

STUDENT GRADE

MANAGEMENT

SYSTEM

FULL PROJECT REPORT

COURSE=INTRODUCTION TO PROBLEM SOLVING AND
PROGRAMMING

SUBMITTED BY-PEEHU GULATI

REGISTRATION NUMBER-25BAI10783

VIT BHOPAL UNIVERSITY

Date-November 22,2025

1. Introduction

The Student Management System is a simple program made with Python that helps students keep track of their grades.

Students can log in with their ID, see their GPA for each semester, add new grades, check their progress in graphs, compare their grades with the class average, compare with a friend's grades, and even predict their future GPA based on their past grades.

The system saves data in CSV files and uses pandas, numpy, and matplotlib to work with and show the information.

2. Problem Statement

Students often have a hard time keeping track of their school information by hand. It is tough to look at how they did each semester, see patterns in their performance, guess future grades, and compare with classmates without a good system.

The goal is to create an easy-to-use system that:

- Holds data
- Calculates GPA
- Displays performance clearly
- Assists in comparisons
- Predicts future GPA

3. Functional Requirements

Here are the things the system can do:

1. Log in with Student ID
2. See the grades for each semester

3. View GPA growth with charts
4. Compare GPA with the class average
5. Compare performance with a classmate
6. Guess GPA for next semester
7. Add new student grades to the system
8. Log out of the system

These functions include everything needed for basic academic analysis.

4. Non-Functional Requirements

- Usability: Easy menu-driven interface
- Performance: Fast data processing
- Reliability: Accurate and consistent results
- Maintainability: Modular code structure
- Scalability: Handles multiple students
- Portability: Works on any system with Python

5. System Architecture

The project is divided into four modules:

- main.py: Handles menu and login
- datamanagement.py: Reads and updates CSV data
- gradecalculator.py: Calculates GPA and predictions
- comparision.py: Plots graphs (growth, class comparison, peer comparison)

6. Design Diagrams

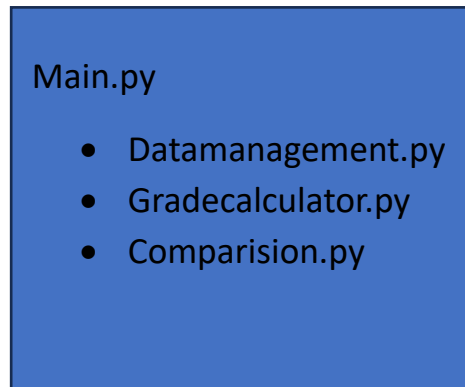
Case description

case	Description
Login	Access system using Student_id
View report card	View Gpa of each semester
Visualize growth	See performance rend line graph
Class avg compare	Compare gpa with avg
Peer comparison	Compare gpa with friend
GPA prediction	Next gpa estimation
Add record	Insert new marks in csv
exit	End the session

Workflow Diagram

- 1) Start
- 2) Enter Student ID
- 3) Verify Id from csv file
- 4) If invalid show error message, if valid display menu.
- 5) Select option shown
- 6) Selected option is performed
- 7) Back to main menu
- 8) Exit

Component Diagram

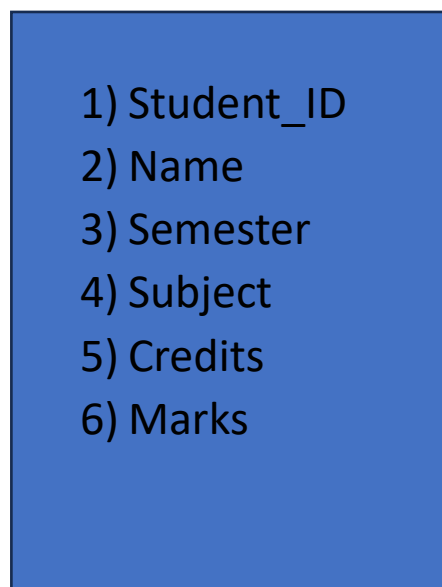


main.py → datamanagement.py → gradecalculator.py -> comparision.py→
back to main.py

ER Description

CSV contains fields:

Student_ID, Name, Semester, Subject, Credits, Marks



7. Design Decisions & Rationale

- CSV chosen for simple storage

- pandas used for efficient data handling
- matplotlib used for meaningful visual output
- NumPy used for GPA prediction
- Modular programming used for clean structure

