❖Python Based Student Management System

Introduction

Background:

The Student Profile Management System is developed to assist educational institutions in managing student data efficiently. With an increasing number of students and the digitalization of educational systems in the educational institute, which makes manual system inefficient and more error prone. Through this system, administrative tasks can be streamlined, and operational efficiency can be enhanced.

Purpose of the System:

This system provides an interface for both the students and the administration. Through this system, administrators are capable of adding the students into the school, modifying the student's data, deleting the data of students and providing a general insight into the students' data. Similarly, through this system, the students can view their profiles, update their profiles, and view the charts of their grades, providing a platform for both the administrators and students for their own functions.

Scope:

The system includes functionality for user authentication, profile management, grade recording, extracurricular tracking, and visualization. It supports both the roles: Admin and Student.

System Overview

Description of the System:

The Student Profile Management System is a software developed using Python as its foundation. It operates through a Command Line Interface, meaning outputs are visible via the terminal in the IDE. It is a file-based management system that enables users to log in, manage student data, generate insights, and visualize performance.

Components and Features:

- User login and authentication
- Admin capabilities: Add student data, manage student data, generate insights of the data, and visualize the performance using Python
- Student capabilities: View/update profile of the student, view their grades, visualize the data
- Persistent storage using text files.

- Visualization through the use of Matplotlib

User Roles:

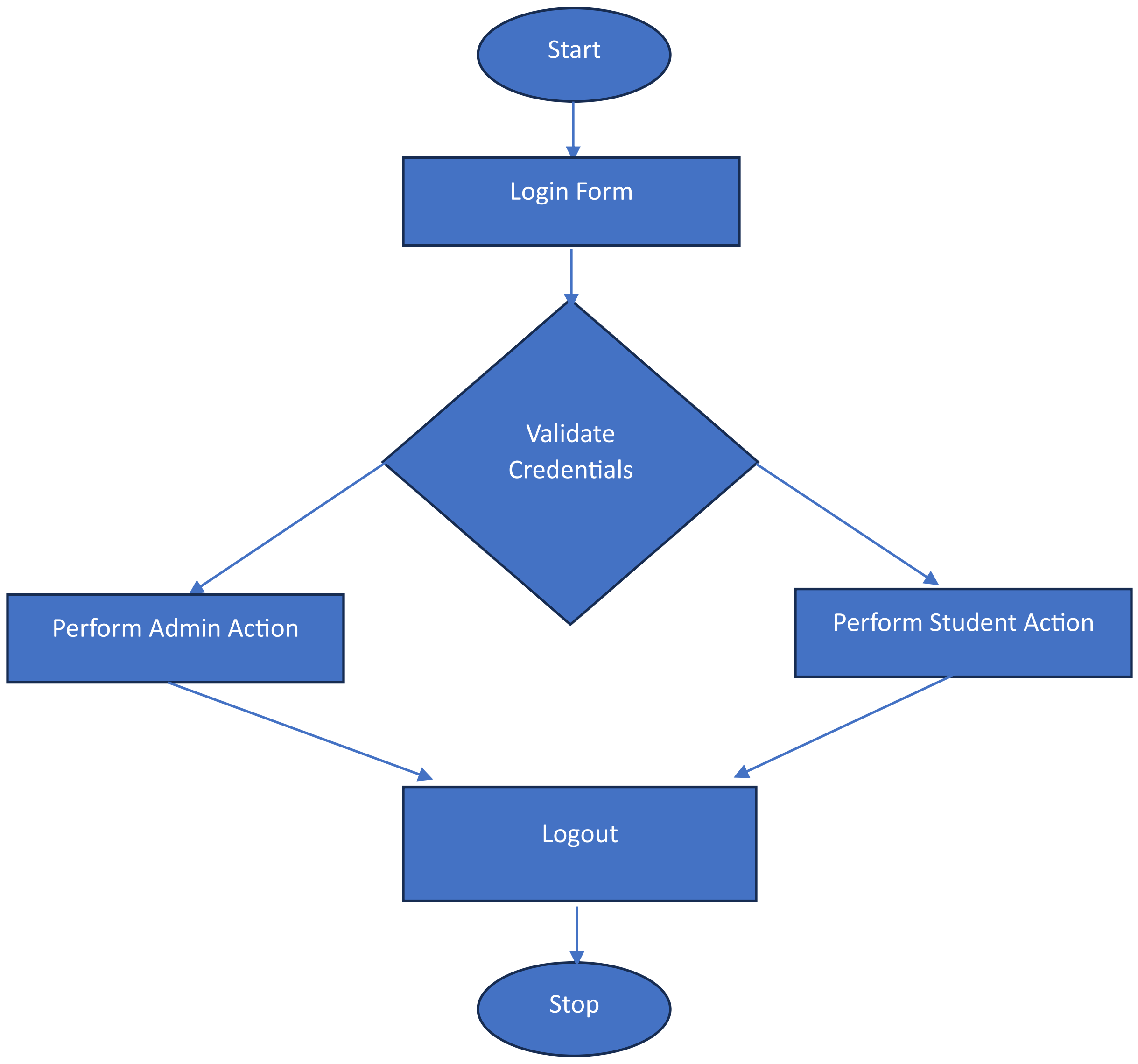
- Admin: Full access to student data and system analytics
- Student: Can view and update your own profile, view grades.

