1. Student Class:

Attributes:

Student ID (unique for each student)

First Name

Last Name

Date of Birth (in the format "YYYY-MM-DD")

GPA (Grade Point Average)

import java.time.LocalDate;  
  
class Student {  
 private static int *idCounter* = 1;  
  
 private int studentId;  
 private String firstName;  
 private String lastName;  
 private String dateOfBirth;  
 private double gpa;  
  
 public Student(String firstName, String lastName, LocalDate dateOfBirth, double gpa) {  
 this.studentId = *idCounter*++;  
 this.firstName = firstName;  
 this.lastName = lastName;  
 this.dateOfBirth = String.*valueOf*(dateOfBirth);  
 this.gpa = gpa;  
 }  
  
 public static int getIdCounter() {  
 return *idCounter*;  
 }  
  
 public static void setIdCounter(int idCounter) {  
 Student.*idCounter* = idCounter;  
 }  
  
 public int getStudentId() {  
 return studentId;  
 }  
  
 public void setStudentId(int studentId) {  
 this.studentId = studentId;  
 }  
  
 public String getFirstName() {  
 return firstName;  
 }  
  
 public void setFirstName(String firstName) {  
 this.firstName = firstName;  
 }  
  
 public String getLastName() {  
 return lastName;  
 }  
  
 public void setLastName(String lastName) {  
 this.lastName = lastName;  
 }  
  
 public String getDateOfBirth() {  
 return dateOfBirth;  
 }  
  
 public void setDateOfBirth(String dateOfBirth) {  
 this.dateOfBirth = dateOfBirth;  
 }  
  
 public double getGpa() {  
 return gpa;  
 }  
  
 public void setGpa(double gpa) {  
 this.gpa = gpa;  
 }  
}

1. Menu System:

Implement a menu system that allows users to choose from the following options:

* + 1. Add a new student record
    2. Update an existing student record
    3. Display all student records
    4. Search for a student by ID
    5. Calculate and display the average GPA of all students
    6. Exit the program

import java.time.LocalDate;  
import java.time.format.DateTimeParseException;  
import java.util.ArrayList;  
import java.util.Scanner;  
  
class StudentRecordSystem {  
 private ArrayList<Student> studentList = new ArrayList<>();  
 private Scanner scanner = new Scanner(System.*in*);  
  
 public void displayMenu() {  
 System.*out*.println("1. Add a new student record");  
 System.*out*.println("2. Update an existing student record");  
 System.*out*.println("3. Display all student records");  
 System.*out*.println("4. Search for a student by ID");  
 System.*out*.println("5. Calculate and display the average GPA of all students");  
 System.*out*.println("6. Exit");  
 }  
  
 public void start() {  
 int choice;  
  
 do {  
 displayMenu();  
 System.*out*.print("Enter your choice: ");  
 choice = scanner.nextInt();  
 scanner.nextLine(); // Consume the newline character  
  
 switch (choice) {  
 case 1:  
 addStudentRecord();  
 break;  
 case 2:  
 updateStudentRecord();  
 break;  
 case 3:  
 displayAllStudentRecords();  
 break;  
 case 4:  
 searchStudentById();  
 break;  
 case 5:  
 calculateAverageGPA();  
 break;  
 case 6:  
 System.*out*.println("Exiting the program. Goodbye!");  
 break;  
 default:  
 System.*out*.println("Invalid choice. Please enter a valid option.");  
 }  
 } while (choice != 6);  
 }  
  
 private void addStudentRecord() {  
 // Implement logic to add a new student record  
 scanner.nextLine();   
 System.*out*.println("Enter first name:");  
 String firstName = scanner.nextLine();  
  
 System.*out*.println("Enter last name:");  
 String lastName = scanner.nextLine();  
  