8. Implementation Details

- Data is stored in student_marks.csv
- GPA is calculated using weighted credit formula
- Graphs included:-
 - Line graph of personal GPA trend
 - Bar graph comparing with class average
- Line graph comparing with a peer
- New records are added by appending to CSV
- The interface is fully text-based

9. Results

- Dashboard menu and Login screen

```
PS C:\Users\peehu\OneDrive\Desktop\CLASSTRACK_PROJECT> python src/main.py
=====
=====STUDENT MANAGMENT SYSTEM=====
=====
PLEASE ENTER STUDENT ID(for e.g 101...) TO CONTINUE THE LOGIN PROCESS AND ACCESS YOUR DATA: 101
YAYYY!!! Login Successful! Welcome, Rahul Trivedi.

-----
Dashboard: Rahul Trivedi
-----
1. SEE My Report Card
2. Visualize My Growth
3. Comparision with Class Average
4. Comparision with Peer
5. Predict Future GPA
6. Add New Marks
7. Exit
Please Select Option (1-7): █
```

- Transcript output

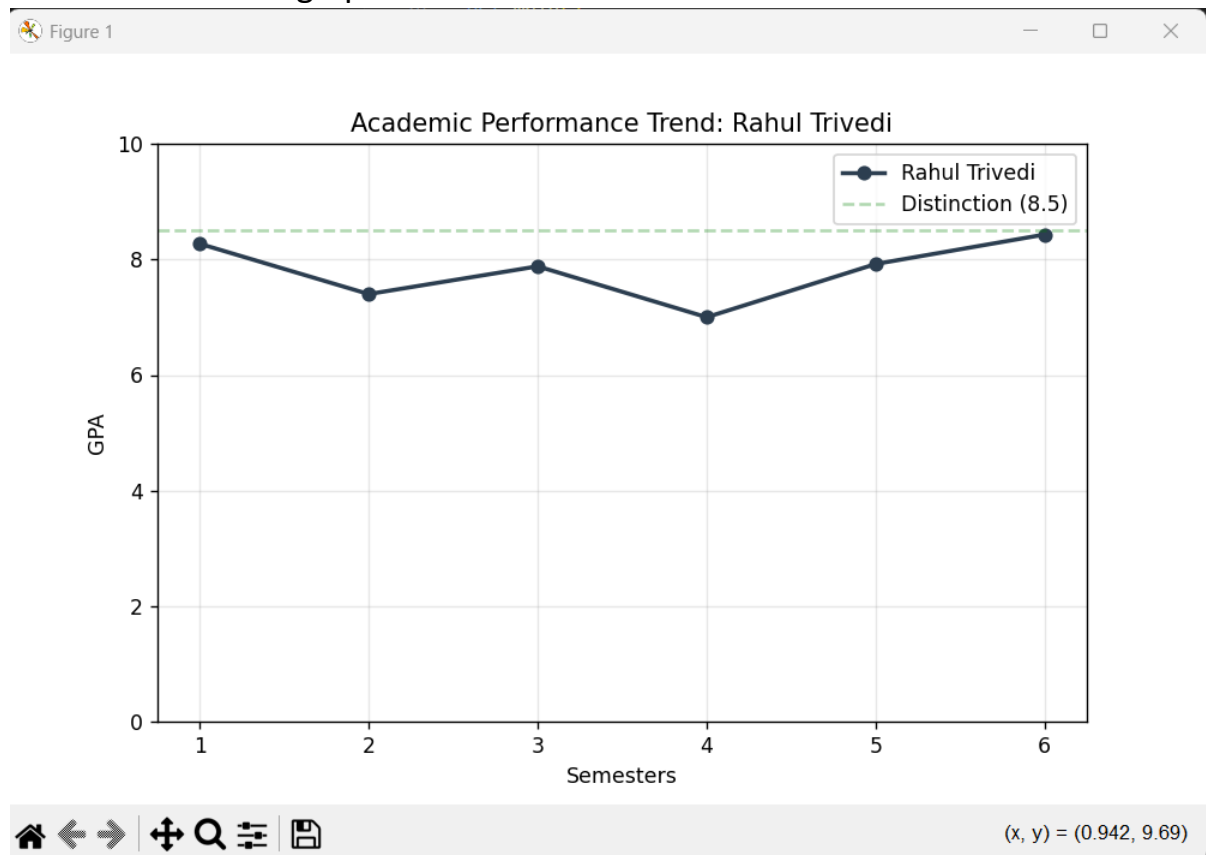
```
PS C:\Users\peehu\OneDrive\Desktop\CLASSTRACK_PROJECT> python src/main.py
=====
=====STUDENT MANAGMENT SYSTEM=====
=====
PLEASE ENTER STUDENT ID(for e.g 101...) TO CONTINUE THE LOGIN PROCESS AND ACCESS YOUR DATA: 101

YAYYY!!! Login Successful! Welcome, Rahul Trivedi.

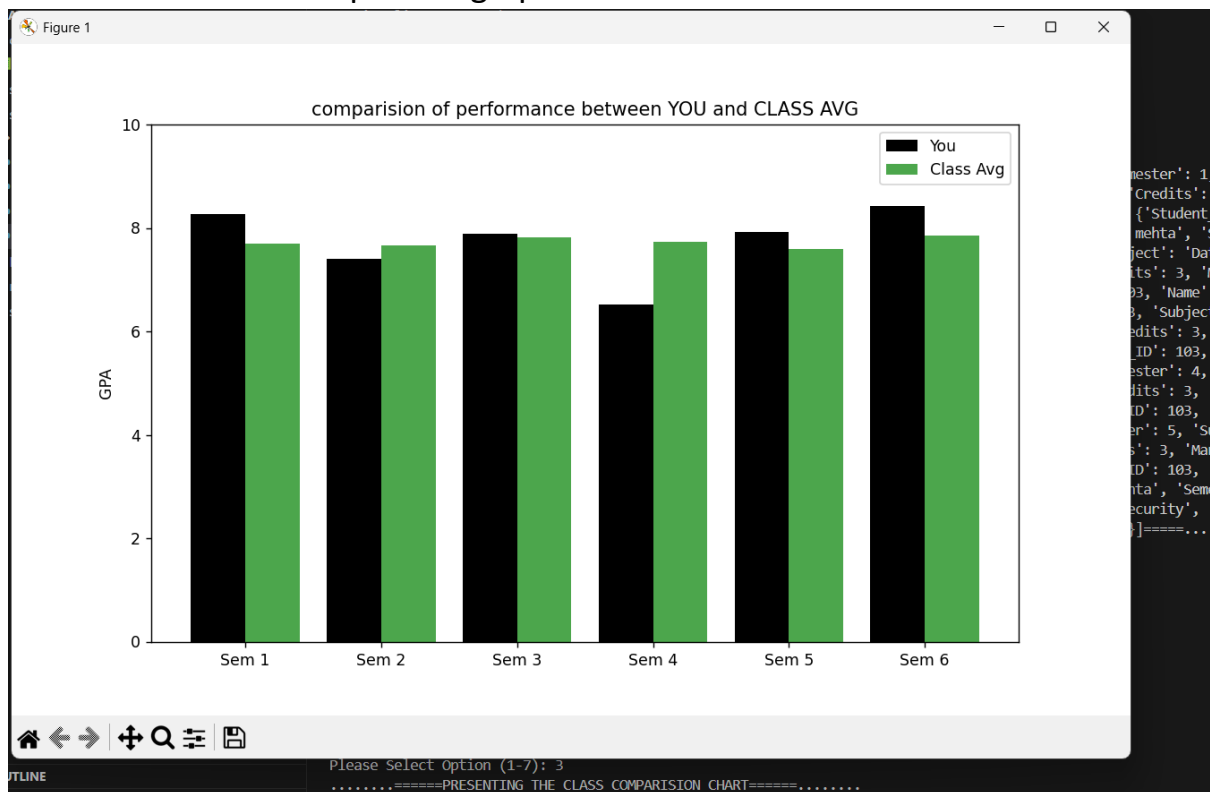
-----
Dashboard: Rahul Trivedi
-----
1. SEE My Report Card
2. Visualize My Growth
3. Comparision with Class Average
4. Comparision with Peer
5. Predict Future GPA
6. Add New Marks
7. Exit
Please Select Option (1-7): 1

[Transcript: Rahul Trivedi]
Semester 1: 8.27 SGPA
Semester 2: 7.4 SGPA
Semester 3: 7.88 SGPA
Semester 4: 6.52 SGPA
Semester 5: 7.92 SGPA
Semester 6: 8.43 SGPA
```