Design and Implementation

- **Data Flow Diagram**

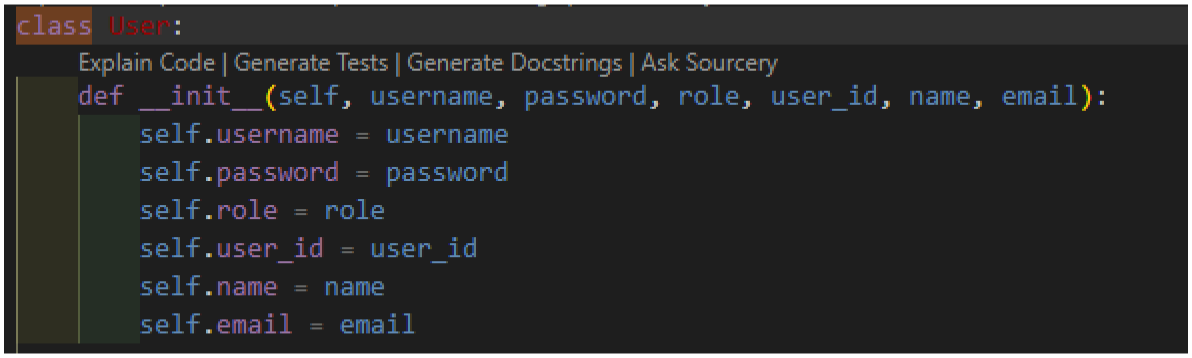
  Login → Role Check → Display Menu → Perform Actions → Update Files