 LocalDate dateOfBirth = null;  
 while (dateOfBirth == null) {  
 System.*out*.println("Enter date of birth (YYYY-MM-DD):");  
 String dobString = scanner.nextLine();  
 try {  
 dateOfBirth = LocalDate.*parse*(dobString);  
 } catch (DateTimeParseException e) {  
 System.*out*.println("Invalid date format. Please try again.");  
 }  
  
 double gpa = -1;  
 while (gpa < 0) {  
 System.*out*.println("Enter GPA:");  
 if (scanner.hasNextDouble()) {  
 gpa = scanner.nextDouble();  
 if (gpa < 0 || gpa > 4) {   
 // Assuming GPA is on a 0 to 4 scale  
 System.*out*.println("GPA must be between 0 and 4. Please try again.");  
 gpa = -1;  
 }  
 } else {  
 System.*out*.println("Invalid GPA. Please enter a number.");  
 scanner.next();   
 }  
 }  
  
 Student newStudent = new Student(firstName, lastName, dateOfBirth, gpa);  
 studentList.add(newStudent);  
 System.*out*.println("Student added successfully with ID: " + newStudent.getStudentId());  
 }  
 }  
  
 private void updateStudentRecord() {  
 System.*out*.print("Enter student ID to update: ");  
 int studentIdToUpdate = scanner.nextInt();  
  
 // Perform search for student by ID  
 for (Student student : studentList) {  
 if (student.getStudentId() == studentIdToUpdate) {  
 System.*out*.println("Enter new details for the student:");  
  
 System.*out*.print("First Name: ");  
 student.setFirstName(scanner.next());  
  
 System.*out*.print("Last Name: ");  
 student.setLastName(scanner.next());  
  
 System.*out*.print("Date of Birth (YYYY-MM-DD): ");  
 student.setDateOfBirth(scanner.next());  
  
 System.*out*.print("GPA: ");  
 student.setGpa(scanner.nextDouble());  
  
  
  
 System.*out*.println("Student record updated successfully.");  
 return;  
 }  
 else {  
 System.*out*.println("student not present in records");  
 }  
 }  
  
 System.*out*.println("Student with ID " + studentIdToUpdate + " not found.");  
 }  
  
 private void displayAllStudentRecords() {  
 if (studentList.isEmpty()) {  
 System.*out*.println("No student records available.");  
 } else {  
 System.*out*.println("Student Records:");  
 for (Student student : studentList) {  
 System.*out*.println(student.getStudentId() + ": " + student.getFirstName() + " " + student.getLastName());  
 }  
 }  
 }  
  
 private void searchStudentById() {  
 // Implement logic to search for a student by ID  
 System.*out*.print("Enter student ID to search: ");  
 int studentIdToSearch = scanner.nextInt();  
   
 for (Student student : studentList) {  
 if (student.getStudentId() == studentIdToSearch) {  
 System.*out*.println("Student found:");  
 System.*out*.println("ID: " + student.getStudentId());  
 System.*out*.println("Name: " + student.getFirstName() + " " + student.getLastName());  
 System.*out*.println("Date of Birth: " + student.getDateOfBirth());  
 System.*out*.println("GPA: " + student.getGpa());  
 return;  
 }  
 }  
   
 System.*out*.println("Student with ID " + studentIdToSearch + " not found.");  
 }  
  
 private void calculateAverageGPA() {  
 if (studentList.isEmpty()) {  
 System.*out*.println("No student records available.");  
 } else {  
 double totalGPA = 0.0;  
  
 for (Student student : studentList) {  
 totalGPA += student.getGpa();  
 }  
  
 double averageGPA = totalGPA / studentList.size();  
 System.*out*.println("Average GPA of all students: " + averageGPA);  
 }  
 }  
}

1. Add a new student record:

private void addStudentRecord() {  
 // Implement logic to add a new student record  
 scanner.nextLine();   
 System.*out*.println("Enter first name:");  
 String firstName = scanner.nextLine();  
  
 System.*out*.println("Enter last name:");  
 String lastName = scanner.nextLine();  
  
