Student Record Management System Documentation

1. Student Class

The `Student` class encapsulates student information with private instance variables:

- `id`: Student ID
- `name`: Student name
- `age`: Student age
- `grade`: Student grade

Constructor

```
javaCopy code
public Student(String id, String name, int age, String grade)
```

Getter Methods

- `getId()`: Returns the student ID.
- `getName()`: Returns the student name.
- `getAge()`: Returns the student age.
- `getGrade()`: Returns the student grade.

Setter Methods

- `setName(String name)`: Sets the student name.
- `setAge(int age)`: Sets the student age.
- `setGrade(String grade)`: Sets the student grade.

Other Methods

- `toString()`: Returns a string representation of the student details.

2. StudentManagement Class

The `StudentManagement` class manages a list of students and provides methods for Administrators.

Private Static Variables

- `students`: ArrayList to store student objects.
- `totalStudents`: Total number of students.

Methods

- `addStudent(String id, String name, int age, String grade)`: Adds a new student to the list.
- `updateStudent(String id, String name, int age, String grade)`: Updates student information.
- `viewStudent(String id)`: Displays details of a specific student.
- `displayTotalStudents()`: Displays the total number of students.

3. Administrator Interface

The `StudentRecordManagementSystem` class serves as the entry point with a menu-driven Interface.

Main Method

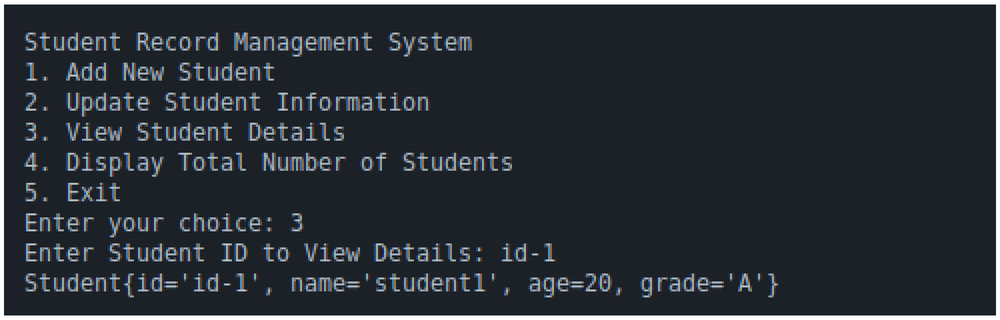
The main method presents options:

1. Add New Student
2. Update Student Information
3. View Student Details
4. Display Total Number of Students
5. Exit

Example Usage

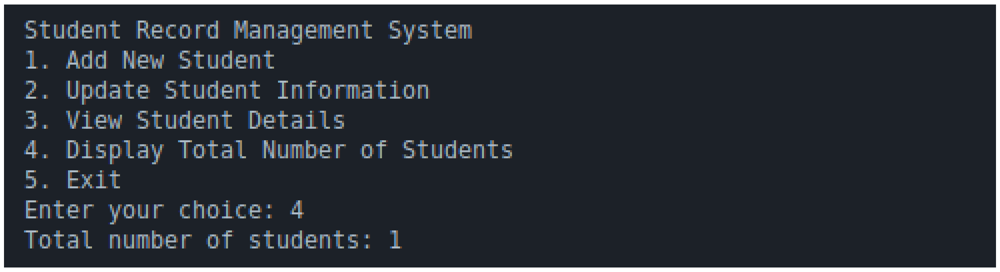
Adding a New Student:

Viewing Student Details:



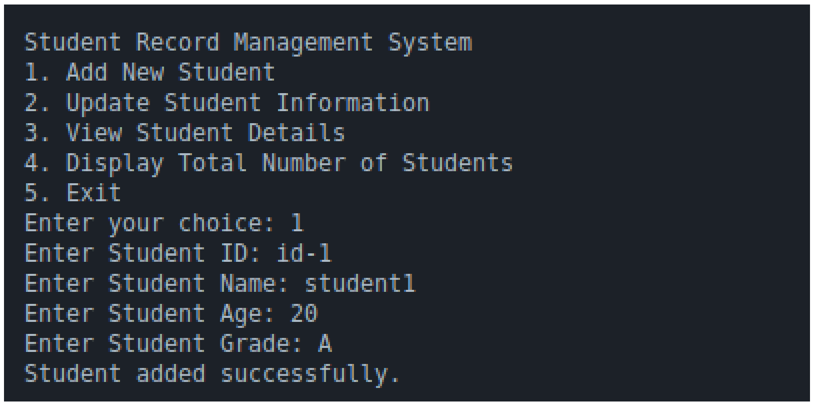
```
Student Record Management System
1. Add New Student
2. Update Student Information
3. View Student Details
4. Display Total Number of Students
5. Exit
Enter your choice: 3
Enter Student ID to View Details: id-1
Student{id='id-1', name='student1', age=20, grade='A'}
```

