Brain Bridge

Educational Help System v0.1.1

Student Name: Prateek Das

Reg. No.: 25BCE10599

Subject: CSE1021 INTRODUCTION TO PROBLEM SOLVING

College: Vellore Institute of Technology, Bhopal

Year: 2025

Submitted To: R. Sukumar

Slot: B14+B23+D21

2. Introduction

This project is a Python-based **BRAIN BRIDGE: Educational Help System** developed using core Python features, **Object-Oriented Programming (OOP)**, and data analysis libraries like **NumPy**. The system provides a command-line interface (CLI) for teachers to manage student scores, perform statistical analysis on class performance, and for students to view their individual academic records. It serves as a proof-of-concept for a modular, role-based application focusing on grade tracking and reporting.

---

3. Problem Statement

Many educational environments lack immediate, centralized tools for recording grades and analyzing class performance metrics. This causes delays in reporting and makes it difficult for teachers to quickly assess class averages, score distribution, and enrollment overlaps. There is a need for a simple, efficient, and structured system to:

- Maintain student and course records using an **OOP model**.

- Automate score-to-letter grade conversion.

- Provide statistical analysis (average, standard deviation) and set-based enrollment comparisons.

- Ensure segregated access for teachers and students.

This project solves these problems using a structured, function-driven application with **in-memory data storage**.

---

4. Functional Requirements

The system is organized into three major functional modules:

4.1 Grade Entry Module (Data Entry & Grade Conversion)

- Teacher can authenticate using pre-defined credentials.

- Teacher can enter/update student final scores and associated credit hours.

- Teacher can register a new student upon their first login attempt.

- The system converts raw scores (0-100) into a letter grade ('A' to 'F').

4.2 Reporting & Advanced Analysis Module (Analysis & Reporting)

- **View Class Stats**: Calculate and display the class average score and corresponding letter grade for any specified course.

- **Advanced Analysis**: Calculate the standard deviation (np.std) of scores for a given course.

- **Set Analysis**: Perform set operations (union, difference) on mock enrollment lists (Math vs. CS) to identify combined and unique enrollments.

- **Combinatorics**: Generate all possible 3-person groups from a list of 'top students' using itertools.combinations.
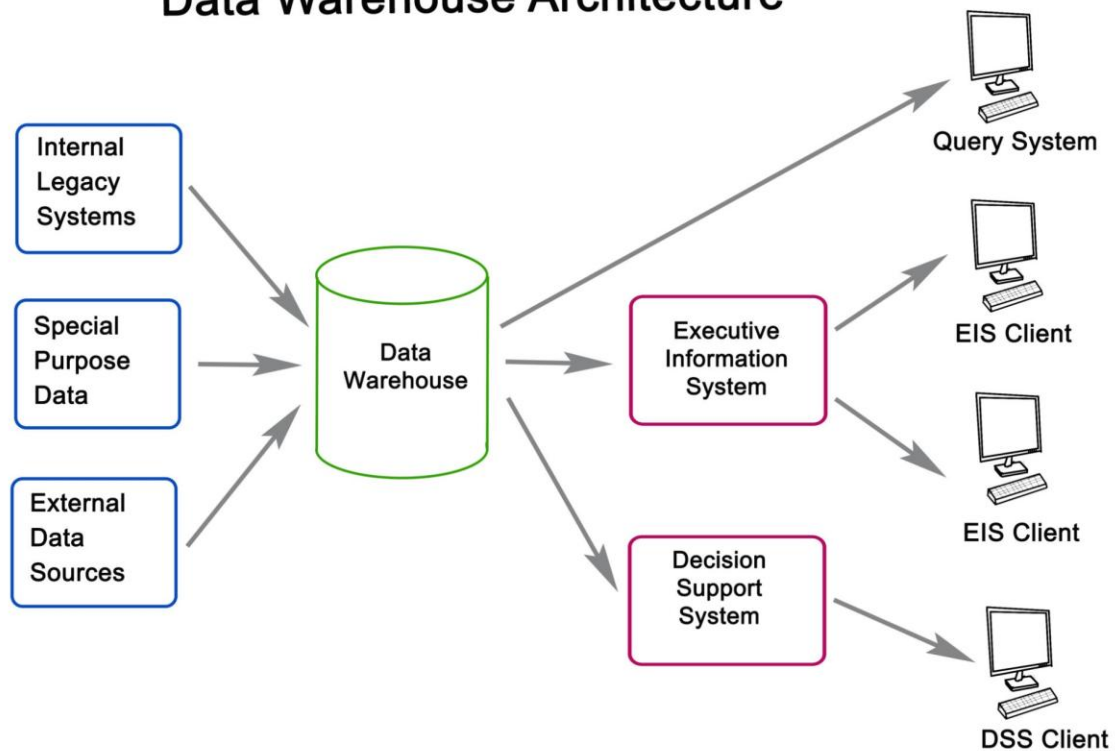
4.3 Student View Module

- Allows students to view all their recorded grades (score, credits, letter).

