Abstract

This report presents the final overview of the **CheckMyGrade Application**, a Python-based student grade management system developed to streamline academic course administration. The application offers comprehensive functionalities for managing student records, courses, professors, grades, and secure user login. To ensure reliability and performance, the system has been thoroughly tested with extensive unit tests.

Introduction

The **CheckMyGrade Application** is designed to simplify academic data management and enhance the user experience through an intuitive command-line interface. The key features of the application include:

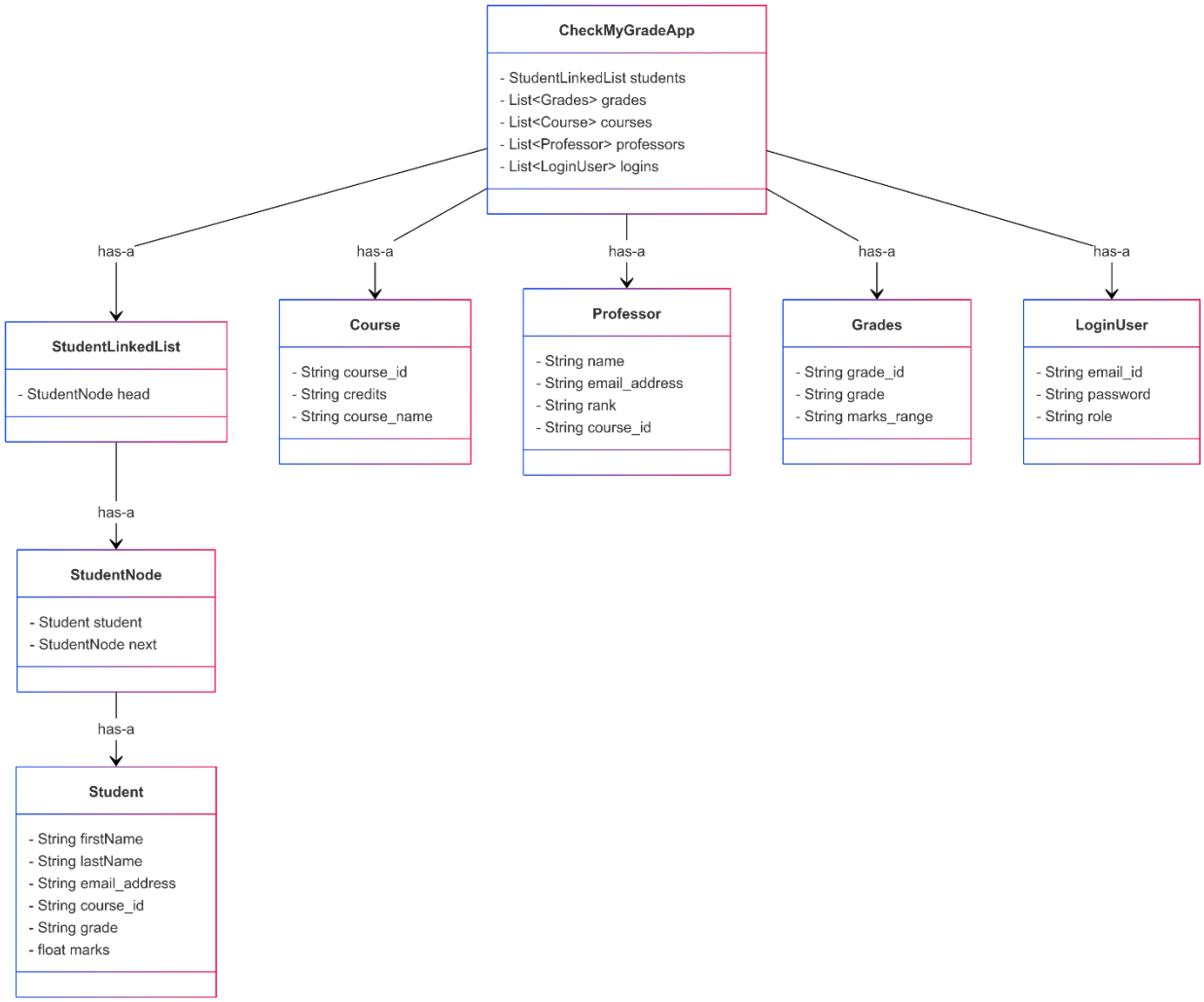
- **Student Management:** Add, update, delete, and search student records using a dynamic linked list structure.
- **Course and Professor Management:** Manage course and professor details, including course-professor associations.
- **Grades and Reporting:** Process and manage grade data, with support for generating detailed reports categorized by course, professor, or student.
- **User Authentication:** Secure login functionality with encrypted password storage and verification.
- **Data Persistence:** Efficient storage and retrieval of data using CSV files for long-term data management.

Project Architecture and Design

The application follows a modular, object-oriented architecture that promotes scalability and ease of maintenance. The primary components include:

- **Student:** Handles individual student information and attributes.
- **StudentLinkedList:** Manages a collection of student records using a linked list data structure.
- **Grades:** Facilitates the management and processing of student grades.
- **Course:** Stores and manages course-related information.
- **Professor:** Maintains data about professors and their assigned courses.
- **LoginUser:** Implements user login mechanisms, including password encryption and authentication protocols.
- **CheckMyGradeApp:** Serves as the central controller, integrating all modules and offering a user-friendly, menu-driven interface.

Class Diagram



Implementation Details

The project is implemented in Python, with file I/O operations performed via CSV files for data storage.

- **Source Code Files:**
  - checkmygrade_app.py – Main application code.
  - test_checkmygrade_app.py – Unit tests for validating application functionalities.
- **Data Files:**
  - Student.csv
  - Course.csv

- Professor.csv
- Grades.csv
- Login.csv

The code is structured to ensure ease of maintenance, modularity, and scalability, making it straightforward to add new features or modify existing ones.

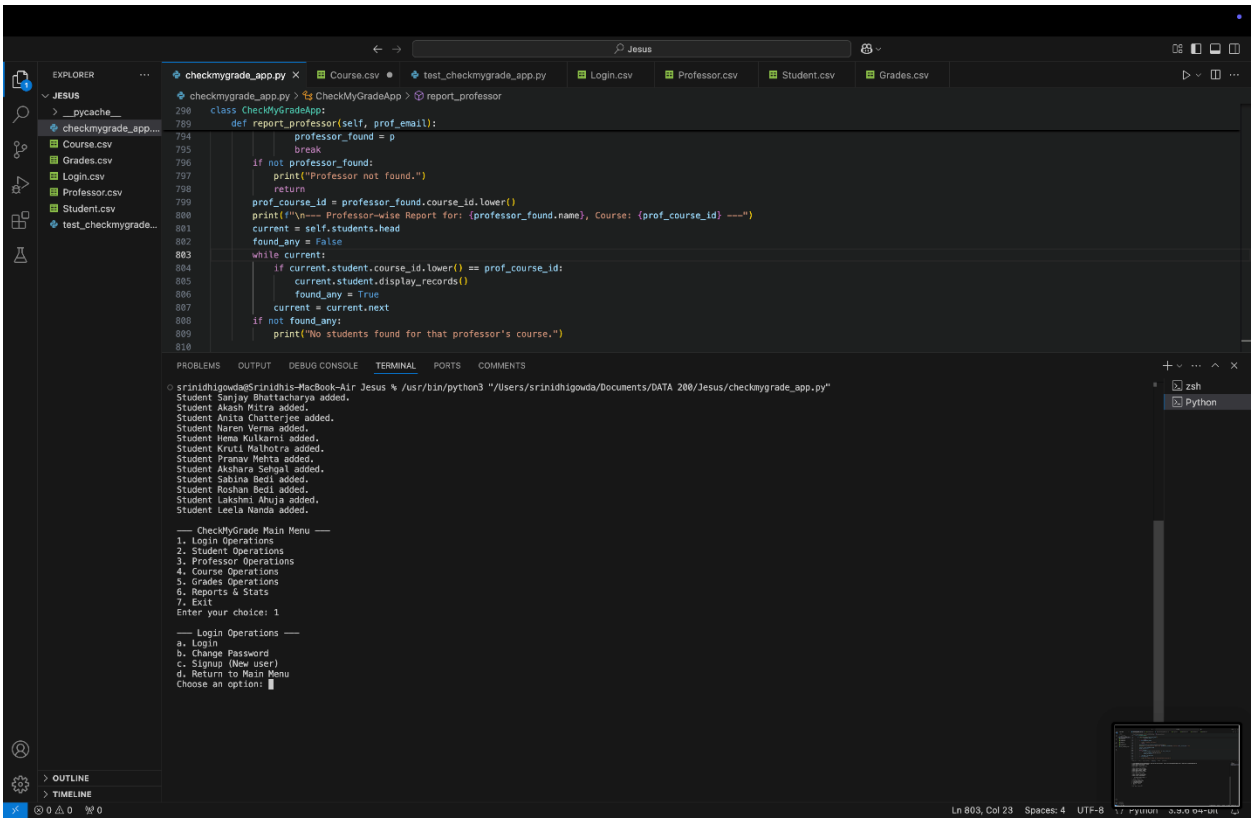
Testing and Terminal Outputs

Below are the terminal screenshots demonstrating the working of the application. All images are embedded using HTML <img> tags with fixed dimensions.

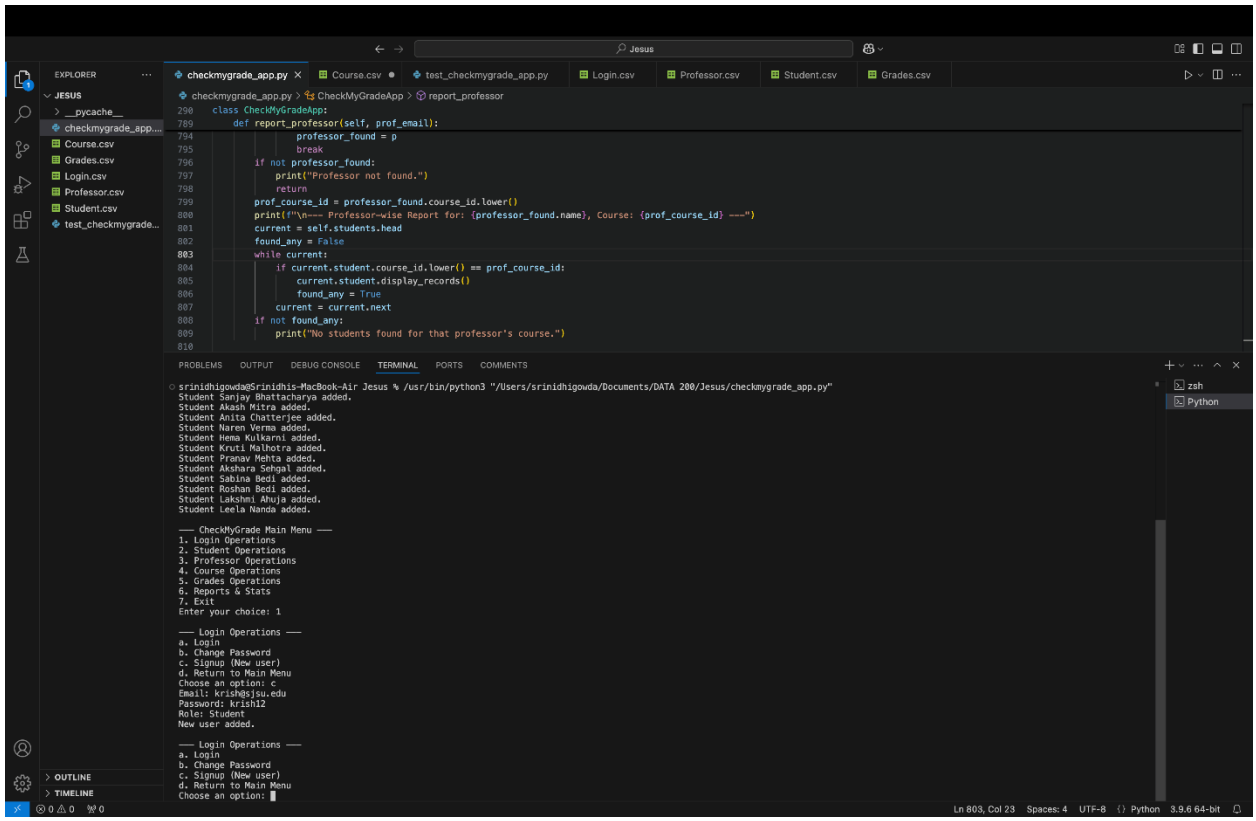
Terminal Screenshot 1: Application Launch and Main Menu

```
Student sharan patil added.
Student Alice Johnson added.
.Student sri nidhi added.
Search completed in 0.000000 seconds.
.Student sharan patil added.
Student deleted.
Student not found. Search completed in 0.000000 seconds.
.Student Alice Johnson added.
Student record updated.
```

