PROJECT REPORT

CSE2006 - Smart Academic Planner & Grade Predictor

1. Introduction

The Smart Academic Planner & Grade Predictor is a Java-based console application developed as part of the **CSE2006** course. The system assists students in managing their academic progress by tracking multiple courses, their corresponding assessments, and predicting future performance based on target grades.

Students often struggle to keep track of assignments, tests, and final exams across multiple subjects. This project provides an efficient and user-friendly solution to simplify academic tracking and grade forecasting using basic Object-Oriented Programming (OOP) concepts.

---

2. Problem Statement

Students face difficulties in monitoring their academic progress, calculating grades, and determining the marks required to reach their desired targets. Without a proper system, it is easy to lose track of scores and weightages. Therefore, a simple tool is needed to calculate ongoing performance and forecast the effort required in future assessments.

---

3. Objectives

- Enable students to **add and manage multiple courses**.
- Allow **assessment management** (quizzes, assignments, exams) with weightages.
- Compute **current grades** based on completed assessments.
- Predict **required marks** to achieve a target percentage.
- Provide a **summary report** of overall semester performance.

---

4. Scope

This project is designed for individual student use. It supports multiple courses and assessments in a console-based Java application. It focuses on academic tracking and grade prediction without requiring a database or graphical user interface.

---

5. System Requirements

Hardware:

- Minimum: Dual-core processor, 2 GB RAM
- Recommended: Quad-core processor, 4 GB RAM

Software:

- Java JDK 17 or above
- Any IDE or Online Java Compiler
- GitHub repository for version control and submission

---

6. Functional Requirements

1. **Course Management**
   - Add new courses
   - List and delete existing courses
2. **Assessment Management**
   - Add, list, and update assessments per course
   - Each assessment includes: name, max marks, weight percentage, scored marks
3. **Grade Calculation**
   - Compute current weighted grades
   - Display completed assessment weights and total progress
4. **Goal Prediction**
   - Input target grade percentage
   - System calculates average marks needed in remaining assessments
5. **Semester Summary**
   - Display grades and completion percentage for all courses

---

7. Non-Functional Requirements

- **Usability:** Simple menu-driven console interface
- **Reliability:** Handles invalid input gracefully
- **Maintainability:** Modular code with clear method structures
- **Portability:** Runs on any system with Java support

8. System Architecture

```
+-------------------------+
|     User Interface     |  <- Console-based Menu
+-------------------------+
          |
          v
+-------------------------+
|   Application Logic    |  <- Course, Assessment, Grade Calculation
+-------------------------+
          |
          v
+-------------------------+
|    Data Storage        |  <- In-memory (ArrayList)
+-------------------------+
```

9. Design Diagrams

9.1 Use Case Diagram

```
+---------+
| Student |
+---------+
    |
    |-- Add Course
    |-- Add Assessment
    |-- View Grades
    |-- Predict Required Marks
    |-- View Summary
```

9.2 Class Diagram

```
+-----------------+
|    Course     |
+-----------------+
| - name         |
| - assessments[] |
+-----------------+
| + addAssessment()|
| + removeAssessment()|
| + calculateGrade() |
+-----------------+


+-----------------+
|   Assessment    |
+-----------------+
| - name         |
| - maxMarks      |
| - weightPercent |
| - scoredMarks   |
+-----------------+
```

---

10. Implementation Details

The project was implemented using Java's Object-Oriented Programming features such as classes, encapsulation, and lists. The Main class handles user interaction, while Course and Assessment classes manage data and computations.

Key Classes:

- **Main.java** – Entry point with all menu and logic handling.
- **Course.java** – Represents a single course containing multiple assessments.
- **Assessment.java** – Stores assessment details (name, max marks, weight, and score).

---

11. Sample Output

```
=== Smart Academic Planner ===

1. Add Course

2. Manage Course

3. Calculate Current Grade

4. Predict Target Grade
```

```
5. Exit

Enter your choice: 1
Enter course name: Mathematics
Course added successfully.
```

---

12. Testing and Validation

Test Case	Description	Expected Output	Result
1	Add Course	Course added to list	Pass
2	Add Assessment	Assessment added successfully	Pass
3	Calculate Grade	Correct weighted grade	Pass
4	Predict Required Marks	Displays correct required average	Pass
5	Invalid Input	Graceful error handling	Pass

---

13. Challenges Faced

- Handling invalid or missing input gracefully.
- Ensuring correct rounding of weighted grade calculations.
- Designing a clean console-based interface.

---

14. Learning Outcomes

- Implemented modular, reusable Java code.
- Gained hands-on experience with Git and GitHub for project submission.
- Learned how to structure a complete software project with documentation.

---

15. Future Enhancements

- Add persistent data storage (using JSON or SQLite)
- Develop a GUI version using JavaFX or Swing
- Enable export of grade reports to PDF/CSV

---

16. Conclusion

The **Smart Academic Planner & Grade Predictor** provides a simple, efficient way for students to track their academic performance and plan for target grades. It fulfills the primary objectives of the **CSE2006** course project, demonstrating practical use of OOP concepts and basic software engineering principles.

---

17. References

- Oracle Java Documentation: https://docs.oracle.com/en/java/
- GitHub Guides: https://guides.github.com/
- GeeksforGeeks: Java Collections Framework

Author: Qusay Saify
Course Code: CSE2006
Program: Integrated M.Tech in AI & ML