- **Flowchart**



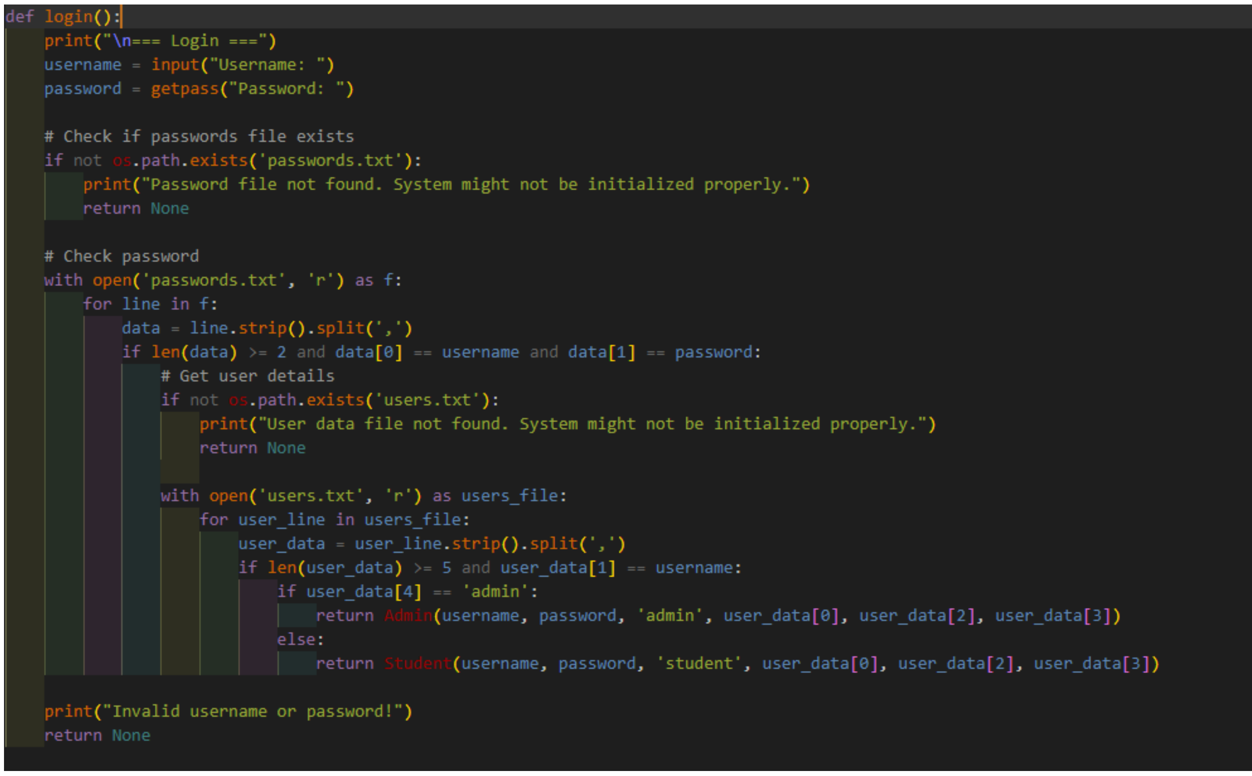
- **Class Diagram:**
    1. User: Base Class
    2. Student (User): Inherits User, including methods for profile and grade viewing
    3. Admin (User): Inherits User, including methods for data management

- Code Snippets:
    1. **User Class initialization:**


Explain Code | Generate Tests | Generate Docstrings | Ask Sourcery

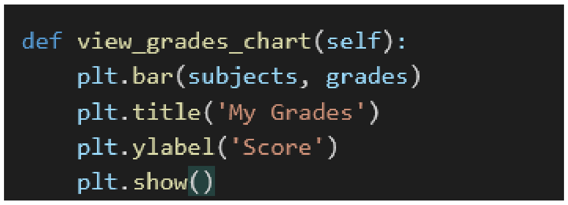
```python
class User:
    def __init__(self, username, password, role, user_id, name, email):
        self.username = username
        self.password = password
        self.role = role
        self.user_id = user_id
        self.name = name
        self.email = email
```

2. **Login authentication logic:**



```python
def login():
    print("\n=== Login ===")
    username = input("Username: ")
    password = getpass("Password: ")

    # Check if passwords file exists
    if not os.path.exists('passwords.txt'):
        print("Password file not found. System might not be initialized properly.")
        return None

    # Check password
    with open('passwords.txt', 'r') as f:
        for line in f:
            data = line.strip().split(',')
            if len(data) >= 2 and data[0] == username and data[1] == password:
                # Get user details
                if not os.path.exists('users.txt'):
                    print("User data file not found. System might not be initialized properly.")
                    return None

                with open('users.txt', 'r') as users_file:
                    for user_line in users_file:
                        user_data = user_line.strip().split(',')
                        if len(user_data) >= 5 and user_data[1] == username:
                            if user_data[4] == 'admin':
                                return Admin(username, password, 'admin', user_data[0], user_data[2], user_data[3])
                            else:
                                return Student(username, password, 'student', user_data[0], user_data[2], user_data[3])

    print("Invalid username or password!")
    return None
```

3. **Grade chart visualization using Matplotlib**



```
def view_grades_chart(self):
    plt.bar(subjects, grades)
    plt.title('My Grades')
    plt.ylabel('Score')
    plt.show()
```

- **Function Description:**
    1. Save_to_files: Saves user credentials and initializes empty records in data files.
    2. View_profile: Displays a student's personal information, grades, and ECAs.
    3. Update_profile: Allows a student to update their name or email.
    4. View_grades_chart: Graphically displays grades using a bar chart.
    5. Add_user: Admin function to register new users into the system.
    6. Modify_student_record: Admin function that allows editing student info, grades, or ECAs
    7. _modify_student_info: Updates a students' username, name, or email address.
    8. _update_username_in_passwords: Synchronizes updated usernames across credential files.
    9. _modify_grades: Updates individual subject grades for a student.
    10. _modify_eca: Updates a student's extracurricular activity list.
    11. Delete_student: Deletes a student's entire record from all files.
    12. Generate_insights: Allows the admin to access data analytics
    13. _average_grades: Computes average grade per subject
    14. _most_active_students: Displays students with the most extracurricular activities
    15. _grade_distribution: Uses boxplot to show the grade spread per subject
    16. Initialize_files: Ensures required text files exist and creates a default admin account.