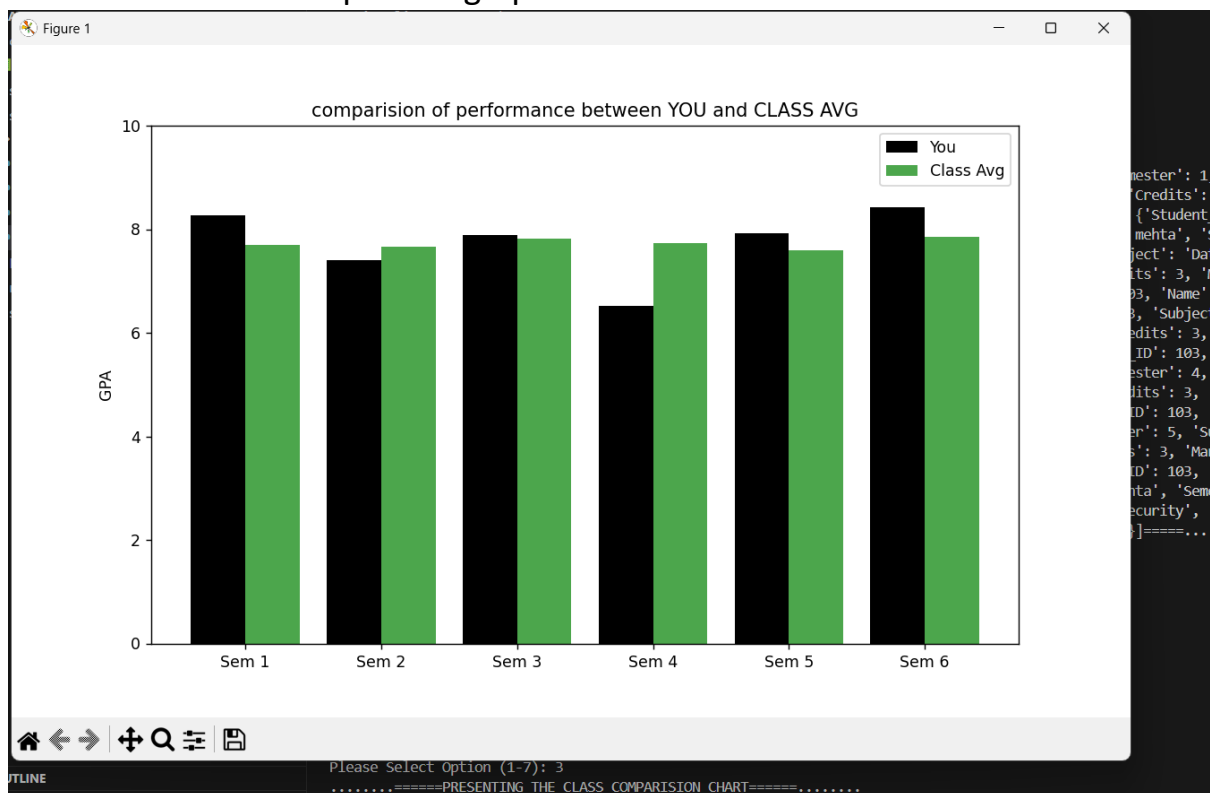
- Trend graph



- Class comparison graph



- Peer comparison graph



- Addition of course


```
-----
Dashboard: Rahul Trivedi
-----
1. SEE My Report Card
2. Visualize My Growth
3. Comparision with Class Average
4. Comparision with Peer
5. Predict Future GPA
6. Add New Marks
7. Exit
Please Select Option (1-7): 6
Semester Number: 4
Subject Name: DSA
Credits (1-4): 4
Marks obtained (0-100): 90
Record Added Successfully.
```

- Exit

```
Dashboard: Rahul Trivedi
-----
1. SEE My Report Card
2. Visualize My Growth
3. Comparision with Class Average
4. Comparision with Peer
5. Predict Future GPA
6. Add New Marks
7. Exit
Please Select Option (1-7): 7
Logging out, Good luck!
```

10. Testing Approach

- Input validation for IDs, credits, marks
- GPA calculation verification
- CSV structure testing
- Graph correctness testing
- Checking predictions with sample data

11. Challenges Faced

- Handling missing or invalid student IDs
- Managing CSV updates
- Maintaining accuracy when multiple records exist
- Structuring code into clear modules
- Understanding GPA formula and credit weightage

12. Learnings & Key Takeaways

- Working with pandas for data management
- Plotting graphs using matplotlib
- Implementing GPA algorithms
- Handling CSV-based storage
- Designing a modular Python project
- Understanding academic data visualization

13. Future Enhancements

- Adding GUI using Tkinter
- Implementing login authentication
- Exporting transcripts as PDF
- Adding teacher/admin module
- Adding attendance or fees module
- Moving storage from CSV to SQL database

14. References

- Python official documentation
- pandas library documentation
- matplotlib documentation
- NumPy documentation