellData -> cellData.getValue().courseIDProperty());

        // Load initial course data
        courseData.addAll(CourseManagement.getCourses());
```

```
32          courseTable.setItems(courseData);
33      }
34  }
35
```

```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.model.Course;
import com.example.studentmanagementsystem.model.CourseManagement;
import com.example.studentmanagementsystem.model.Student;
import com.example.studentmanagementsystem.model.StudentManagement;
import javafx.collections.FXCollections;
import javafx.fxml.FXML;
import javafx.scene.control.Alert;
import javafx.scene.control.ComboBox;
import javafx.scene.control.TextField;

/**
 * The type Assign grade controller.
 */
public class AssignGradeController {

    @FXML
    private ComboBox<Student> studentComboBox;
    @FXML
    private ComboBox<Course> courseComboBox;
    @FXML
    private TextField gradeField;

    @FXML
    private void initialize() {
        studentComboBox.setItems(FXCollections.observableArrayList(StudentManagement.getStudents()));
        courseComboBox.setItems(FXCollections.observableArrayList(CourseManagement.getCourses()));
    }

    /**
     * Handle submit.
```

```
33       */
34      @FXML
35      public void handleSubmit() {
36          Student selectedStudent = studentComboBox.
    getSelectionModel().getSelectedItem();
37          Course selectedCourse = courseComboBox.
    getSelectionModel().getSelectedItem();
38          String grade = gradeField.getText();
39
40          if (selectedStudent != null &&
    selectedCourse != null && !grade.isEmpty()) {
41              selectedStudent.setGradeForCourse(
    selectedCourse, grade);
42              showAlert(Alert.AlertType.INFORMATION,
    "Success", "Grade has been successfully assigned."
    );
43          } else {
44              showAlert(Alert.AlertType.ERROR, "Error
    ", "Failed to assign grade. Please ensure all
    fields are correctly filled.");
45          }
46      }
47
48      private void showAlert(Alert.AlertType type,
    String title, String message) {
49          Alert alert = new Alert(type);
50          alert.setTitle(title);
51          alert.setHeaderText(null);  // No header
    text
52          alert.setContentText(message);
53          alert.showAndWait();
54      }
55  }
56
```

File - /Users/insydeowt/SMS/src/main/java/com/example/studentmanagementsystem/controller/EnrollStudentController.java

```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.model.CourseManagement;
import com.example.studentmanagementsystem.model.Student;
import com.example.studentmanagementsystem.model.Course;
import com.example.studentmanagementsystem.model.StudentManagement;
import javafx.collections.FXCollections;
import javafx.fxml.FXML;
import javafx.scene.control.ComboBox;
import javafx.scene.control.Alert;
import javafx.stage.Stage;

/**
 * The type Enroll student controller.
 */
public class EnrollStudentController {

    @FXML
    private ComboBox<Student> studentComboBox;

    @FXML
    private ComboBox<Course> courseComboBox;

    /**
     * Initialize.
     */
    @FXML
    public void initialize() {
        // Populate the studentComboBox with the list of students
        studentComboBox.setItems(FXCollections.observableArrayList(StudentManagement.getStudents()));

        // Populate the courseComboBox with the list of courses
```

```
33          courseComboBox.setItems(FXCollections.
    observableArrayList(CourseManagement.getCourses
    ()));
34      }
35
36      /**
37       * Handle enroll.
38       */
39      @FXML
40      public void handleEnroll() {
41          // logic to enroll the selected student in
    the selected course
42          // get the selected student and course
43          Student selectedStudent = studentComboBox.
    getSelectionModel().getSelectedItem();
44          Course selectedCourse = courseComboBox.
    getSelectionModel().getSelectedItem();
45
46          // enroll the student in the course
47          if (selectedStudent != null &&
    selectedCourse != null) {
48              selectedStudent.enrollInCourse(
    selectedCourse);
49
50              // Show success alert
51              showAlert(Alert.AlertType.INFORMATION,
    "Enrollment Successful", "Student has been
    successfully enrolled in the course.");
52
53              // Close the dialog box after
    enrollment
54              closeCurrentWindow();
55
56          } else {
57              // Show error alert
58              showAlert(Alert.AlertType.ERROR, "
    Enrollment Error", "Failed to enroll the student.
    Please ensure both student and course are selected
    .");
59          }
60      }
```

```
61
62      private void showAlert(Alert.AlertType type,
    String title, String message) {
63          Alert alert = new Alert(type);
64          alert.setTitle(title);
65          alert.setContentText(message);
66          alert.showAndWait();  // This makes the
    alert modal
67      }
68
69      private void closeCurrentWindow() {
70          Stage stage = (Stage) studentComboBox.
    getScene().getWindow();
71          stage.close();
72      }
73  }
74
```

