**Student Record Management System Documentation**

**1. Introduction**

The Student Record Management System (SRMS) is a Java programming based application designed for colleges and universities to efficiently manage their student records efficiently. It allows administrators to perform tasks such as adding new students, updating student information, and viewing student details. This documentation provides a comprehensive overview of the project, including class and method descriptions, access modifiers, and instructions for running and interacting with the program.

**2. Class and Method Documentation**

**2.1. Student Class**

The **Student** class represents a student entity with the following private instance variable:

* **private String StudentName**: Store the name of the student.
* **private int StudentID**: Store the unique ID of the student.
* **private int StudentAge**: Store the age of the student.
* **private String StudentGrade**: Store the grade of the student.

Methods:

* **public Student (String StudentName, int StudentID, int StudentAge, String StudentGrade)**: Constructor method to initialize a new student object with the provided information.
* Getter and Setter methods are provided for accessing and updating student information. These include **getStudentName ()**, **setStudentName (String StudentName)**, **getStudentID ()**, **getStudentAge ()**, **setStudentAge (int age)**, **getStudentGrade ()**, and **setStudentGrade (String grade)**.

**2.2. Student Management Class**

The **StudentManagement** class is responsible for managing the list of students and their information. It includes the following private static variables:

* **private static List<Student> studentList**: A list to store student objects.
* **private static int TotalStudentNumber:** A count of the total number of students.

Methods:

* **public static void addStudent(String StudentName, int StudentID, int StudentAge, double StudentGrade)**: Adds a new student to the **studentList** with the provided information and updates the **totalStudents** count.
* **public static void updateStudent(int StudentID, String StudentName, int StudentAge, double StudentGrade)**: Updates the information of an existing student identified by their ID.
* **public static Student getStudentDetails(int StudentID)**: Retrieves and returns the details of a student based on their ID.
* **public static int getTotalStudents()**: Getter method to retrieve the total number of students.

**2.3. Administrator Class**

The **Administrator** class provides a command-line interface for administrators to interact with the system. It displays a menu with options to add a new student, update student information, view student details, and exit the program.

**3. Access Modifiers**

Access modifiers are used in this project to control access to class members (variables and methods). The following access modifiers are used:

* **public**: Indicates that a member is accessible from any other class.
* **private**: Restricts access to the member only within the class it is declared in.

These modifiers are crucial for maintaining data encapsulation and ensuring that sensitive information is not accessed or modified unintentionally.

**4. Interacting with the Administrator Interface**

Upon running the program, you will be presented with a menu-driven interface. Follow the on-screen prompts to perform the following tasks:

* Add a new student by selecting option 1 and providing student information.
* Update student information by selecting option 2 and providing the student's ID and new information.
* View student details by selecting option 3 and providing the student's ID.
* Check the total number of registered student by selecting option 4.
* Exit the program by selecting option 5.

**Administrator Class:**

public class Administrator {

  public static void main(String[] args) {

    CommonFunctions fn = new CommonFunctions();

    int choice;

    do {

      System.out.println(

          "\n\n\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Student Record Management System   \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*############");

      System.out.println(

          "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*1. Add Student                     \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*############");

      System.out.println(

          "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*2. Update Student                  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*############");

      System.out.println(

          "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*3. View Student Details            \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*############");

          System.out.println(

          "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*4. Check Total Student             \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*############");

      System.out.println(

          "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*5. Exit                            \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*############");

      // System.out.print("\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enter your choice: \t");

      choice = fn.getNumberFromUser(

          "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Enter your choice: \t");

      switch (choice) {

        case 1:

          // Add Student

          String name = fn.getStringFromUser("Enter student name: ");

          int id = fn.getNumberFromUser("Enter student ID: ");

          int age = fn.getNumberFromUser("Enter student age: ");

          String grade = fn.getStringFromUser("Enter student grade: ");

          Student stdExist =  SMS.getStudentDetails(id);

          if(stdExist == null){

            SMS.addStudent(name, id, age, grade);

            System.out.println(

              "\n\t\t\t###\*\*\*\*\*\*\*  Student added successfully.\n");