 LocalDate dateOfBirth = null;  
 while (dateOfBirth == null) {  
 System.*out*.println("Enter date of birth (YYYY-MM-DD):");  
 String dobString = scanner.nextLine();  
 try {  
 dateOfBirth = LocalDate.*parse*(dobString);  
 } catch (DateTimeParseException e) {  
 System.*out*.println("Invalid date format. Please try again.");  
}  
  
double gpa = -1;  
while (gpa < 0) {  
 System.*out*.println("Enter GPA:");  
 if (scanner.hasNextDouble()) {  
 gpa = scanner.nextDouble();  
 if (gpa < 0 || gpa > 4) {   
 // Assuming GPA is on a 0 to 4 scale  
 System.*out*.println("GPA must be between 0 and 4. Please try again.");  
 gpa = -1;  
 }  
 } else {  
 System.*out*.println("Invalid GPA. Please enter a number.");  
 scanner.next();   
 }  
 }  
  
 Student newStudent = new Student(firstName, lastName, dateOfBirth, gpa);  
 studentList.add(newStudent);  
 System.*out*.println("Student added successfully with ID: " + newStudent.getStudentId());  
 }  
}

2. Update an existing student record:

private void updateStudentRecord() {  
 System.*out*.print("Enter student ID to update: ");  
 int studentIdToUpdate = scanner.nextInt();  
  
 // Perform search for student by ID  
 for (Student student : studentList) {  
 if (student.getStudentId() == studentIdToUpdate) {  
 System.*out*.println("Enter new details for the student:");  
  
 System.*out*.print("First Name: ");  
 student.setFirstName(scanner.next());  
  
 System.*out*.print("Last Name: ");  
 student.setLastName(scanner.next());  
  
 System.*out*.print("Date of Birth (YYYY-MM-DD): ");  
 student.setDateOfBirth(scanner.next());  
  
 System.*out*.print("GPA: ");  
 student.setGpa(scanner.nextDouble());  
  
  
  
 System.*out*.println("Student record updated successfully.");  
 return;  
 }  
 else {  
 System.*out*.println("student not present in records");  
 }  
 }  
  
 System.*out*.println("Student with ID " + studentIdToUpdate + " not found.");  
}

1. Display all student records:

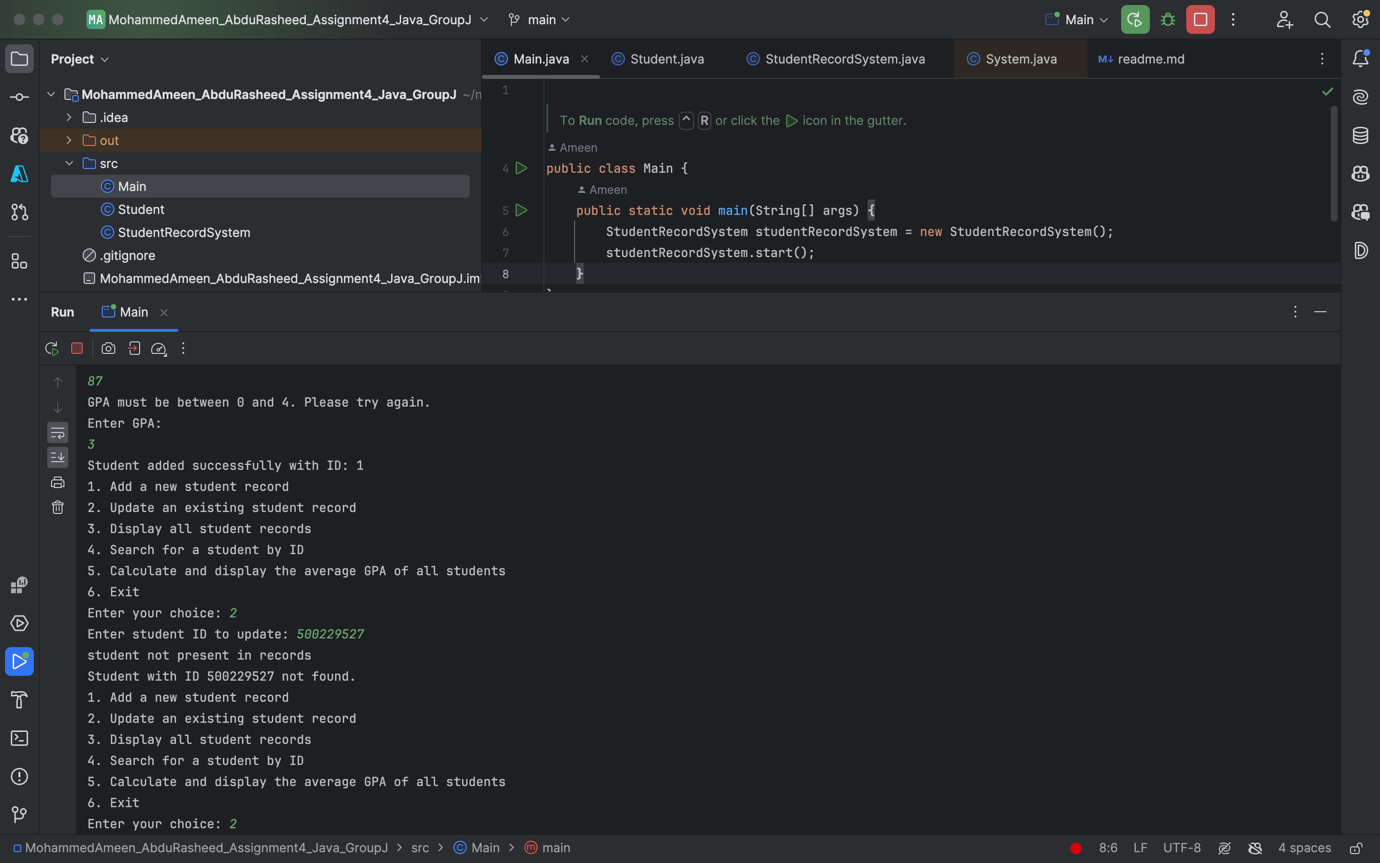
private void displayAllStudentRecords() {  
 if (studentList.isEmpty()) {  
 System.*out*.println("No student records available.");  
 } else {  
 System.*out*.println("Student Records:");  
 for (Student student : studentList) {  
 System.*out*.println(student.getStudentId() + ": " + student.getFirstName() + " " + student.getLastName());  
 }  
 }  
}

