```
import java.util.*;

public class App {
   static Scanner scanner = new Scanner(System.in);
   static HashMap<String, ArrayList<Double>> students = new HashMap<>();
   static HashMap<String, String> notes = new HashMap<>(); // Stores optional notes about
students

   public static void main(String[] args) {
      System.out.println("Grade Book");
      System.out.println("Press Enter to continue...");
      scanner.nextLine();

      clearConsole();

      while (true) {
         System.out.println("\nWelcome to the Grade Book!");
         System.out.println("Choose an action:");
         System.out.println("1. Add Student");
         System.out.println("2. View Student Details");
         System.out.println("3. Edit Student (Name, Grades, Notes)");
         System.out.println("4. Delete Student");
         System.out.println("5. Calculate GPA (Student & Class)");
         System.out.println("6. Exit");

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

```
int choice = scanner.nextInt();
scanner.nextLine(); // Consume newline

switch (choice) {
    case 1:
        addStudent();
        break;
    case 2:
        viewStudent();
        break;
    case 3:
        editStudent();
        break;
    case 4:
        deleteStudent();
        break;
    case 5:
        calculateAverage();
        break;
    case 6:
        System.out.println("Exiting program...");
        scanner.close();
        return;
    default:
        System.out.println("Invalid choice. Try again.");
}
```

```
    }
}

// Clears console for better UI
public static void clearConsole() {
    try {
        if (System.getProperty("os.name").contains("Windows")) {
            new ProcessBuilder("cmd", "/c", "cls").inheritIO().start().waitFor();
        } else {
            new ProcessBuilder("clear").inheritIO().start().waitFor();
        }
    } catch (Exception e) {
        System.out.println("Error clearing console.");
    }
}

// Adds a new student with optional grades
public static void addStudent() {
    System.out.print("Enter student name: ");
    String name = scanner.nextLine();
    if (students.containsKey(name)) {
        System.out.println(name + " already exists!");
        return;
    }

    students.put(name, new ArrayList<>());
```

```
    System.out.println(name + " has been added.");
}

// View student grades & notes
public static void viewStudent() {
    System.out.print("Enter student name: ");
    String name = scanner.nextLine();

    if (!students.containsKey(name)) {
        System.out.println(name + " was not found.");
        return;
    }

    ArrayList<Double> grades = students.get(name);
    System.out.println("\nStudent: " + name);
    System.out.println("Grades: " + (grades.isEmpty() ? "No grades available" : grades));
    System.out.println("Notes: " + notes.getOrDefault(name, "No notes available"));
}

// Edit student (change name, add/delete grades, edit notes)
public static void editStudent() {
    System.out.print("Enter student name: ");
    String name = scanner.nextLine();

    if (!students.containsKey(name)) {
        System.out.println(name + " was not found.");
```

```
    return;
}

System.out.println("\n1. Change name");
System.out.println("2. Add grade");
System.out.println("3. Remove grade");
System.out.println("4. Add/Edit notes");
System.out.print("Choose an option: ");
int choice = scanner.nextInt();
scanner.nextLine(); // Consume newline

switch (choice) {
    case 1:
        System.out.print("Enter new name: ");
        String newName = scanner.nextLine();
        students.put(newName, students.remove(name)); // Move data to new key
        notes.put(newName, notes.remove(name));
        System.out.println(name + " has been renamed to " + newName);
        break;

    case 2:
        System.out.print("Enter grade to add (0.0 - 4.0 scale): ");
        double grade = scanner.nextDouble();
        if (grade < 0.0 || grade > 4.0) {
            System.out.println("Invalid grade. Must be between 0.0 and 4.0.");
        } else {
```

```
            students.get(name).add(grade);
            System.out.println("Added grade " + grade + " to " + name);
        }
        break;

    case 3:
        System.out.println("Current grades: " + students.get(name));
        System.out.print("Enter grade to remove: ");
        double removeGrade = scanner.nextDouble();
        if (students.get(name).remove(removeGrade)) {
            System.out.println(removeGrade + " removed from " + name);
        } else {
            System.out.println(removeGrade + " not found.");
        }
        break;

    case 4:
        System.out.print("Enter new note for " + name + ": ");
        String note = scanner.nextLine();
        notes.put(name, note);
        System.out.println("Note updated.");
        break;

    default:
        System.out.println("Invalid option.");
}
```

```
}

// Deletes a student
public static void deleteStudent() {
    System.out.print("Enter student name to delete: ");
    String name = scanner.nextLine();

    if (students.remove(name) != null) {
        notes.remove(name);
        System.out.println(name + " has been removed.");
    } else {
        System.out.println(name + " was not found.");
    }
}

// Calculate individual & class GPA
public static void calculateAverage() {
    System.out.println("\n1. Individual GPA");
    System.out.println("2. Class GPA");
    System.out.print("Choose an option: ");
    int choice = scanner.nextInt();
    scanner.nextLine(); // Consume newline

    if (choice == 1) {
        System.out.print("Enter student name: ");
        String name = scanner.nextLine();
```

```
        if (!students.containsKey(name)) {
            System.out.println(name + " was not found.");
            return;
        }

        ArrayList<Double> grades = students.get(name);
        if (grades.isEmpty()) {
            System.out.println(name + " has no grades.");
        } else {
            double avg =
grades.stream().mapToDouble(Double::doubleValue).average().orElse(0.0);
            System.out.println(name + "'s GPA: " + avg);
        }
    } else if (choice == 2) {
        if (students.isEmpty()) {
            System.out.println("No students in the class.");
            return;
        }

        double totalGPA = 0;
        int studentCount = 0;

        for (ArrayList<Double> grades : students.values()) {
            if (!grades.isEmpty()) {
```

```
                totalGPA +=
grades.stream().mapToDouble(Double::doubleValue).average().orElse(0.0);
                studentCount++;
            }
        }

        if (studentCount == 0) {
            System.out.println("No grades entered yet.");
        } else {
            System.out.println("Class Average GPA: " + (totalGPA / studentCount));
        }
    } else {
        System.out.println("Invalid choice.");
    }
  }
}
```