          }else{

            System.out.println(

              "\n\t\t\t Error! Student Already Exists.\n");

          }

          break;

        case 2:

          // Update Student

          int stdID = fn.getNumberFromUser("Enter student ID to update: ");

          Student std = SMS.getStudentDetails(stdID);

          if (std != null) {

            String newName = fn.getStringFromUser("Enter new student name: ");

            int newAge = fn.getNumberFromUser("Enter new student age: ");

            String newGrade = fn.getStringFromUser("Enter new student grade: ");

            SMS.updateStudent(stdID, newName, newAge, newGrade);

            System.out.println(

                "\n\t\t\t###\*\*\*\*\*\*\*  Student updated successfully.\n");

          } else {

            System.out.println(

                "\n\t\t Student Record not found with ID: " + stdID + " \n");

          }

          break;

        case 3:

          // View Student Details

          stdID = fn.getNumberFromUser("Enter student ID to view details: ");

          std = SMS.getStudentDetails(stdID);

          if (std != null) {

            System.out.println(

                "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Student Detail  :");

            System.out.println(

                "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Name              : " +

                    std.getStudentName());

            System.out.println(

                "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  ID                : " +

                    std.getStudentID());

            System.out.println(

                "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Age               : " +

                    std.getStudentAge());

            System.out.println(

                "\t\t\t\t############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Grade             : " +

                    std.getStudentGrade());

          } else {

            System.out.println("Student Record not found with ID: " + stdID);

          }

          break;

        case 4:

          System.out.println("Total number of Student = "+SMS.getTotalStudents());

          break;

        case 5:

          System.out.println("############\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*    Quiting........");

          break;

        default:

          System.out.println("Invalid choice! Please try again.");

      }

    } while (choice != 5);

  }

}

**Student Class:**

public class Student {

    private String StudentName;

    private int StudentID;

    private int StudentAge;

    private String StudentGrade;

    public Student(String StudentName, int StudentID, int StudentAge, String StudentGrade) {

        this.StudentName = StudentName;

        this.StudentID = StudentID;

        this.StudentAge = StudentAge;

        this.StudentGrade = StudentGrade;

    }

    // Getter and Setter methods for all instance variables

    public String getStudentName () {

        return StudentName;

    }

    public void setStudentName ( String StudentName) {

        this.StudentName = StudentName;

    }

    public int getStudentID () {

        return StudentID;

    }

    public int getStudentAge () {

        return StudentAge;

    }

    public void setStudentAge ( int StudentAge) {

        this.StudentAge = StudentAge;

    }

    public String getStudentGrade () {

        return StudentGrade;

    }

    public void setStudentGrade ( String StudentGrade) {

        this.StudentGrade = StudentGrade;

    }

}

**SMS (Student Management System) Class:**

import java.util.ArrayList;

import java.util.List;

public class SMS {

    private static List<Student> studentList = new ArrayList<>();

    private static int TotalStudentNumber = 0;

    // Method to add a new student to the list

    public static void addStudent(String StudentName, int StudentID, int StudentAge, String StudentGrade) {

        Student std = new Student(StudentName, StudentID, StudentAge, StudentGrade);

        studentList.add(std);

        TotalStudentNumber++;

    }

    // Method to update student information

    public static void updateStudent(int StudentID, String StudentName, int StudentAge, String StudentGrade) {

        for (Student std : studentList) {

            if (std.getStudentID() == StudentID) {

                std.setStudentName(StudentName);

                std.setStudentAge(StudentAge);

                std.setStudentGrade(StudentGrade);

                return;

            }

        }

    }

    // Method to get student details by ID

    public static Student getStudentDetails(int StudentID) {

        for (Student std : studentList) {

            if (std.getStudentID() == StudentID) {

                return std;

            }

        }

        return null; // Return null if student not found

    }

    // Getter method for TotalStudentNumber

    public static int getTotalStudents() {

        return TotalStudentNumber;

