**Student Record Management System Documentation**

**Introduction**

The Student Record Management System helps administrators handle student records more efficiently. This documentation provides clear instructions on running the program and using the administrator interface.

**System Requirements**

- Java Development Kit (JDK) installed

- An Integrated Development Environment (IDE) like IntelliJ IDEA or Eclipse (optional but recommended)

**Running the Program**

1. Download the `Student.java`, `StudentManagement.java`, and `AdministratorInterface.java` files.

2. Open your preferred IDE and create a new Java project.

3. Import the downloaded files into your project.

4. Compile and run the `AdministratorInterface.java` file to start the program.

**Interacting with the Administrator Interface**

When you run the program, you'll see these menu options:

1. \*\*Add Student\*\*: Add a new student by entering their name, ID, age, and grade.

2. \*\*Update Student\*\*: Update a student's information by providing their ID and entering the new details.

3. \*\*View Student Details\*\*: See the details of a student by entering their ID.

4. \*\*Exit\*\*: Close the program.

To use the interface:

- Choose an option by entering the corresponding number.

- Follow the prompts to input the required information.

- Once done, you'll get a confirmation message.

‘’’

import java.util.Scanner;

public class AdministratorInterface {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

while (true) {

System.out.println("Student Record Management System");

System.out.println("1. Add Student");

System.out.println("2. Update Student");

System.out.println("3. View Student Details");

System.out.println("4. Exit");

System.out.print("Enter your choice: ");

int choice = scanner.nextInt();

switch (choice) {

case 1:

addStudent();

break;

case 2:

updateStudent();

break;

case 3:

viewStudentDetails();

break;

case 4:

System.out.println("Exiting...");

scanner.close();

System.exit(0);

default:

System.out.println("Invalid choice. Try again.");

}

}

}

private static void addStudent() {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter student name: ");

String name = scanner.nextLine();

System.out.print("Enter student ID: ");

int id = scanner.nextInt();

// Prompt for age and grade as well...

// Create new student object and add to student list

Student newStudent = new Student(name, id, age, grade);

StudentManagement.addStudent(newStudent);

System.out.println("Student added successfully!");

}

private static void updateStudent() {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter student ID to update: ");

int studentId = scanner.nextInt();

// Check if student exists

Student existingStudent = StudentManagement.getStudent(studentId);

if (existingStudent == null) {

System.out.println("Student not found.");

return;

}

System.out.print("Enter updated student name: ");

String name = scanner.nextLine();

// Prompt for updating age and grade as well...

// Create updated student object

Student updatedStudent = new Student(name, existingStudent.getId(), age, grade);

// Update student information

StudentManagement.updateStudent(studentId, updatedStudent);

System.out.println("Student information updated successfully!");

}

private static void viewStudentDetails() {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter student ID to view details: ");

int studentId = scanner.nextInt();

// Check if student exists

Student student = StudentManagement.getStudent(studentId);

if (student == null) {

System.out.println("Student not found.");

return;

}

// Display student details

System.out.println("Student Details:");

System.out.println("Name: " + student.getName());

System.out.println("ID: " + student.getId());

// Display age and grade as well...

}

}

‘’’