Displaying Total Number of Students:

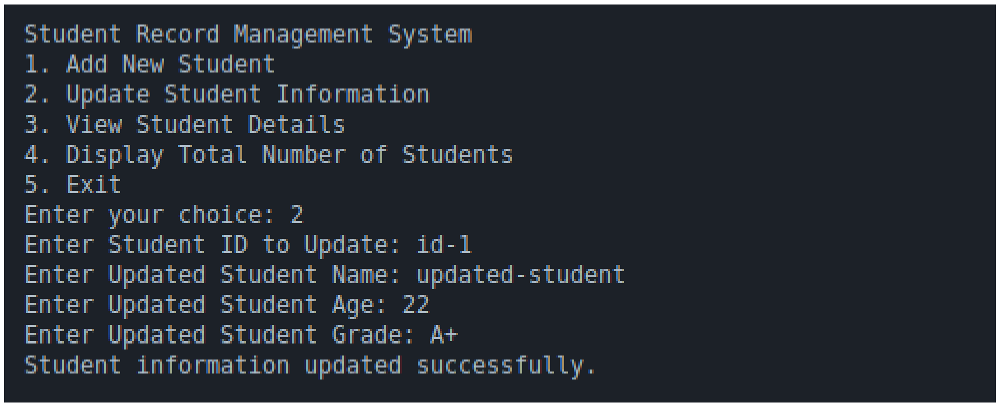


```
Student Record Management System
1. Add New Student
2. Update Student Information
3. View Student Details
4. Display Total Number of Students
5. Exit
Enter your choice: 4
Total number of students: 1
```

Updating Student Information:

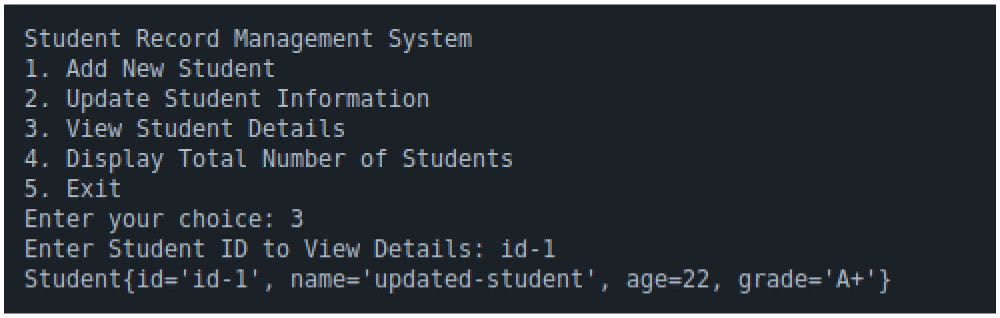


```
Student Record Management System
1. Add New Student
2. Update Student Information
3. View Student Details
4. Display Total Number of Students
5. Exit
Enter your choice: 1
Enter Student ID: id-1
Enter Student Name: student1
Enter Student Age: 20
Enter Student Grade: A
Student added successfully.
```



```
Student Record Management System
1. Add New Student
2. Update Student Information
3. View Student Details
4. Display Total Number of Students
5. Exit
Enter your choice: 2
Enter Student ID to Update: id-1
Enter Updated Student Name: updated-student
Enter Updated Student Age: 22
Enter Updated Student Grade: A+
Student information updated successfully.
```

Viewing Updated Student Details:



```
Student Record Management System
1. Add New Student
2. Update Student Information
3. View Student Details
4. Display Total Number of Students
5. Exit
Enter your choice: 3
Enter Student ID to View Details: id-1
Student{id='id-1', name='updated-student', age=22, grade='A+'}
```

Error Handling

- The program handles cases where the student ID is not found or invalid inputs are provided.