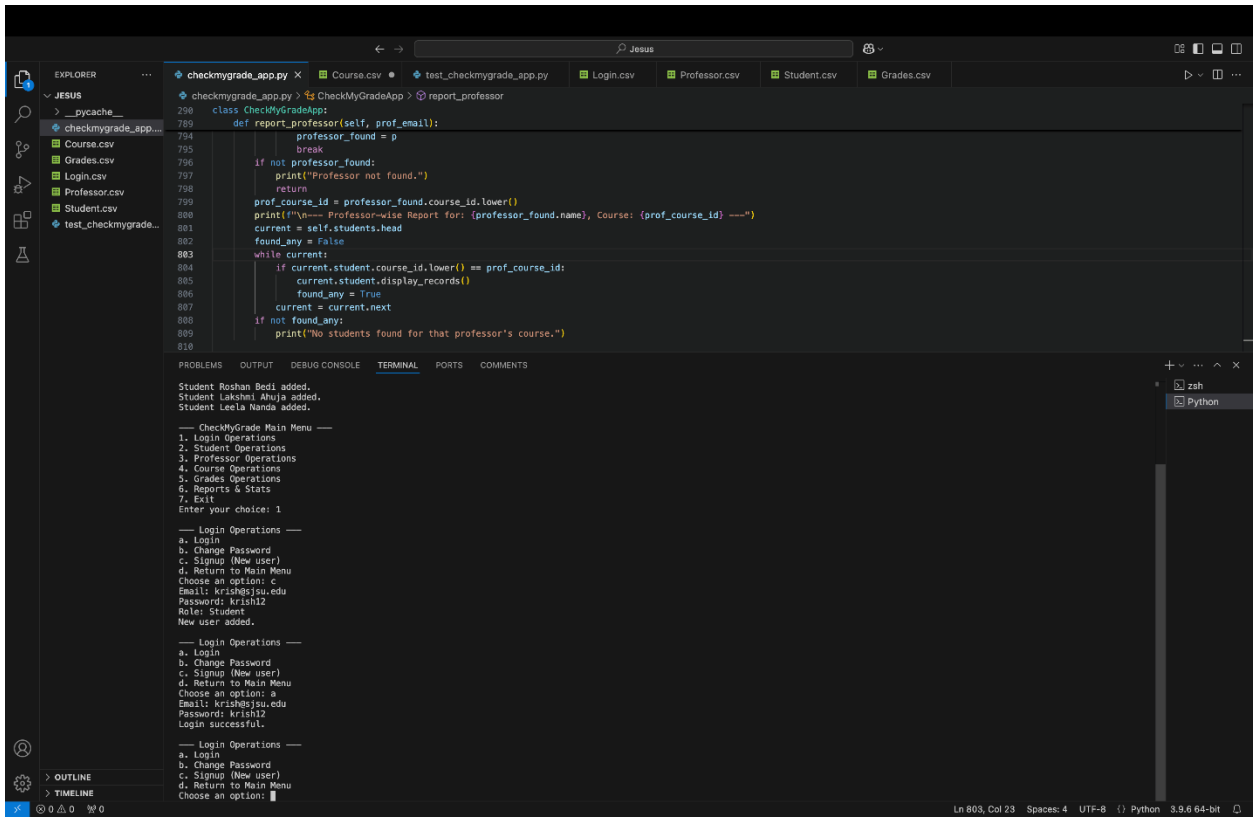
Terminal Screenshot 2: Login Operations



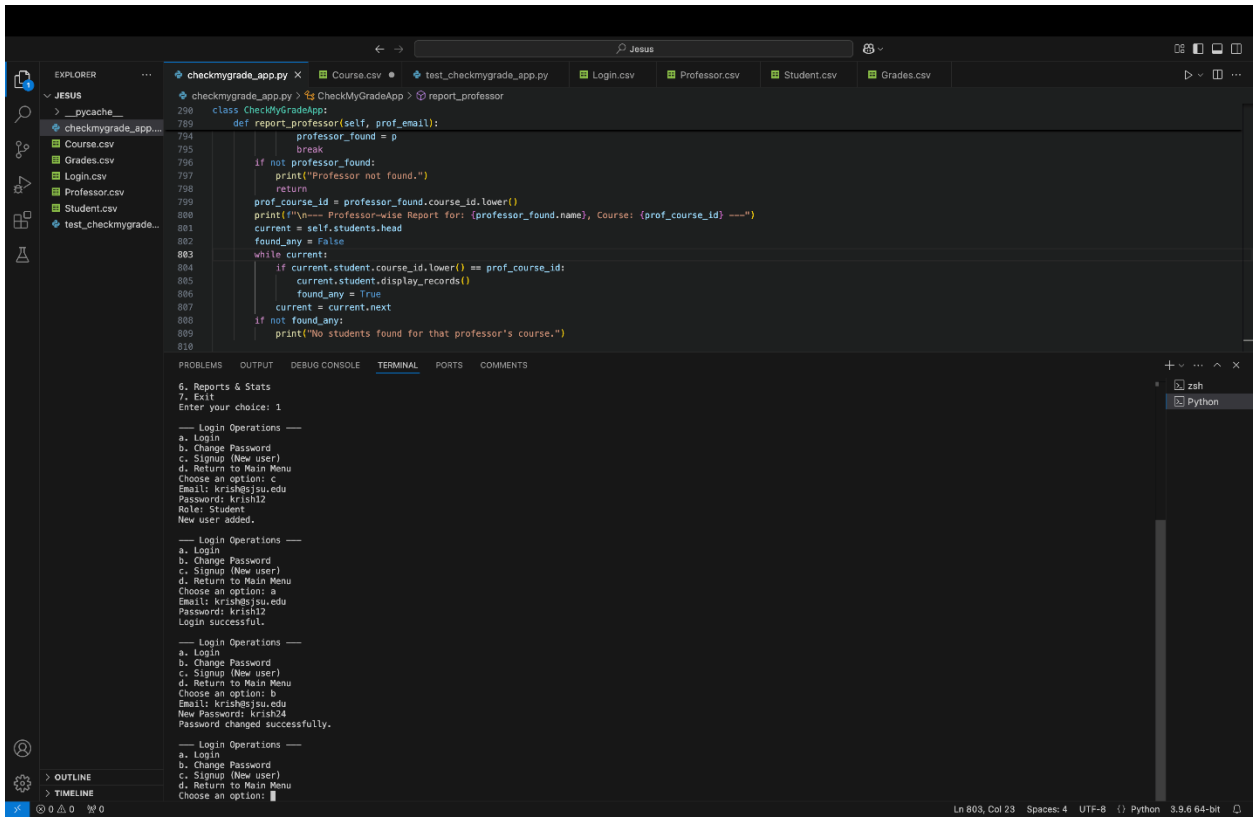
- **Signup a New User:**



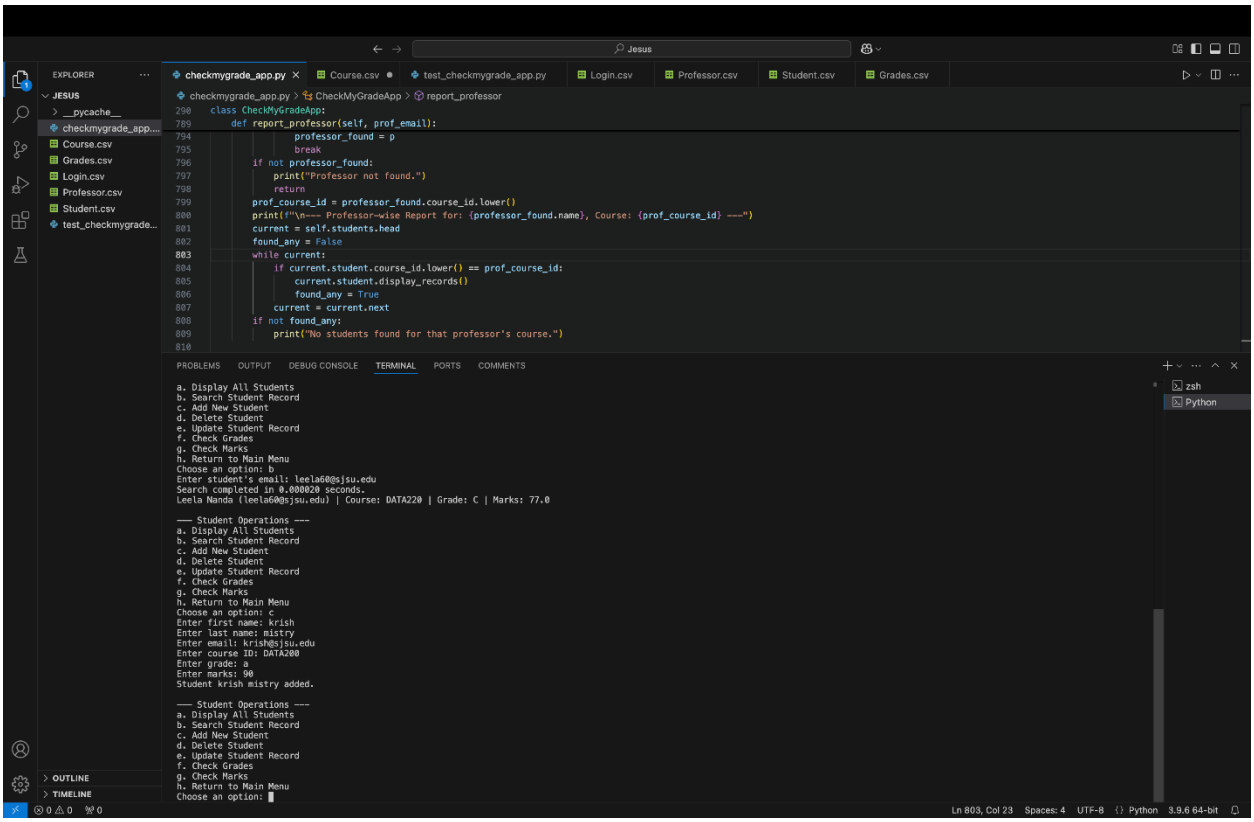
- **After Signup, Login Check of the New User:**