**Screenshots**

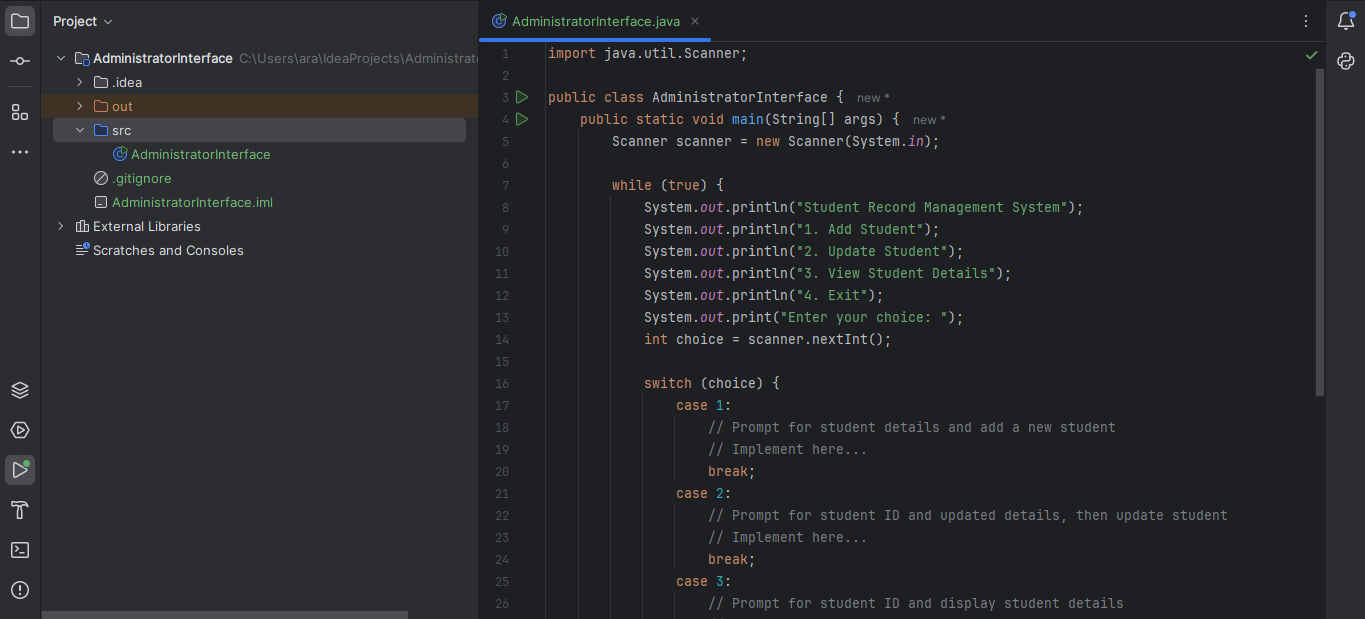


Figure 1. Screenshot depicting code in the IDE

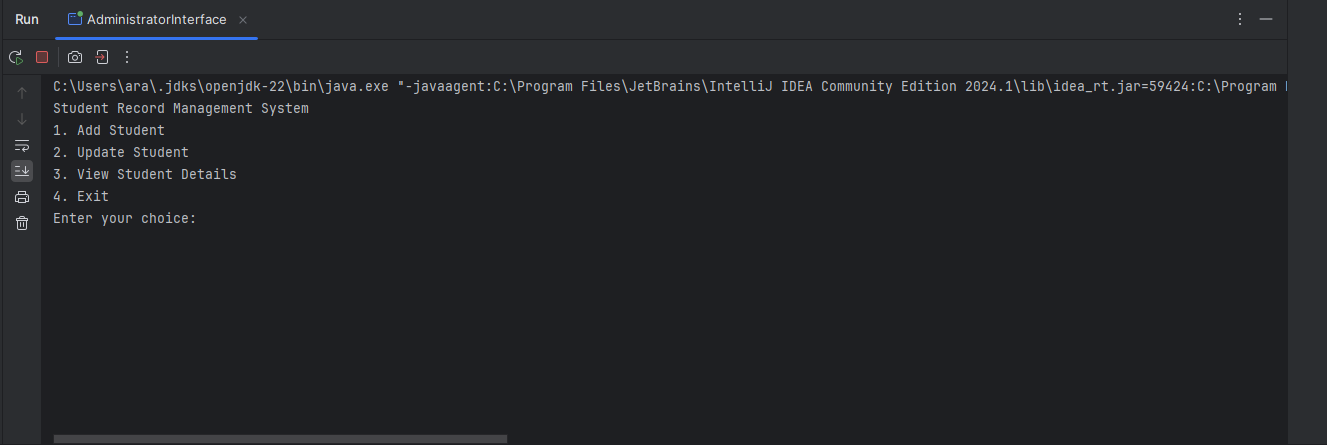


Figure 2. Terminal prompting the user for input

**Conclusion**

With this system, administrators can manage student records effectively, improving organization and efficiency.

**References:**

1. Horstmann, C. S., & Cornell, G. (2013). \*Core Java Volume I--Fundamentals\* (9th ed.). Prentice Hall.

2. Sierra, K., & Bates, B. (2014). \*Head First Java\* (2nd ed.). O'Reilly Media.

3. Eckel, B. (2006). \*Thinking in Java\* (4th ed.). Prentice Hall.