```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.controller.MainController;
import com.example.studentmanagementsystem.model.Student;
import javafx.scene.control.Alert;
import javafx.scene.control.TextField;
import javafx.stage.Stage;
import javafx.fxml.FXML;

/**
 * The type Update student controller.
 */
public class UpdateStudentController {
    @FXML
    private TextField nameField;
    @FXML
    private TextField idField;
    @FXML
    private TextField ageField;

    private Student selectedStudent;
    private MainController mainController;

    /**
     * Sets main controller.
     *
     * @param mainController the main controller
     */
    public void setMainController(MainController mainController) {
        this.mainController = mainController;
    }

    /**
     * Sets selected student.
     *
     * @param student the student
     */
```

```
38      public void setSelectedStudent(Student student
    ) {
39          this.selectedStudent = student;
40          // Pre-fill the fields:
41          nameField.setText(student.getName());
42          idField.setText(student.getID());
43          ageField.setText(String.valueOf(student.
    getAge()));
44      }
45
46      /**
47       * Handle update.
48       */
49      @FXML
50      public void handleUpdate() {
51          // Get updated data from fields:
52          String updatedName = nameField.getText();
53          String updatedId = idField.getText();
54          String ageStr = ageField.getText();
55
56          if (updatedName != null && !updatedName.
    trim().isEmpty() &&
57                  updatedId != null && !updatedId.
    trim().isEmpty() &&
58                  ageStr != null && !ageStr.trim().
    isEmpty()) {
59
60              try {
61                  int updatedAge = Integer.parseInt(
    ageStr);
62                  selectedStudent.setName(updatedName
    );
63                  selectedStudent.setID(updatedId);
64                  selectedStudent.setAge(updatedAge);
65
66                  // Update the student data in the
    main controller and refresh the table:
67                  mainController.updateStudent(
    selectedStudent);
68
69                  // Show success alert
```

```
70                  showAlert(Alert.AlertType.
   INFORMATION, "Update Successful", "Student data
   has been successfully updated.");
71
72                  // Close the update form:
73                  closeCurrentWindow();
74              } catch (NumberFormatException e) {
75                  // Show error alert for invalid
   age input
76                  showAlert(Alert.AlertType.ERROR, "
   Update Error", "Invalid age input. Please enter a
   valid number.");
77              }
78
79          } else {
80              // Show error alert
81              showAlert(Alert.AlertType.ERROR, "
   Update Error", "Failed to update the student.
   Please ensure all fields are correctly filled.");
82          }
83      }
84
85      /**
86       * Handle cancel.
87       */
88      @FXML
89      public void handleCancel() {
90          // Close the update form without saving
   changes:
91          closeCurrentWindow();
92      }
93
94      private void showAlert(Alert.AlertType type,
   String title, String message) {
95          Alert alert = new Alert(type);
96          alert.setTitle(title);
97          alert.setContentText(message);
98          alert.showAndWait();  // This makes the
   alert modal
99      }
100
```

```
101     private void closeCurrentWindow() {
102         ((Stage) nameField.getScene().getWindow
    ()).close();
103     }
104 }
105
```

```java
package com.example.studentmanagementsystem.controller;

import com.example.studentmanagementsystem.model.Student;
import javafx.fxml.FXML;
import javafx.scene.control.Label;
import javafx.stage.Stage;

/**
 * The type Student details controller.
 */
public class StudentDetailsController {

    @FXML
    private Label nameLabel, idLabel, ageLabel, gradesLabel, coursesLabel;

    private Stage dialogStage;

    /**
     * Sets student details.
     *
     * @param student the student
     */
    public void setStudentDetails(Student student) {
        nameLabel.setText("Name: " + student.getName());
        idLabel.setText("ID: " + student.getID());
        ageLabel.setText("Age: " + student.getAge());

        // Display the grades and courses for the student
        gradesLabel.setText("Grades: " + student.getAllGrades().toString());
        coursesLabel.setText("Courses: " + student.getAllGrades().keySet().toString());
    }

```

```
33      @FXML
34      private void handleClose() {
35          dialogStage.close();
36      }
37
38      /**
39       * Sets dialog stage.
40       *
41       * @param dialogStage the dialog stage
42       */
43      public void setDialogStage(Stage dialogStage) {
44          this.dialogStage = dialogStage;
45      }
46
47      /**
48       * Update grade table.
49       *
50       * @param selectedStudent the selected student
51       */
52      public void updateGradeTable(Student
    selectedStudent) {
53          // update the label with the new grade
54          gradesLabel.setText("Grades: " +
    selectedStudent.getAllGrades().toString());
55      }
56  }
57
```