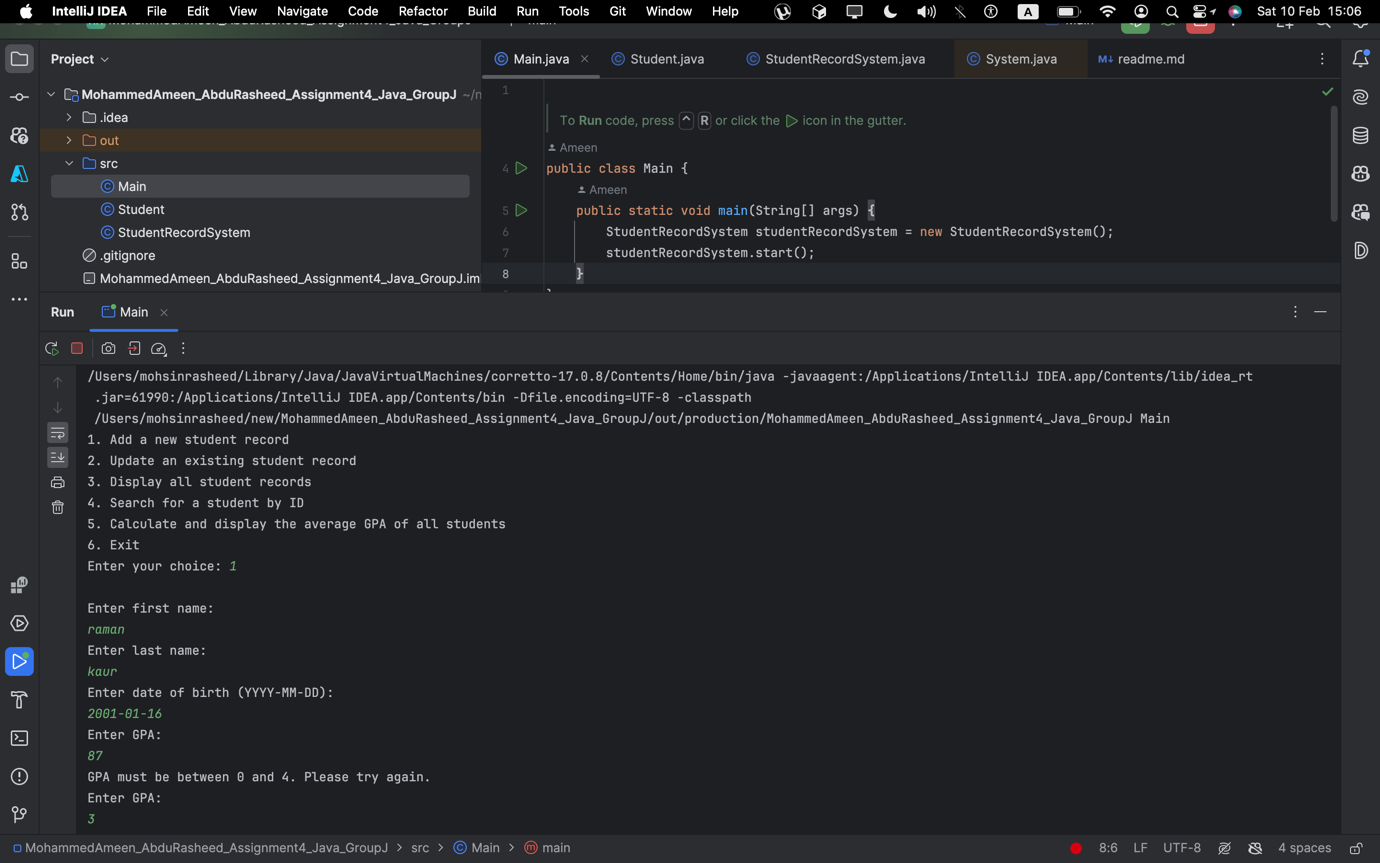
4.Search for a student by ID:

private void searchStudentById() {  
 // Implement logic to search for a student by ID  
 System.*out*.print("Enter student ID to search: ");  
 int studentIdToSearch = scanner.nextInt();  
  
 for (Student student : studentList) {  
 if (student.getStudentId() == studentIdToSearch) {  
 System.*out*.println("Student found:");  
 System.*out*.println("ID: " + student.getStudentId());  
 System.*out*.println("Name: " + student.getFirstName() + " " + student.getLastName());  
 System.*out*.println("Date of Birth: " + student.getDateOfBirth());  
 System.*out*.println("GPA: " + student.getGpa());  
 return;  
 }  
 }  
  
 System.*out*.println("Student with ID " + studentIdToSearch + " not found.");  
}

5. Calculate and display the average GPA of all students:

private void calculateAverageGPA() {  
 if (studentList.isEmpty()) {  
 System.*out*.println("No student records available.");  
 } else {  
 double totalGPA = 0.0;  
  
 for (Student student : studentList) {  
 totalGPA += student.getGpa();  
 }  
  
 double averageGPA = totalGPA / studentList.size();  
 System.*out*.println("Average GPA of all students: " + averageGPA);  
 }  
}





A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

Contributions:

Group members contributions can be seen from GitHub and Jira

https://github.com/ameyster1999/MohammedAmeen\_AbduRasheed\_Assignment4\_Java\_GroupJ

GitHub:

A screenshot of a computer

Description automatically generated

Jira:

A screenshot of a computer

Description automatically generated

Commit history:

A screenshot of a computer

Description automatically generated  
SubmittedBy:  
Ramandeep Kaur(500229527)-ramandeepkaur105@loyalistcollege.com  
Hammad Ul Hassan(500230292)-hammadulhassan@loyalistcollege.com  
Mohammed Ameen Abdu Rasheed(500225970)- mohammedameenabdu@loyalistcollege.com  
Syed Muhammed Fakhr e Sajjad(500217679)- @loyalistcollege.com  
Ramanpreet Kaur(500218959)-ramanpreetkaur29@loyalistcollege.com  
Sindhuja Periavalli(500228575)- sindhujaperavali21@gmail.com