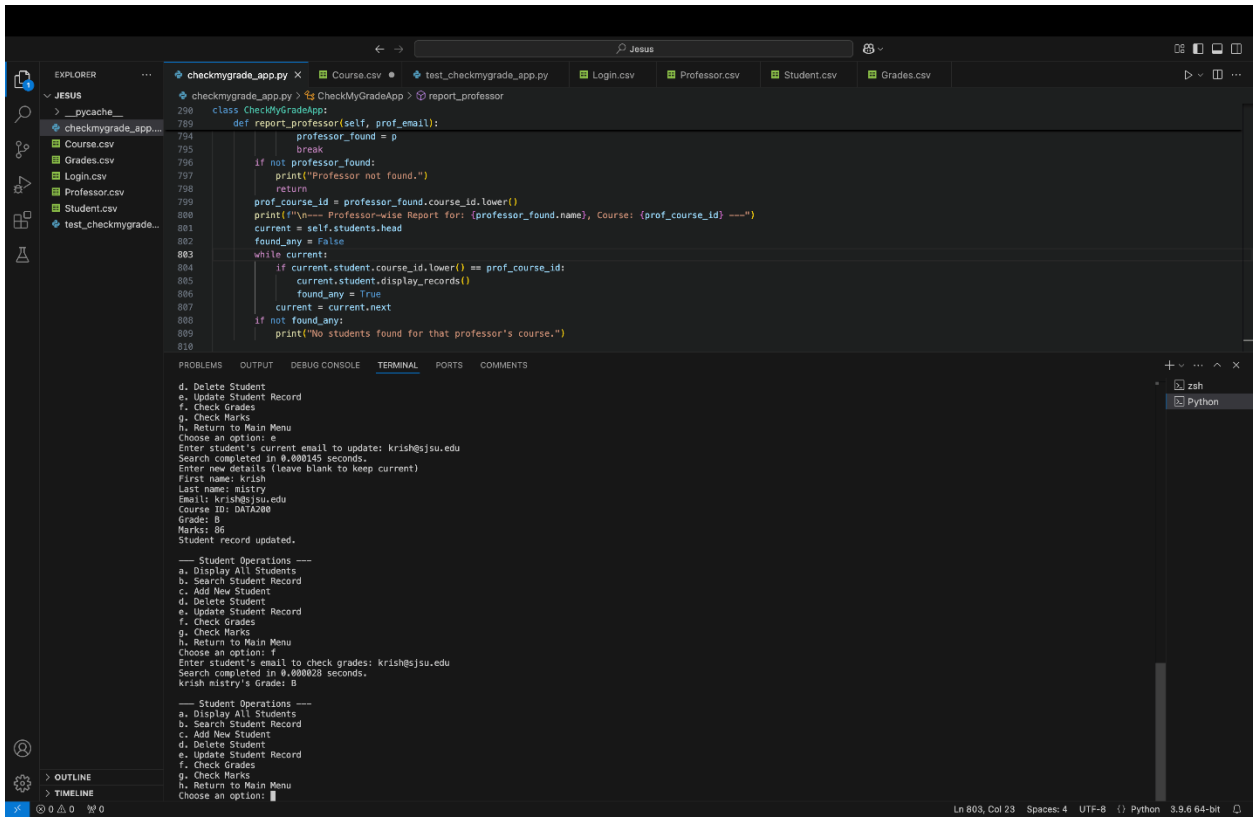
- **Change of Password of the User:**



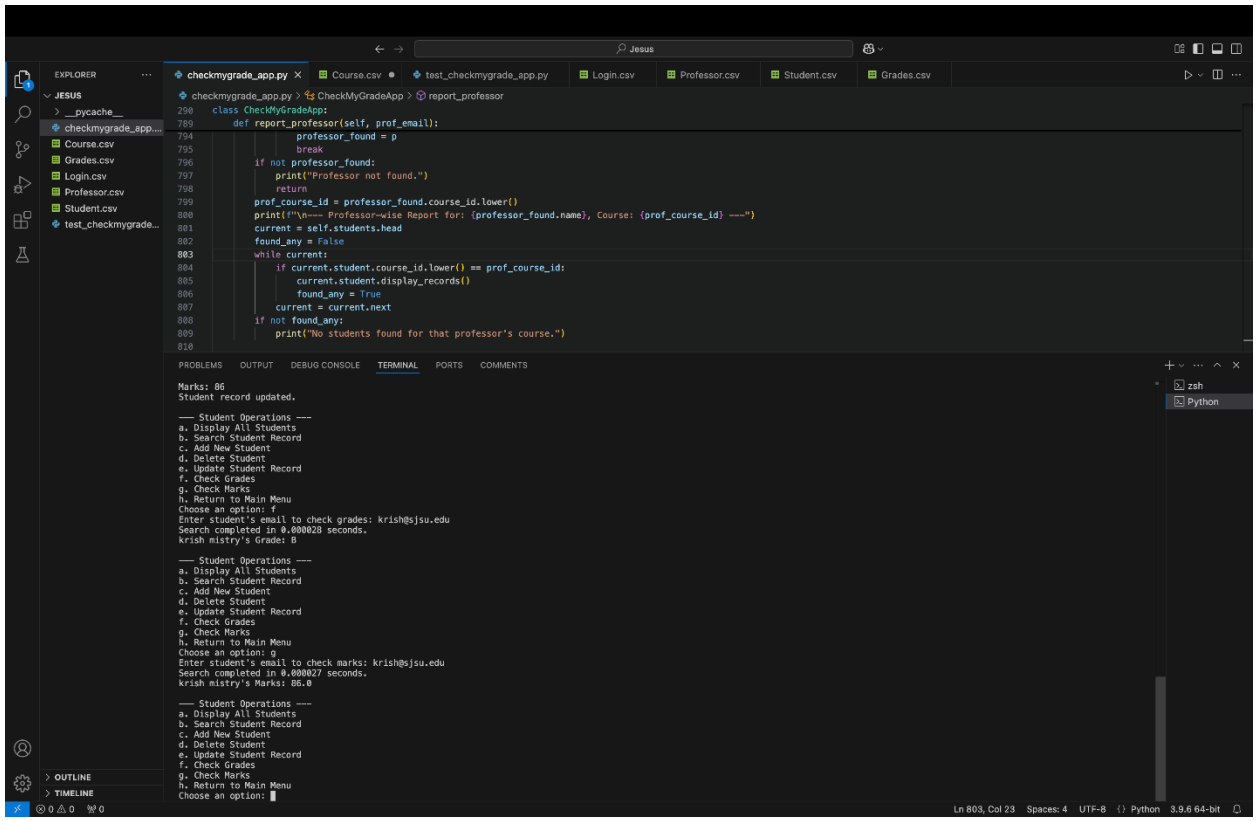
Terminal Screenshot 3: Student Operations



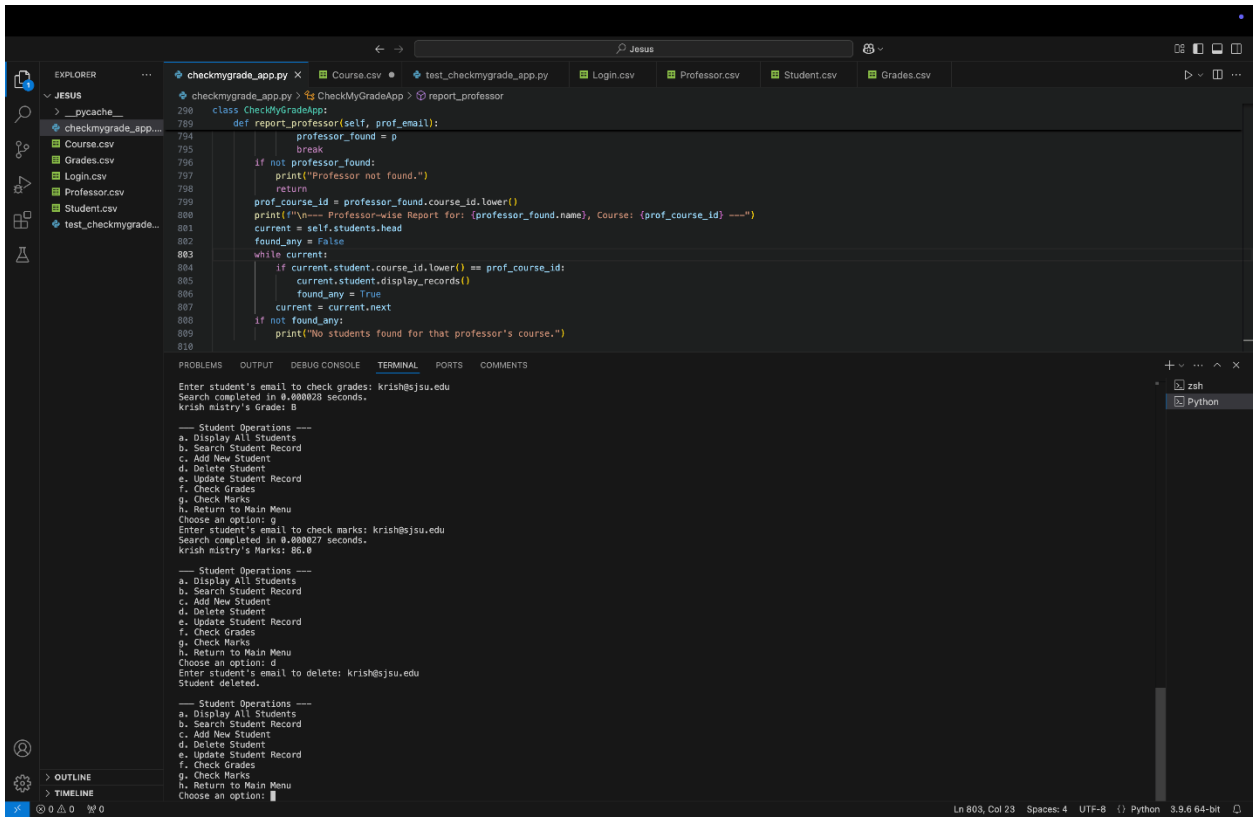
- **Update Student and Checking the Grades:**



- **Check Marks:**

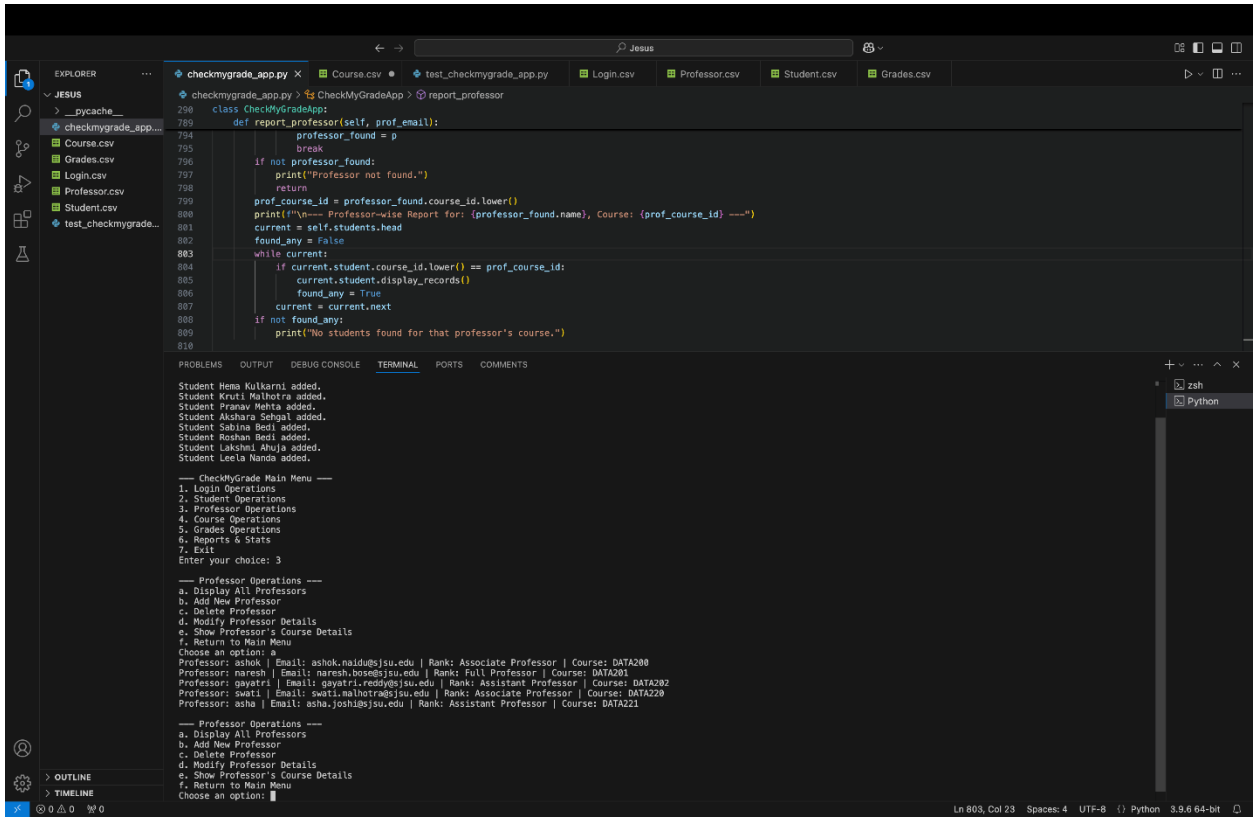

checkmygrade_app.py
Course.csv
test_checkmygrade_app.py
Login.csv
Professor.csv
Student.csv
Grades.csv
EXPLORER
JESUS
__pycache__
checkmygrade_app....
Course.csv
Grades.csv
Login.csv
Professor.csv
Student.csv
test_checkmygrade...
checkmygrade_app.py > CheckMyGradeApp > report_professor
290 class CheckMyGradeApp:
789 def report_professor(self, prof_email):
794 professor_found = p
795 break
796 if not professor_found:
797 print("Professor not found.")
798 return
799 prof_course_id = professor_found.course_id.lower()
800 print(f"\n--- Professor-wise Report for: {professor_found.name}, Course: {prof_course_id} ---")
801 current = self.students.head
802 found_any = False
803 while current:
804 if current.student.course_id.lower() == prof_course_id:
805 current.student.display_records()
806 found_any = True
807 current = current.next
808 if not found_any:
809 print("No students found for that professor's course.")
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
Marks: 86
Student record updated.
--- Student Operations ---
a. Display All Students
b. Search Student Record
c. Add New Student
d. Delete Student
e. Update Student Record
f. Check Grades
g. Check Marks
h. Return to Main Menu
Choose an option: f
Enter student's email to check grades: krish@sjsu.edu
Search completed in 0.000028 seconds.
krish mistry's Grade: B
--- Student Operations ---
a. Display All Students
b. Search Student Record
c. Add New Student
d. Delete Student
e. Update Student Record
f. Check Grades
g. Check Marks
h. Return to Main Menu
Choose an option: g
Enter student's email to check marks: krish@sjsu.edu
Search completed in 0.000027 seconds.
krish mistry's Marks: 86.0
--- Student Operations ---
a. Display All Students
b. Search Student Record
c. Add New Student
d. Delete Student
e. Update Student Record
f. Check Grades
g. Check Marks
h. Return to Main Menu
Choose an option:
zsh
Python
OUTLINE
TIMELINE
Ln 803, Col 23
Spaces: 4
UTF-8
Python
3.9.6 64-bit