Running the Program

1. Compile the Java file: `javac StudentRecordManagementSystem.java`
2. Run the program: `java StudentRecordManagementSystem`

~~~ java

````java
import java.util.ArrayList;
import java.util.Scanner;

class Student {
   private String id;
   private String name;
   private int age;
   private String grade;

   // Constructor
   public Student(String id, String name, int age, String grade) {
       this.id = id;
       this.name = name;
       this.age = age;
       this.grade = grade;
   }

   // Getter methods
   public String getId() {
       return id;
   }

   public String getName() {
       return name;
   }

   public int getAge() {
       return age;
   }

   public String getGrade() {
       return grade;
   }

   // Setter methods
   public void setName(String name) {
~~~

```java
        this.name = name;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public void setGrade(String grade) {
        this.grade = grade;
    }

    // toString method to display student details
    @Override
    public String toString() {
        return "Student{" +
                "id='" + id + '\'' +
                ", name='" + name + '\'' +
                ", age=" + age +
                ", grade='" + grade + '\'' +
                '}';
    }
}

class StudentManagement {
    private static ArrayList<Student> students = new ArrayList<>();
    private static int totalStudents = 0;

    // Method to add a new student
    public static void addStudent(String id, String name, int age, String
grade) {
        Student newStudent = new Student(id, name, age, grade);
        students.add(newStudent);
        totalStudents++;
        System.out.println("Student added successfully.");
    }

    // Method to update student information
    public static void updateStudent(String id, String name, int age,
String grade) {
        for (Student student : students) {
            if (student.getId().equals(id)) {
                student.setName(name);
                student.setAge(age);
                student.setGrade(grade);
                System.out.println("Student information updated
successfully.");
                return;
            }
        }
        System.out.println("Student ID not found.");
    }

    // Method to view student details
    public static void viewStudent(String id) {
        for (Student student : students) {
            if (student.getId().equals(id)) {
                System.out.println(student);
                return;
            }
        }
        System.out.println("Student ID not found.");
    }

    // Method to display the total number of students
    public static void displayTotalStudents() {
        System.out.println("Total number of students: " + totalStudents);
    }
}

public class StudentRecordManagementSystem {
    public static void main(String[] args) {
        try (Scanner scanner = new Scanner(System.in)) {
            while (true) {
                System.out.println("\nStudent Record Management System");
                System.out.println("1. Add New Student");
                System.out.println("2. Update Student Information");
                System.out.println("3. View Student Details");
                System.out.println("4. Display Total Number of Students");
                System.out.println("5. Exit");
                System.out.print("Enter your choice: ");
                int choice = scanner.nextInt();
                scanner.nextLine(); // Consume the newline character

                switch (choice) {
                    case 1:
                        System.out.print("Enter Student ID: ");
                        String id = scanner.nextLine();
                        System.out.print("Enter Student Name: ");
                        String name = scanner.nextLine();
                        System.out.print("Enter Student Age: ");
                        int age = scanner.nextInt();
                        scanner.nextLine(); // Consume the newline
character
                        System.out.print("Enter Student Grade: ");
                        String grade = scanner.nextLine();
                        StudentManagement.addStudent(id, name, age, grade);
                        break;

                    case 2:
                        System.out.print("Enter Student ID to Update: ");
                        String updateId = scanner.nextLine();
                        System.out.print("Enter Updated Student Name: ");
                        String updatedName = scanner.nextLine();
                        System.out.print("Enter Updated Student Age: ");
                        int updatedAge = scanner.nextInt();
                        scanner.nextLine(); // Consume the newline
character
                        System.out.print("Enter Updated Student Grade: ");
                        String updatedGrade = scanner.nextLine();
                        StudentManagement.updateStudent(updateId,
updatedName, updatedAge, updatedGrade);
                        break;

                    case 3:
                        System.out.print("Enter Student ID to View Details:
");
                        String viewId = scanner.nextLine();
                        StudentManagement.viewStudent(viewId);
                        break;
```

```java
                    case 4:
                        StudentManagement.displayTotalStudents();
                        break;

                    case 5:
                        System.out.println("Exiting Program. Goodbye!");
                        System.exit(0);

                    default:
                        System.out.println("Invalid choice. Please enter a 
number between 1 and 5.");
                }
            }
        }
    }
}
```

```
```