    }

}

**Common Class:**

import java.util.Scanner;

public class CommonFunctions {

    private static Scanner scan = new Scanner(System.in);

    int getNumberFromUser(String message) {

        System.out.print(message);

        while (!scan.hasNextInt()) {

            System.out.print("Invalid! Please Enter any Number: ");

            scan.nextLine();

          }

          int number = scan.nextInt();

        scan.nextLine(); // Consume the newline character

        return number;

    }

    String getStringFromUser(String message) {

        System.out.print(message);

        try {

            return scan.nextLine();

        } catch(Exception e){

            System.err.println("Invalid input: ");

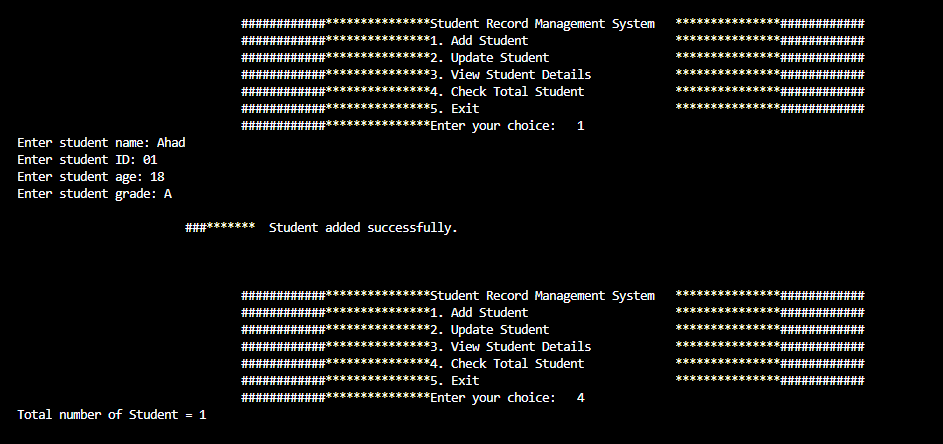
            return getStringFromUser(message);