- **Delete Student:**

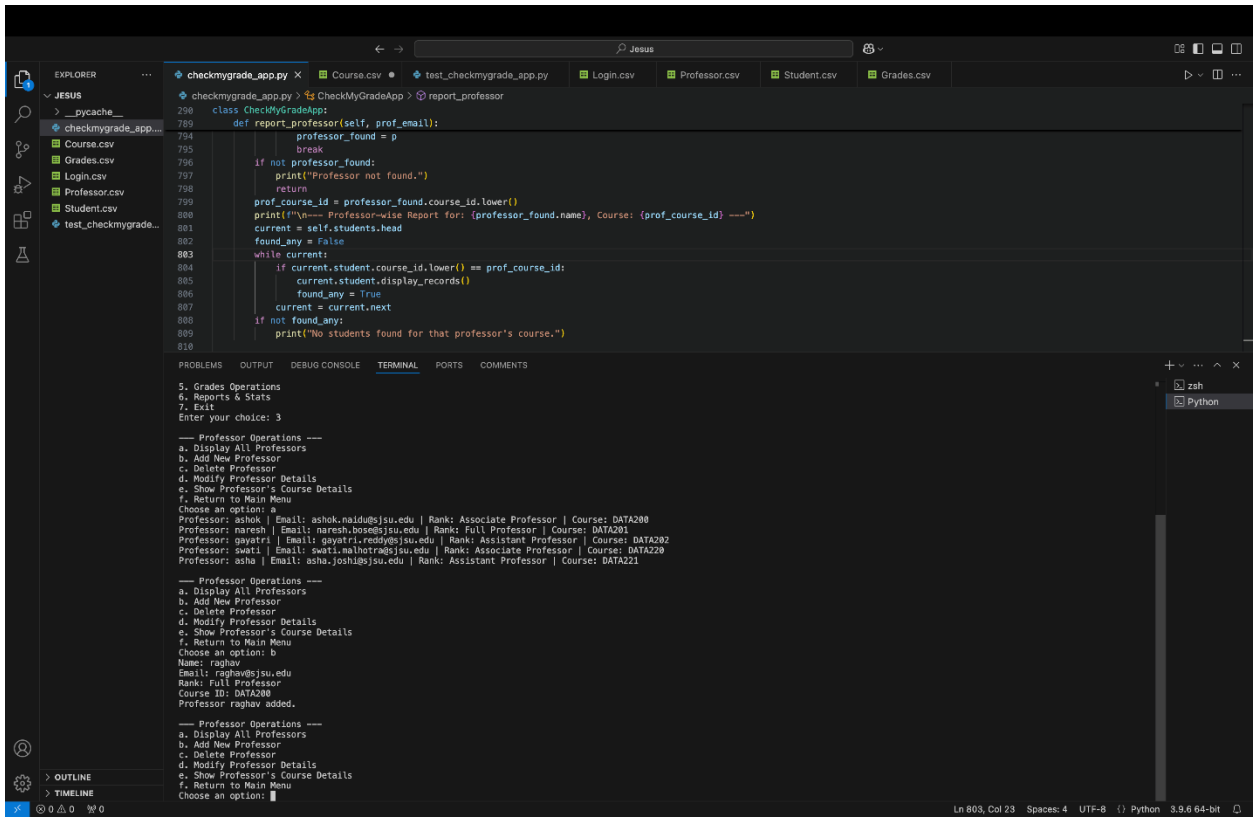


Terminal Screenshot 4: Professor Operations

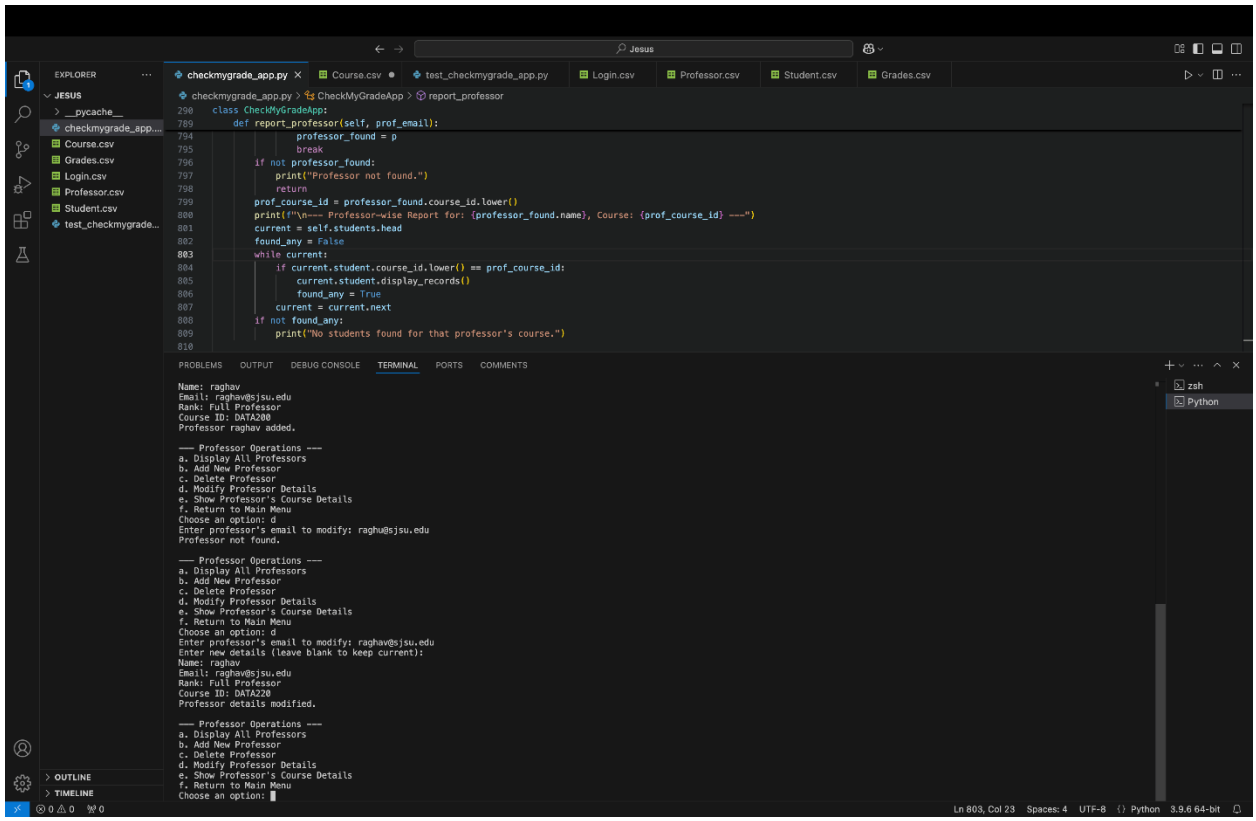
- **Display All Professors:**



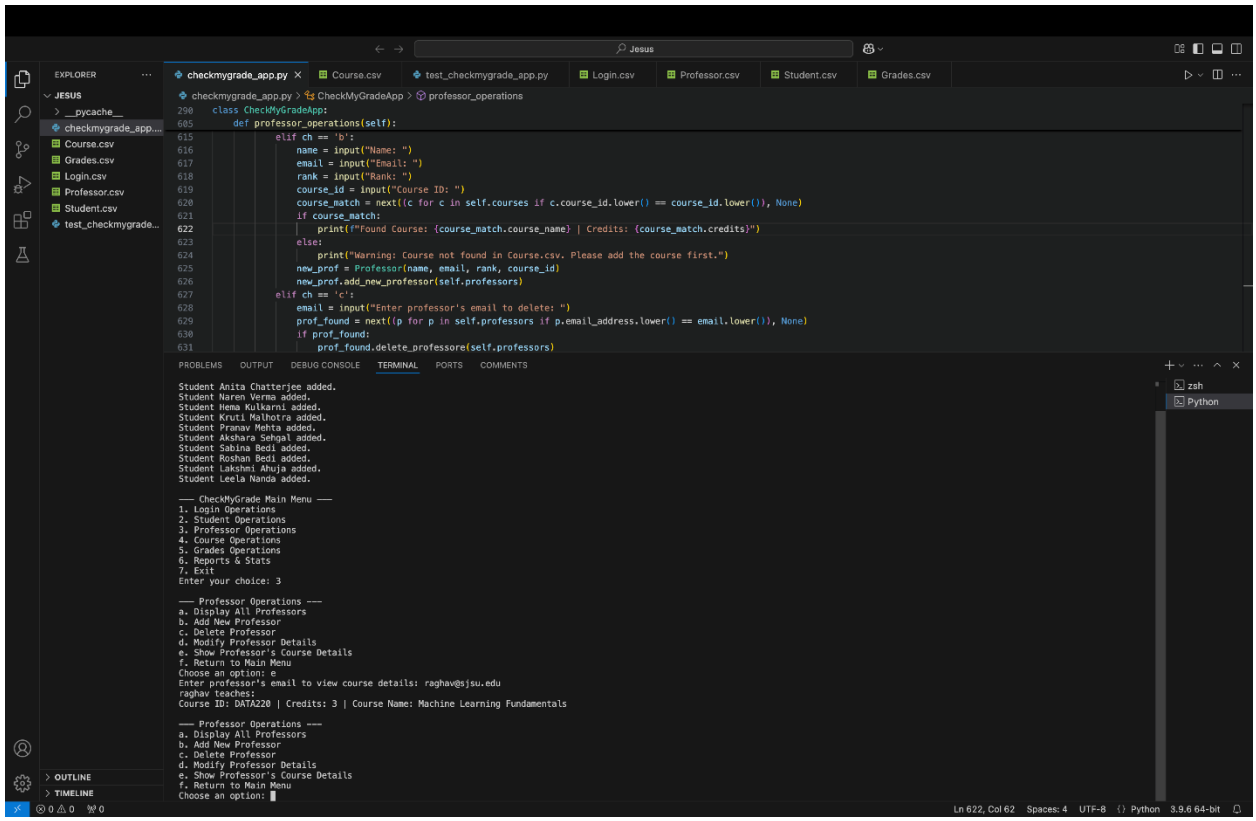
- **Add New Professor:**



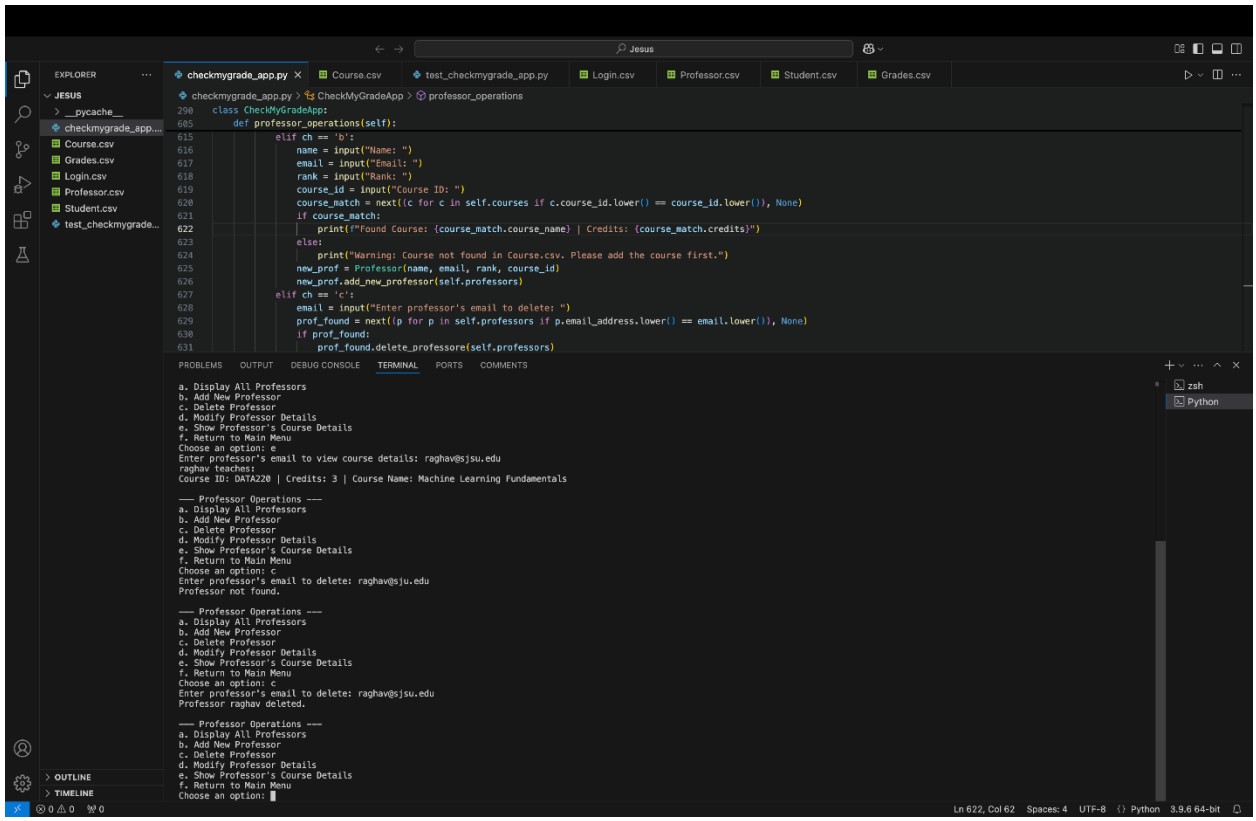
- **Modify Professor Details:**