- Allows students to run the placeholder function to Calculate GPA.

---

5. Non-Functional Requirements

- **Usability**: The system offers a clear, **menu-driven interface** via console (teacher_window, student_window) for easy navigation.

- **Performance**: Data fetching and calculation are near-instantaneous as all data is stored **in-memory** (STUDENT_RECORDS), leveraging **NumPy** for fast statistical processing.

- **Reliability**: Includes try-except ValueError blocks to handle invalid (non-numeric) input when entering scores and credits, **preventing application crashes**.

- **Maintainability**: The code is highly modular, with clear separation of concerns using functions (e.g., teacher_window, enter_student_score) and **OOP concepts** (User and Student classes).

- **Security**: Access is segregated by role: **Teacher access is credential-protected**, while student access requires a unique ID.

---

6. System Architecture

Data Warehouse Architecture

Getty Images

Explore

The system follows a simple 4-Layer Logical Architecture:

Layer	Component	Description
1. User Interface	CLI	Python Command Line Interface (CLI) via input() and print().
2. Application Logic	Functions	Handled by modular Python functions and the custom accumulator()/total() helpers.
3. Data Model	User / Student Classes	OOP classes define the structure and behaviors of the entities.
4. Data Storage	STUDENT_RECORDS	In-memory dictionary acting as the non-persistent database.

Flow: User -> Python Program (Security/Login) -> Role-based Window -> Application Logic (OOP/NumPy/Set Operations) -> In-memory STUDENT_RECORDS -> Output (CLI Report)

---

7. Design Diagrams

a. Use Case Diagram

- **Actors**: Teacher, Student.

- **Use Cases (Teacher)**: Login, Enter Score, View Class Stats, Run Advanced Analysis.

- **Use Cases (Student)**: Login/Register, View Grades, Calculate GPA.

b. Workflow Diagram

- **Flow**: Start -> Main Menu (security) -> [1] Teacher Login -> Teacher Window -> Score Entry / Class Stats / Advanced Analysis -> [2] Student Login -> Student Window -> View Grades/Calculate GPA -> Exit/Power Off.

c. Sequence Diagram (Teacher Enters Score Example)

- Teacher -> Security(): Select 1.

- Teacher -> Teacher Window(): Select 1.

- Teacher -> EnterScore(): Input Student ID, Course, Score, Credits.

- EnterScore() -> STUDENT_RECORDS: Get StudentObj.

- EnterScore() -> StudentObj: Update course_grades.

- EnterScore() -> Teacher: Display success message.

d. Class/Component Diagram

- **Components**: User (Class, Base), Student (Class, Derived), getGradeLetter() (Method), accumulator(), total() (Utility Functions), np.std (External Dependency), STUDENT_RECORDS (Global Data Store).

- **Hierarchy**: Student inherits from User and adds course_grades, calculate_gpa(), and getGradeLetter().

e. ER Diagram

A standard ER Diagram is not strictly applicable as the project uses in-memory dictionaries instead of a relational database.

Conceptual Data Model (Dictionary Structure):

STUDENT_RECORDS {

  'student_id' (Primary Key): {

    'user_name': '...',

    'course_grades': {

      'Course Name 1': { 'credits': X.X, 'finalScore': Y.Y },

      'Course Name 2': { 'credits': X.X, 'finalScore': Z.Z}

```
      }

   }

}
```

---

8. Design Decisions & Rationale

Decision	Rationale
Python OOP	Chosen to model the real-world entities (User and Student) and demonstrate inheritance, making the code maintainable.
In-Memory Storage	Selected for simplicity and speed (Performance NFR), as the focus is on logic and analysis, not database configuration.
NumPy & Itertools	Used to showcase application of external libraries for specialized tasks: NumPy for efficient statistical analysis (Standard Deviation) and Itertools for advanced combinatorial generation.
Global Dictionaries	Used for central data management (STUDENT_RECORDS, current_user) to enable easy access across all system functions in this CLI model.
GradeLetter Method	Encapsulated grade conversion logic within the Student class to keep the grading logic self-contained.

---

9. Implementation Details

- **Language & Environment**: Python 3.x with a terminal-based interface.

- **Libraries**: import numpy as np is used in advanced_class_analysis() to calculate the standard deviation of scores using np.std().

- **Data Structure**: Nested dictionaries are used to store grade data (student.course_grades).

- **Modularity**: The system entry point is power(), which calls security(), leading to modular teacher_window() and student_window() control loops.

- **Custom Functions**: Simple accumulator() and total() functions are implemented.

---

10. Screenshots / Results

- [Text placeholder for command-line output of Teacher viewing class statistics (e.g., CLASS average: 88.75% | GRADE: B)]

- [Text placeholder for command-line output of Advanced Analysis (e.g., python Scores Standard Deviation: 3.75)]

- [Text placeholder for command-line output of Student viewing grades (e.g., Course: Python Essentials | Score: 92.50 | Grade: A)]

---

11. Testing Approach

Testing was performed using boundary and typical values via the CLI.