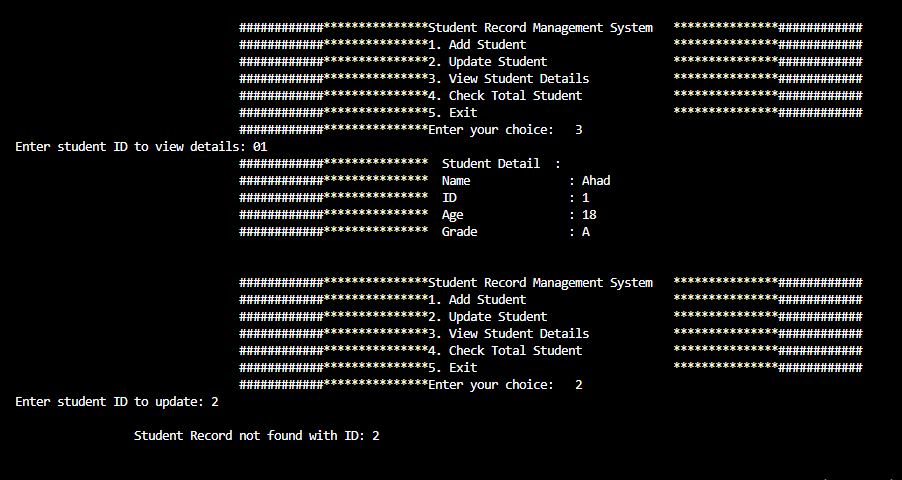
        }

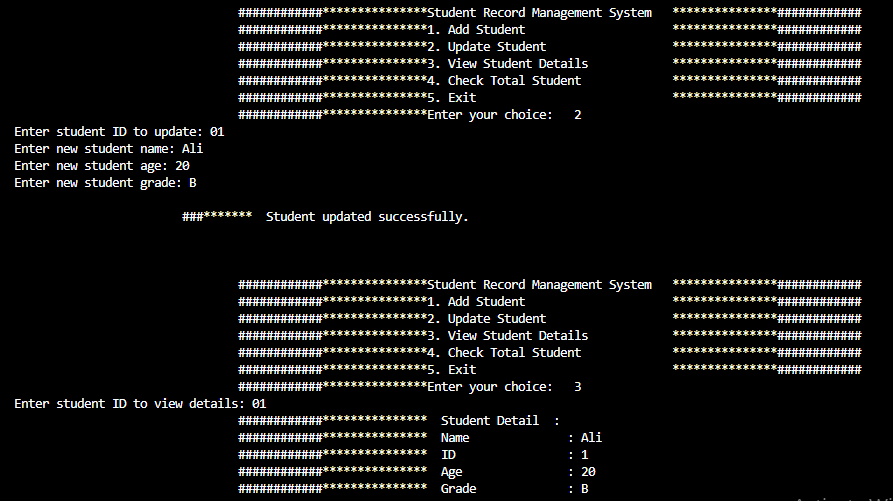
    }

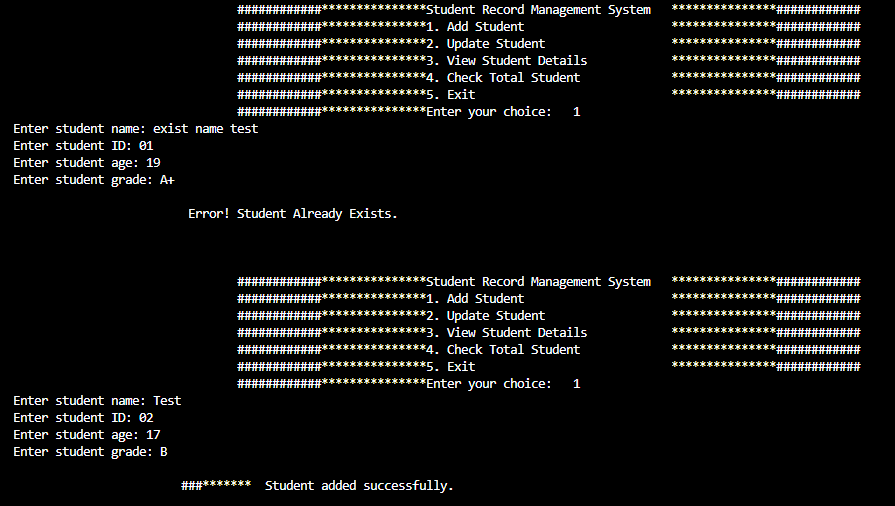
}

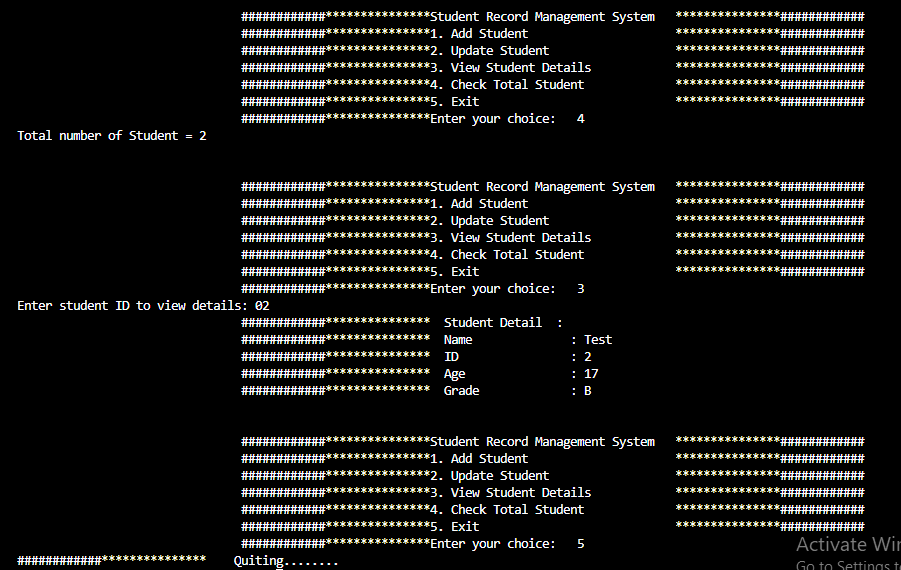
**Output:**











**The End**