- **Show Professor Course Details:**

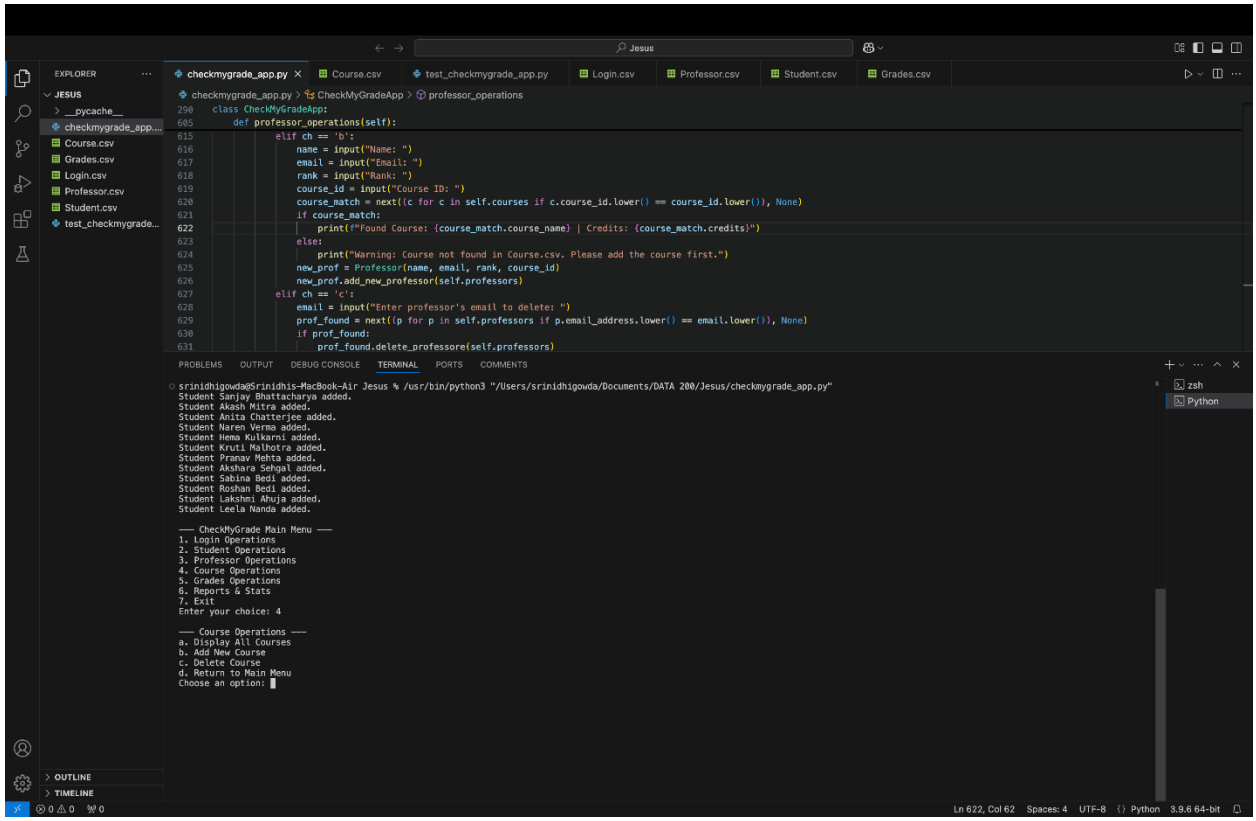


- **Delete Professor:**

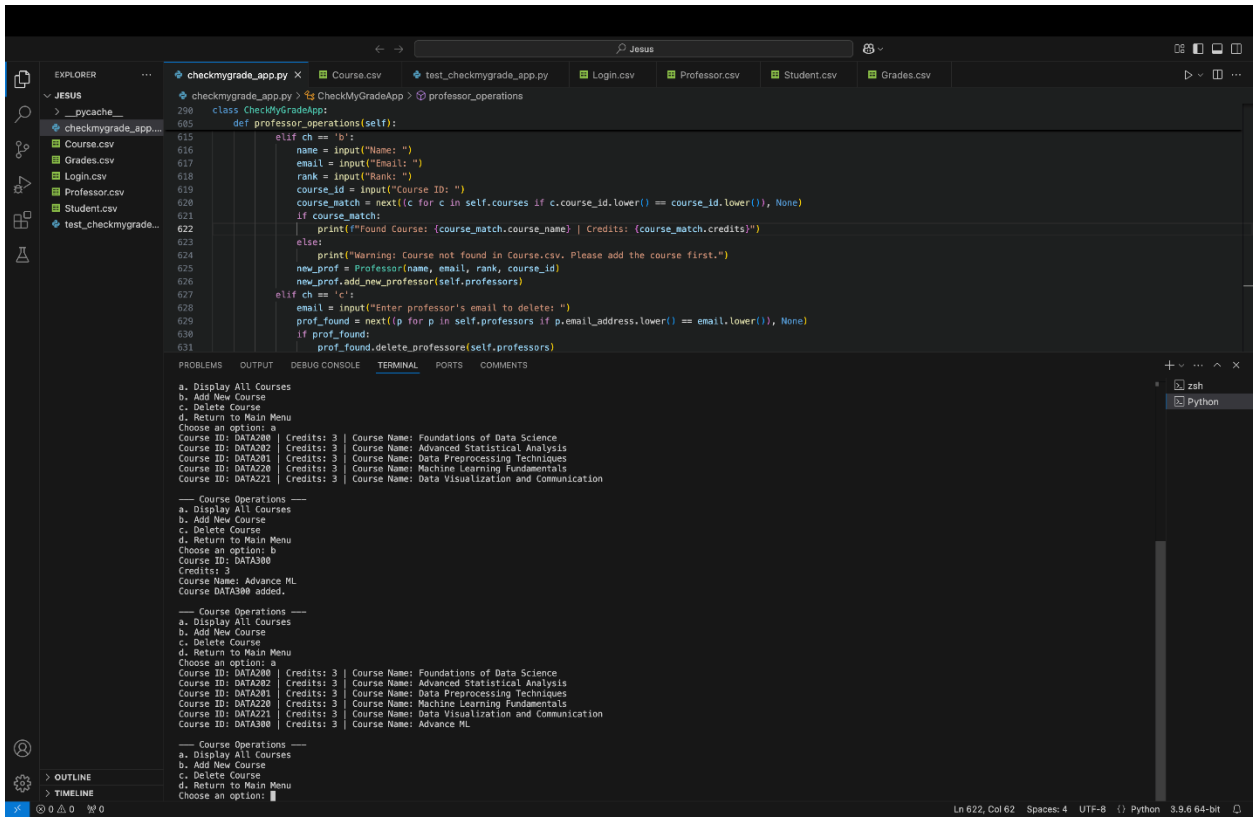


Terminal Screenshot 5: Course Operations

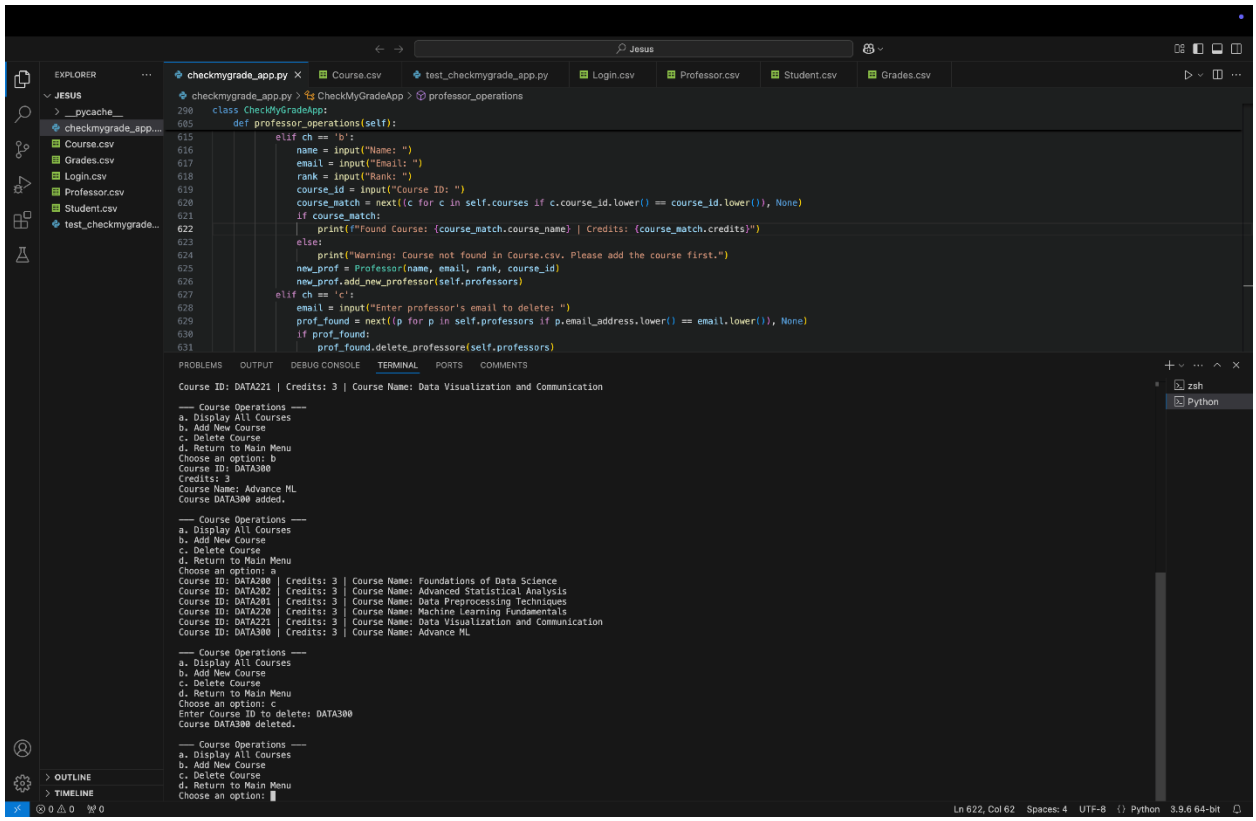
- **Display All Courses:**



- **Add New Course and Display:**

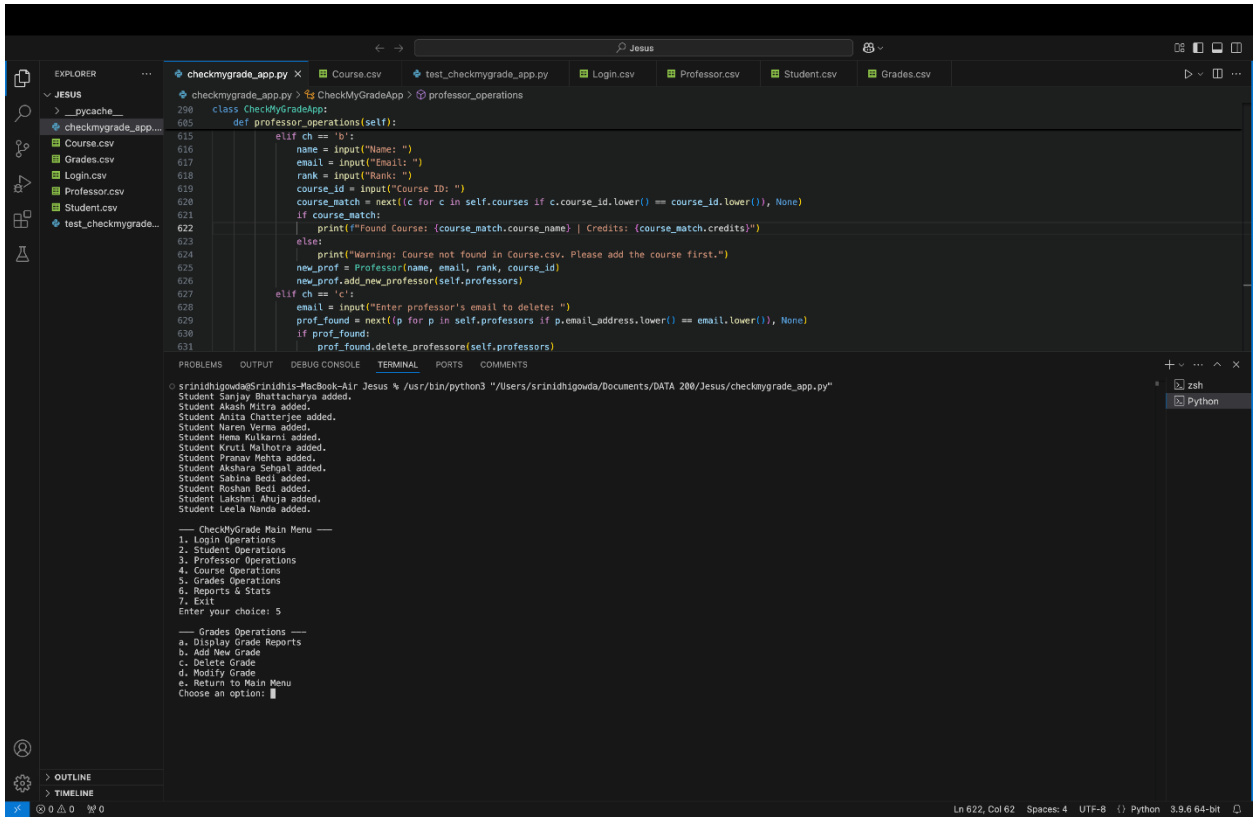