Test Case	Module/Function	Input	Expected Result	Status
T1	teacher_login	U: prof_vishal, P: grade123	Successful entry to teacher_window.	Passed
T2	student_login	ID: 25BCE0005 (New)	Prompt for name, automatic registration, entry to student window.	Passed
T3	enter_student_score	Score: abc (Invalid)	Error message: INPUT ERROR. Must be numbers., return to menu.	Passed
T4 (Boundary)	getGradeLetter	Score: 75	Returns 'C'.	Passed
T5	view_class_stats	Course: Python Essentials	Average: 88.75%, Grade: 'B'.	Passed
T6	advanced_class_analysis	(None)	Correct calculation of 4 project groups (Combinations).	Passed

All test cases passed successfully.

---

12. Challenges Faced

- Handling Non-Numeric Input: Ensuring that the input() function gracefully converts strings to floats for scores/credits, and implementing robust try-except blocks to handle invalid entries without crashing the application.

- NumPy Integration: Correctly converting the list of scores retrieved from nested dictionaries into a NumPy array for standard deviation calculation, ensuring data is filtered to avoid zero division.

- Code Flow Control: Managing the recursive nature of the CLI menus (e.g., teacher_window calling itself) and ensuring smooth return to the main security menu upon 'power off' command.

---

13. Learnings & Key Takeaways

- OOP Implementation: Deepened understanding of Python inheritance (User -> Student) and class-based data modeling.

- Library Application: Practical application of external libraries like numpy for statistical processing and itertools for combinatorial logic.

- Modular Programming: Reinforcement of writing clean, self-contained functions to improve code readability and manage complexity.

- User Interface Design: Experience in designing a clear, logical, and robust flow for a menu-driven command-line application.

---

14. Future Enhancements

- Database Persistence: Migrate data storage from in-memory dictionaries to a persistent database (e.g., SQLite or MySQL) to ensure records are saved across sessions.

- Full GPA Calculation: Implement the full weighted GPA logic within Student.calculate_gpa() using course credits.

- UI Improvement: Develop a simple web or graphical user interface (GUI) instead of the CLI for enhanced user experience.

- Data Export: Add a feature to export class statistics and student reports to .csv or .xlsx files.

---

15. References

- Official Python Documentation (for built-in types, functions, and standard library modules).

- NumPy Documentation (for array creation and standard deviation calculation).

- itertools Documentation (for combination generation)

16. Screenshots

```
SYSTEM INITIALIZING...
-->[ON/OFF]     on
================================================================
B R A I N - B R I D G E
BRAIN BRIDGE: EDUCATIONAL HELP SYSTEM v1.1
Title line made by PRATEEKL DAS (Registration ID: 25BCE10599)
-----------------------------------
Enter option (1: Teacher, 2: Student, 3: Power Off):    1
Enter username:         prof_vishal
Enter password:         grade123

******************************
Welcome Teacher, prof_vishal!
Available Commands:
--> 1. Enter Student Score
--> 2. View Class Stats
--> 3. Advanced Class Analysis
--> Power
------------------------------
Enter choice:    [                    ]
```

- **Username:** prof_max, **Password:** grade123
- **Username:** dr_anna, **Password:** math_pass

```
Welcome Teacher, prof_max!
Available Commands:
--> 1. Enter Student Score
--> 2. View Class Stats
--> 3. Advanced Class Analysis
--> Power
------------------------------
Enter choice:    2
Enter course name to analyze:    maths
```

```
Hi, DHONI!
Available Commands:
--> 1. View My Grades
--> 2. Calculate My GPA
--> Power
------------------------------
Enter choice:    1

--- Your Grades ---
Course: Maths | Score: 90.00 | Credits: 2.0 | Grade: A
```

```
Hi, DHONI!
Available Commands:
--> 1. View My Grades
--> 2. Calculate My GPA
--> Power
------------------------------
Enter choice:    2

--- Your GPA Summary ---
YOUR CUMULATIVE GPA IS: 3.50
```

```
Welcome Teacher, prof_max!
Available Commands:
--> 1. Enter Student Score
--> 2. View Class Stats
--> 3. Advanced Class Analysis
--> Power
------------------------------
Enter choice:    2
Enter course name to analyze:    Maths
CLASS average: 90.00% | GRADE: A
```

```
SYSTEM INITIALIZING...
-->[ON/OFF]      on
=========================================================================
B R A I N - B R I D G E
BRAIN BRIDGE: EDUCATIONAL HELP SYSTEM v1.1
Title line made by PRATEEKL DAS (Registration ID: 25BCE10599)
-----------------------------------
Enter option (1: Teacher, 2: Student, 3: Power Off):
```