- **Delete Course:**

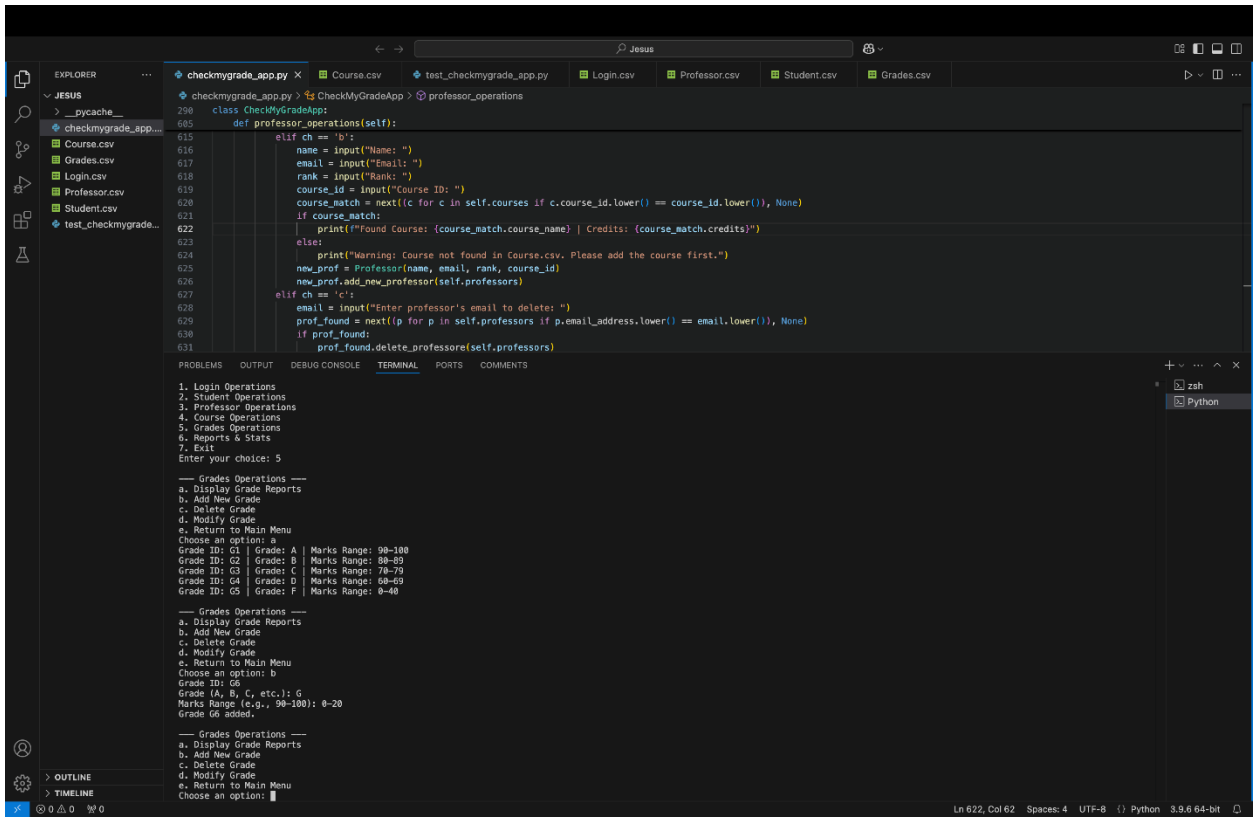


Terminal Screenshot 6: Grade Operations

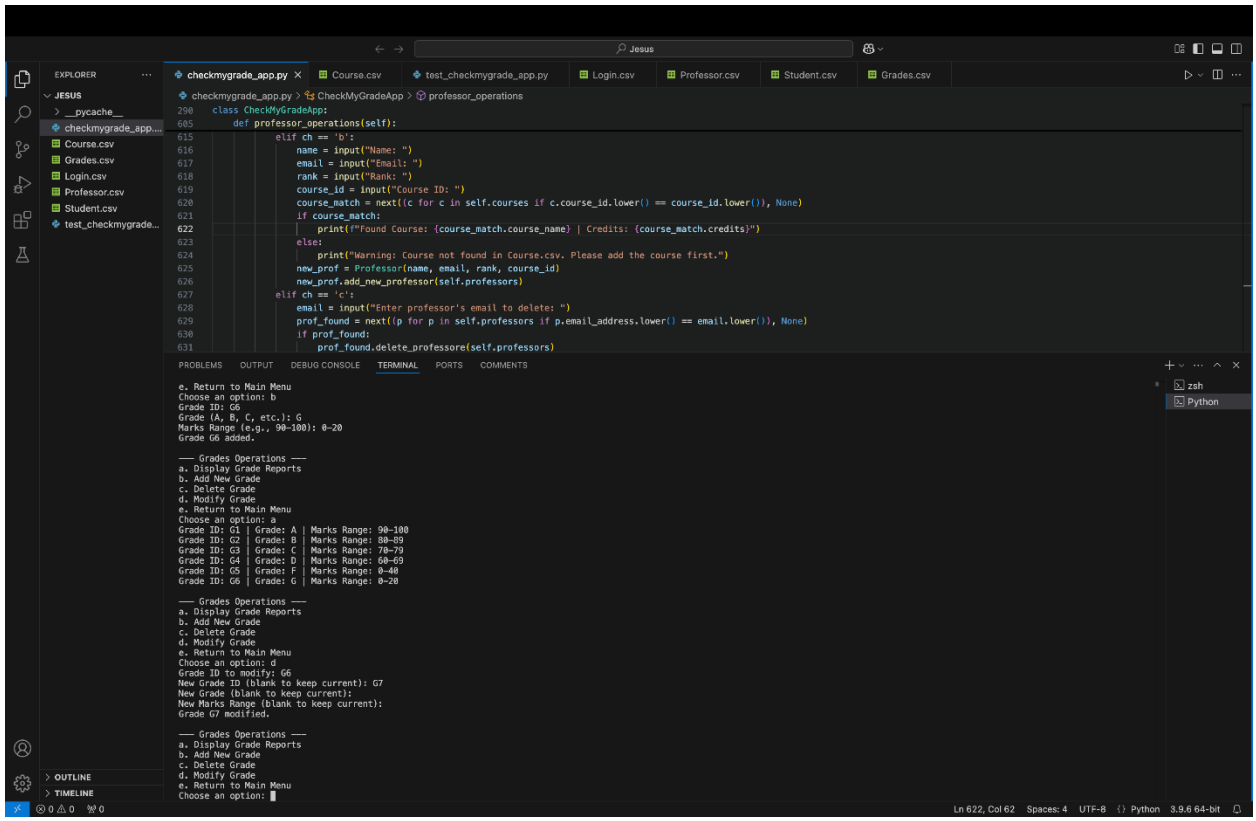
- **Display Grade Operations:**



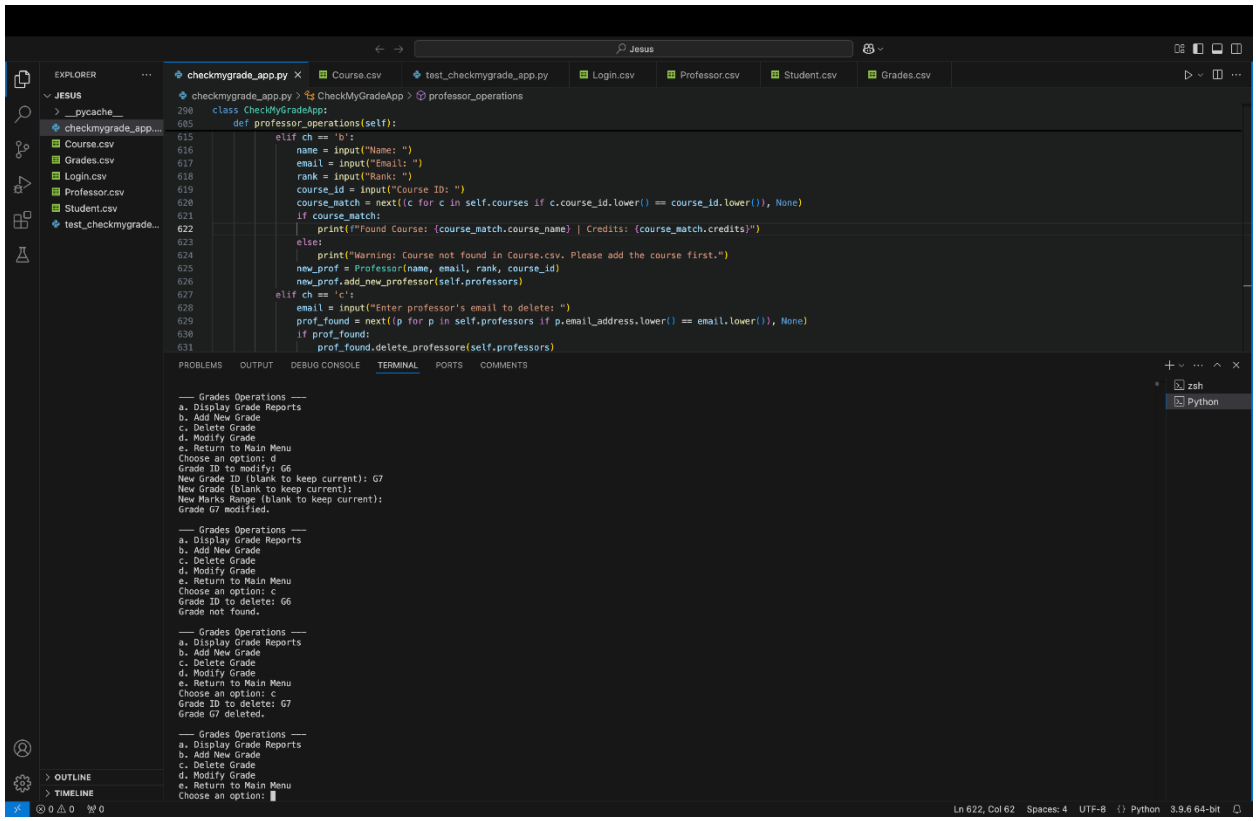
- **Adding a New Grade:**



- **Modifying a Grade:**

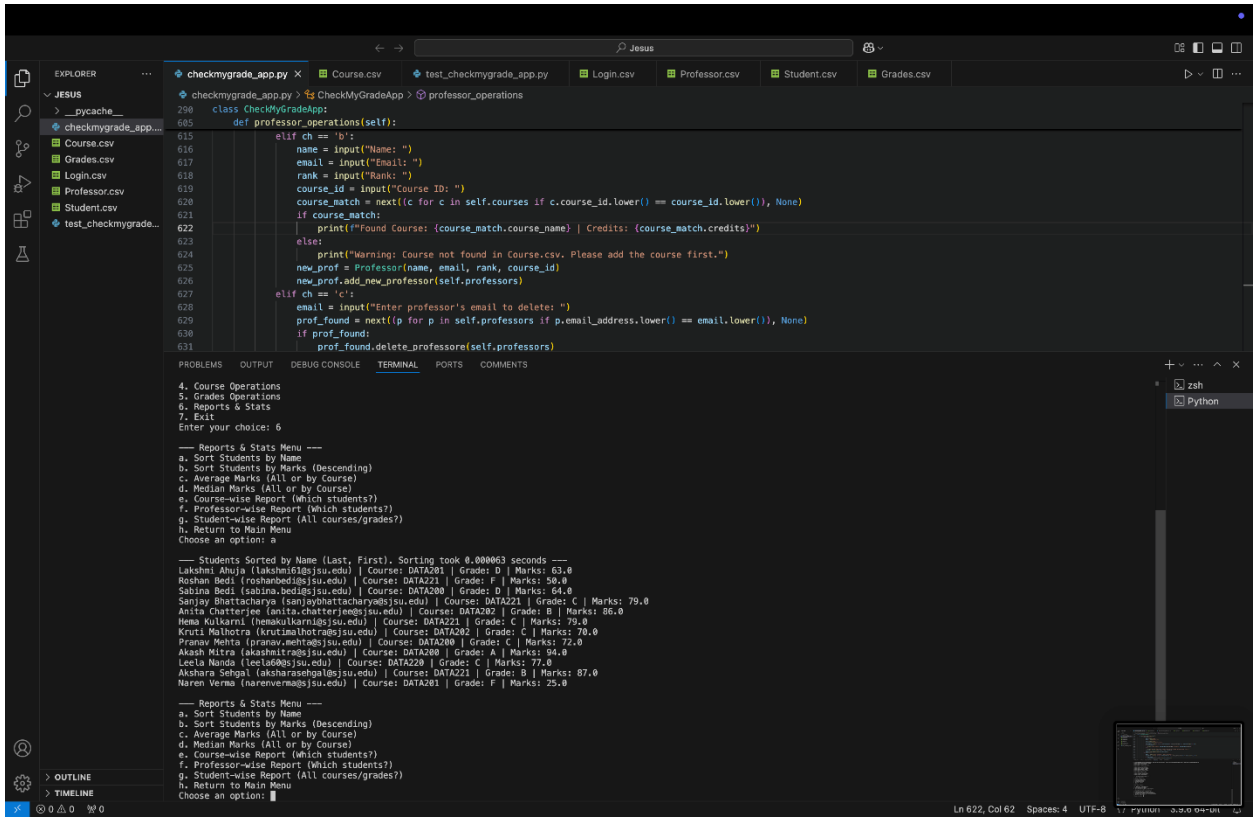


- **Delete a Grade:**

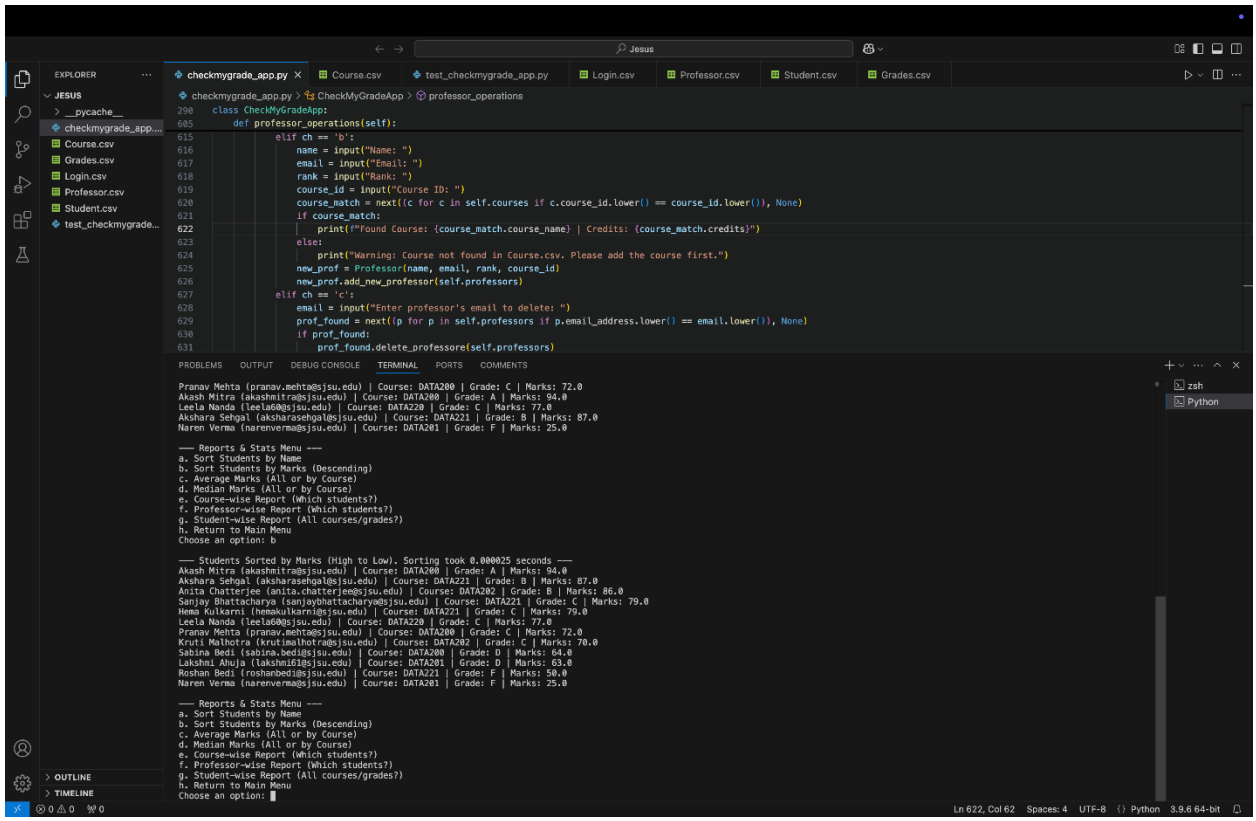


Terminal Screenshot 7: Reports & Stats Operations

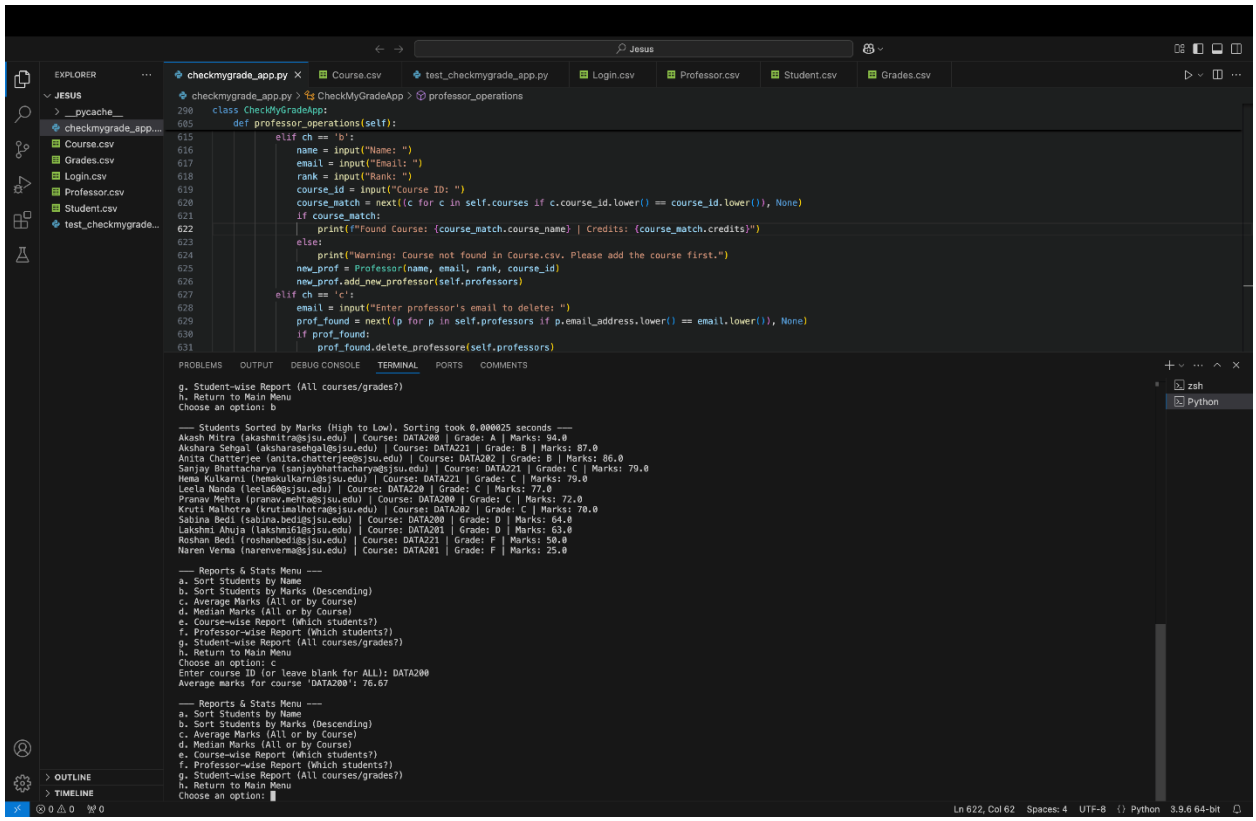
- **Sort by Name:**



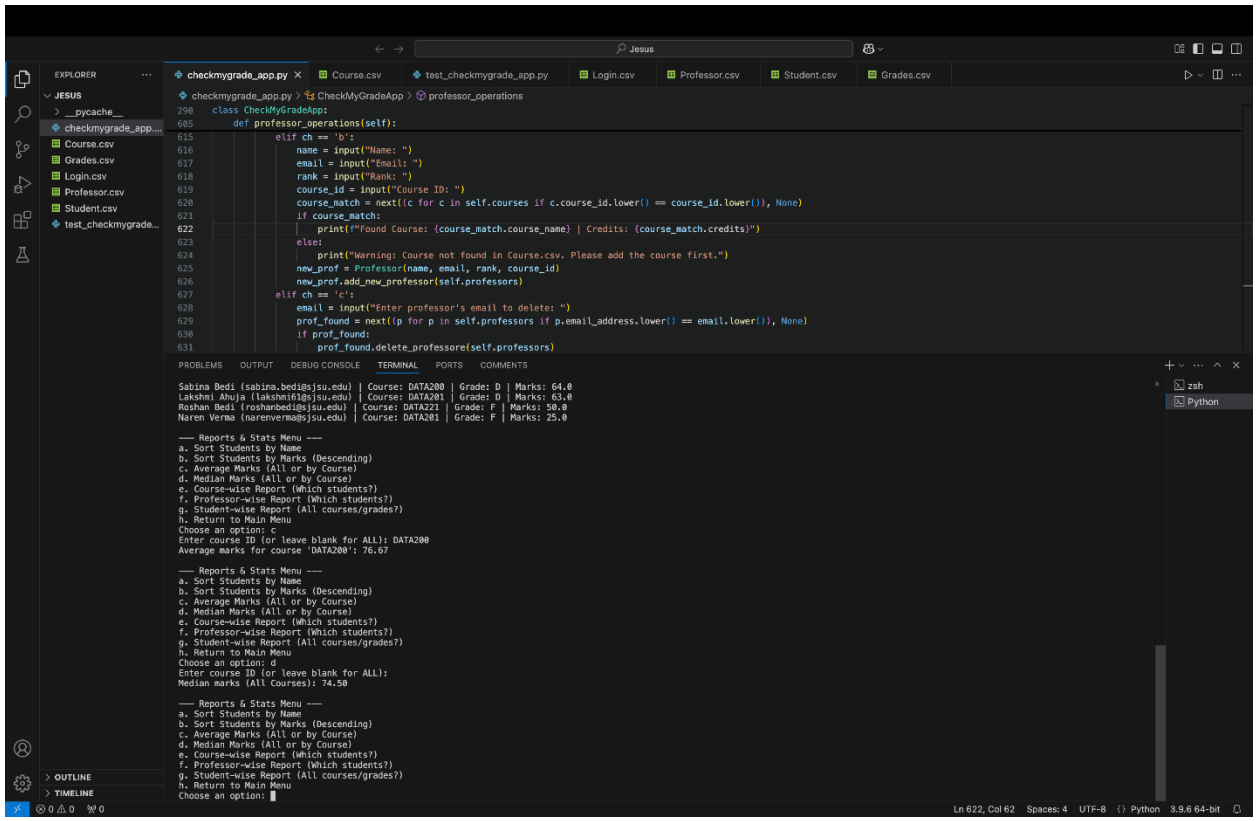
- **Sort by Student Marks:**



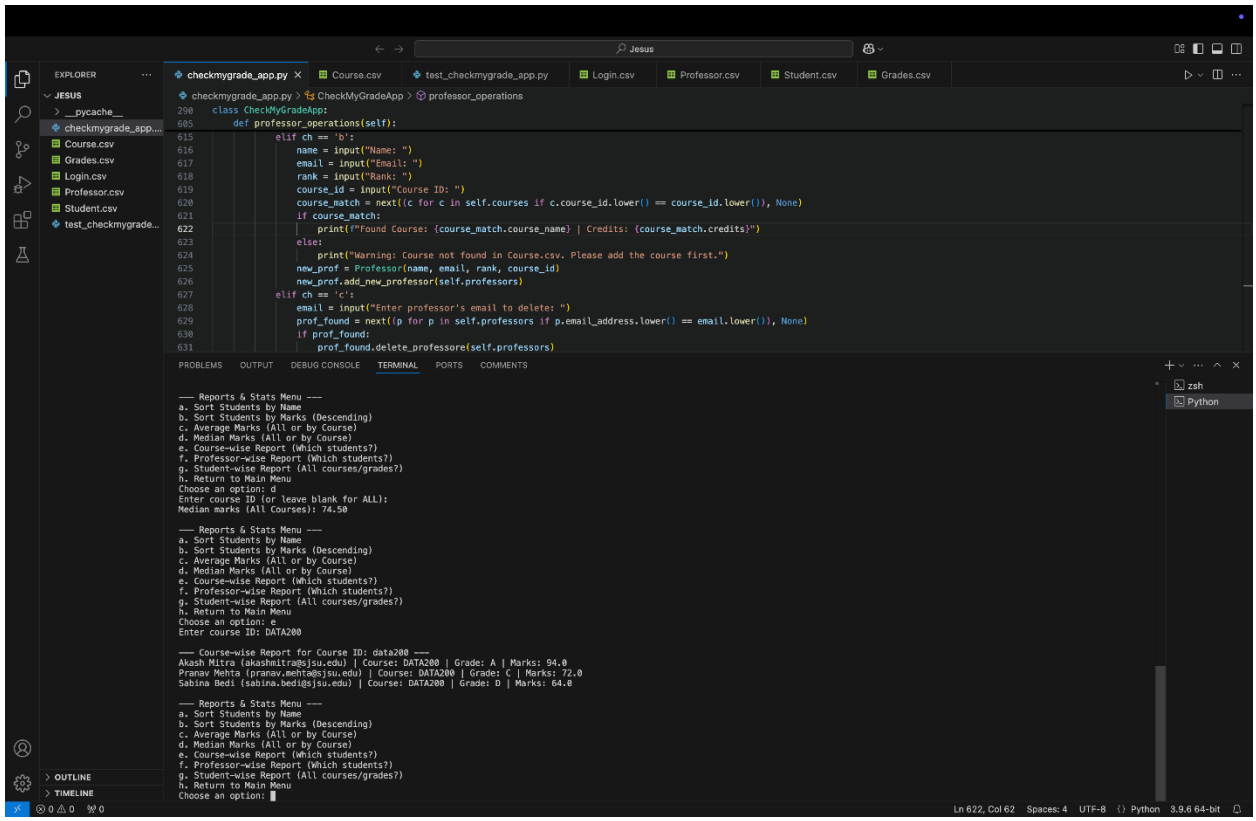
- **Average Marks of Course ID DATA200:**



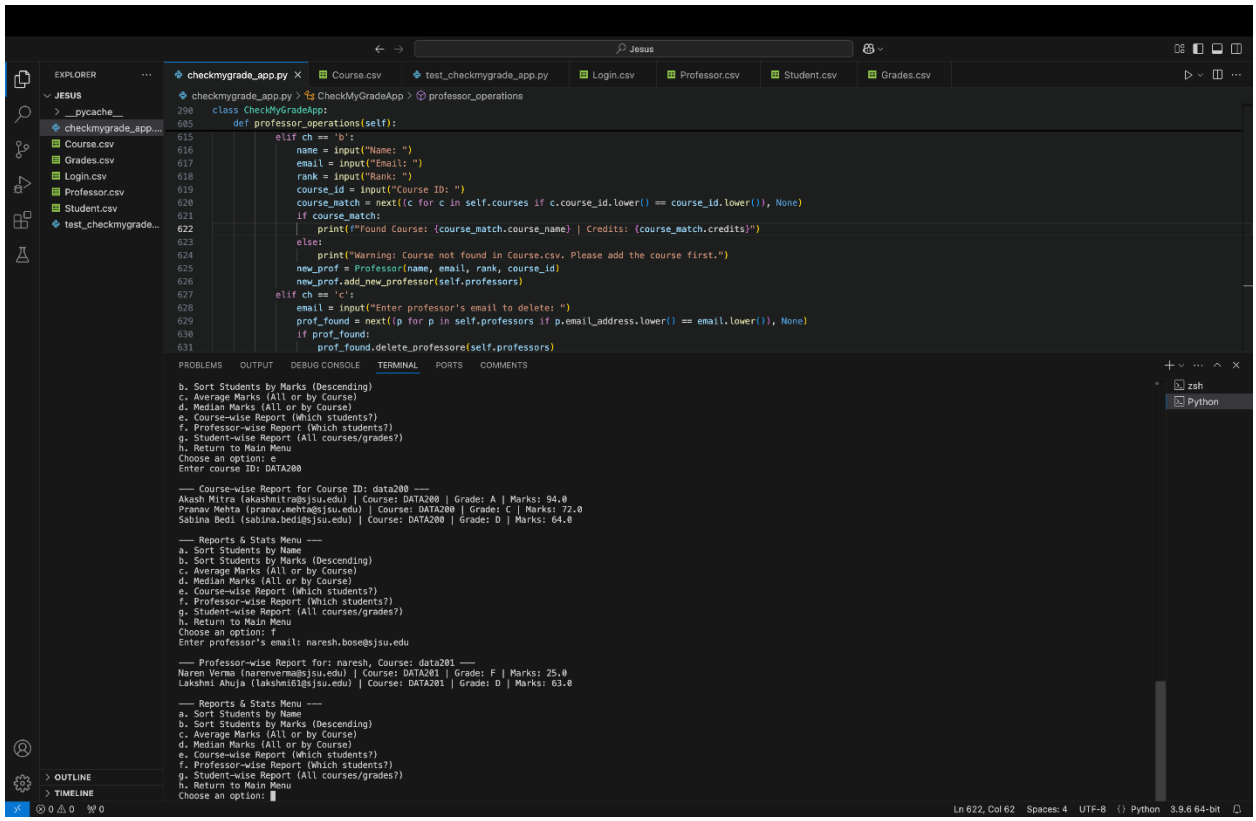
- **Median Marks of All Courses:**



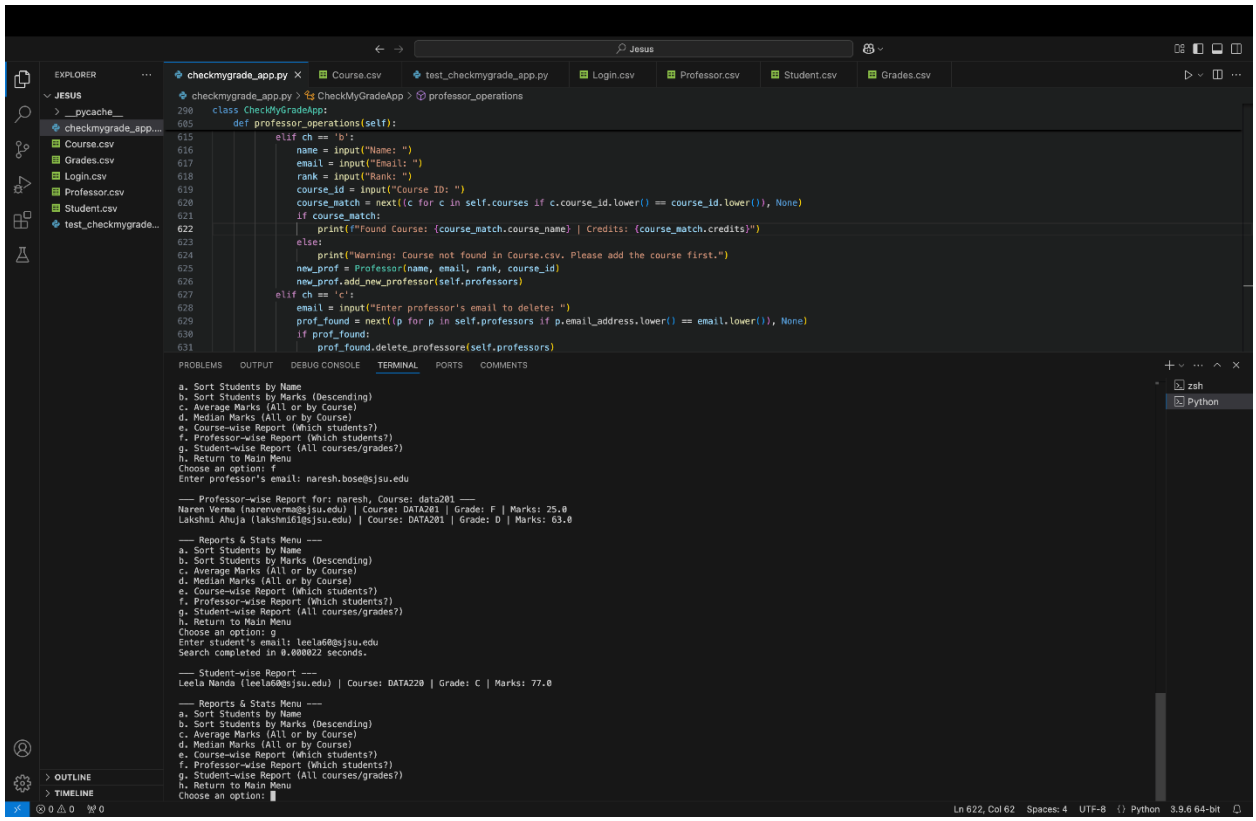
- **Course-wise Report:**



- **Professor-wise Report:**

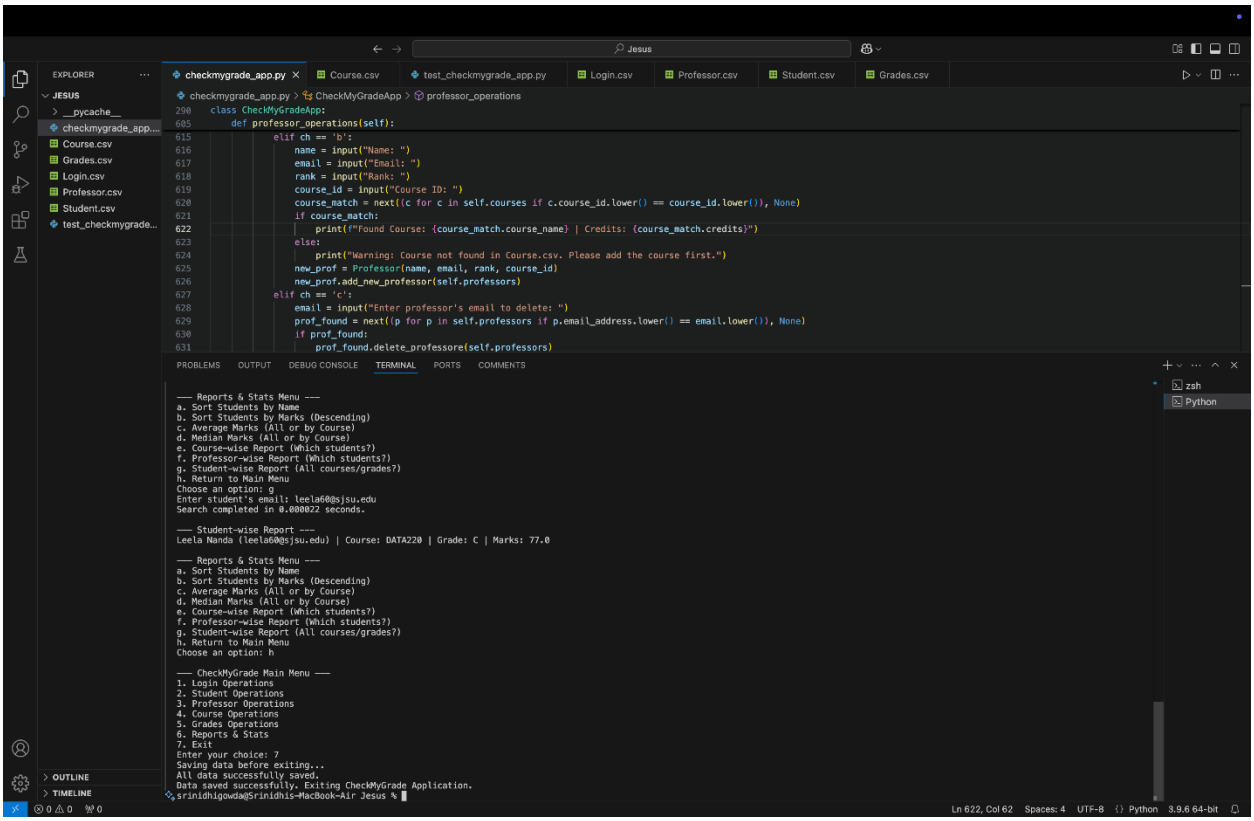


- **Student-wise Report:**



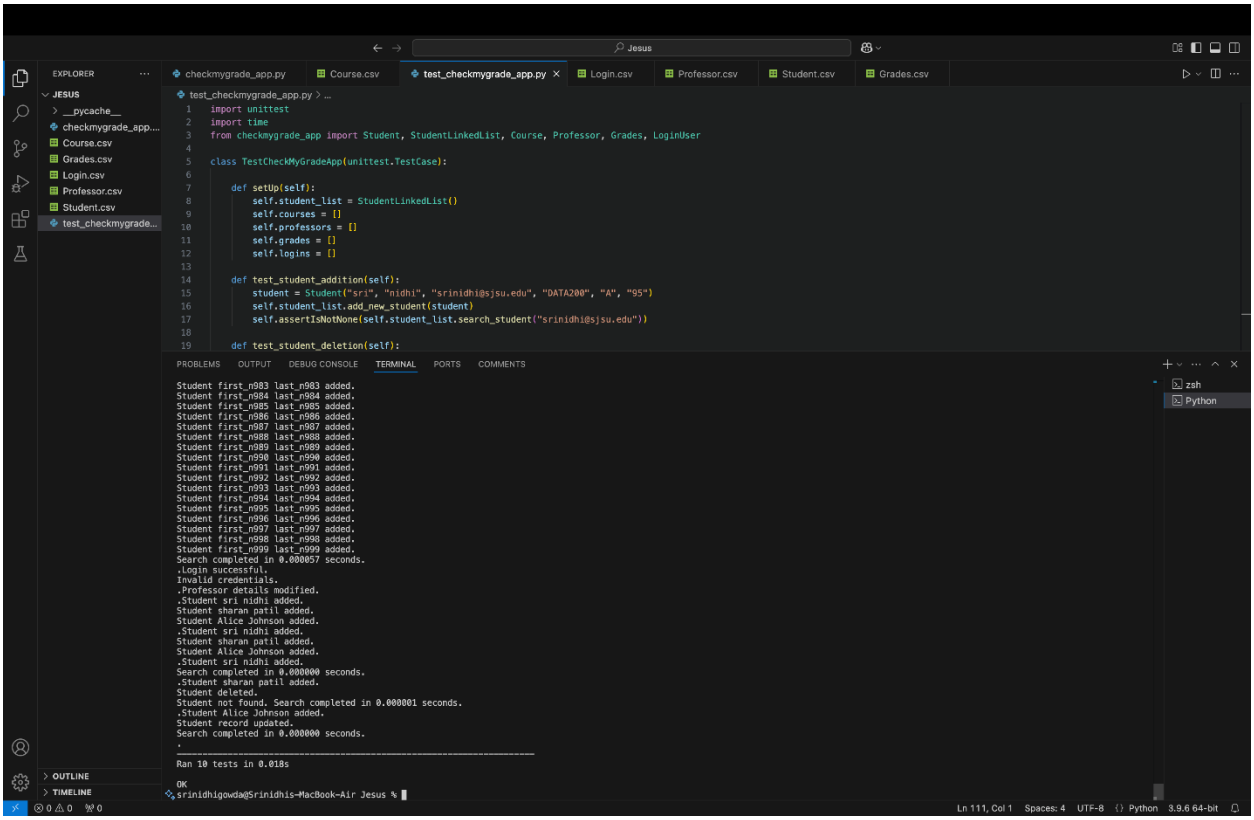
Terminal Screenshot 8: End of the Operations

At the end of the operation, when we exit the code, the program automatically saves all the data before exiting:



Terminal Screenshot 9: Unit Testing Operation

This tests the performance of the code:



Conclusion

The CheckMyGrade Application effectively meets the project requirements for managing student grades and related academic data. The modular design and robust implementation ensure that the system is both maintainable and scalable. Unit tests confirm that the major functionalities operate as expected, providing a solid foundation for further enhancements.

Appendices

- **Source Code:** Refer to checkmygrade_app.py and test_checkmyGrade_app.py for the complete codebase.
- **Data Files:** CSV files (Student.csv, Course.csv, Professor.csv, Grades.csv, Login.csv) store persistent data.
- **Class Diagram:** See the image above for the visual representation of the system architecture.
- **Terminal Screenshots:** All relevant terminal screenshots are embedded above.