17. Username_from_id: Fetches the username for a given user ID

18. Student_menu and admin_menu: Handle role-specific menu navigation

19. main: Main execution points to start the system.

User Guides

- **How to Use the System:**
  1. Run the Python script.

  2. Login using the credentials

  3. For Admin: Username: admin, password: admin123

  4. For Student: Login As Admin, add new user, register as student and login using the student username and password

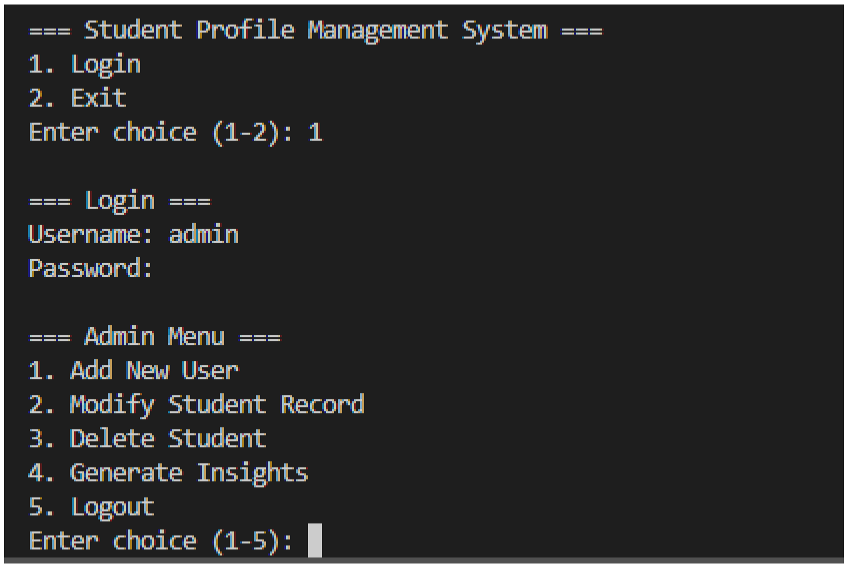
- Step-By-Step Instructions for Each Features:

  1. Admin Login → Admin Menu → Select Task (Add User, Modify, Delete, Insights)

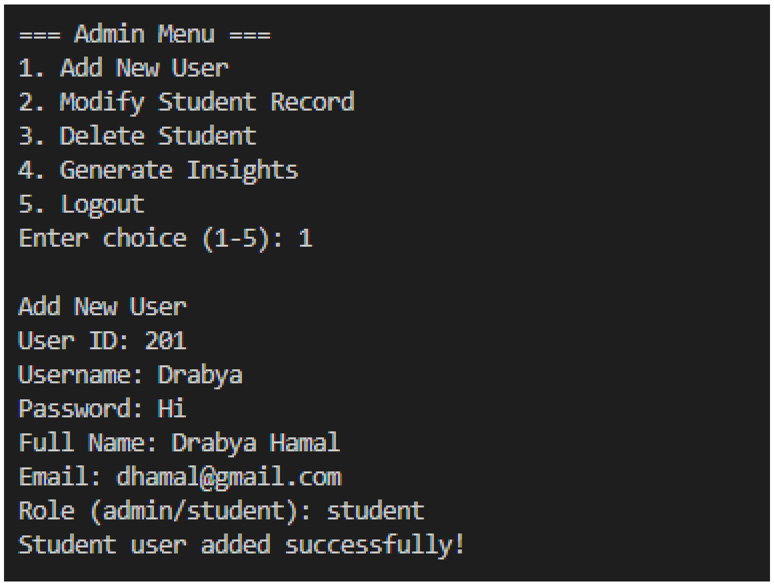
  2. Student Login → Student Menu → Select Task (View/Update Profile, View Grades)

Testing

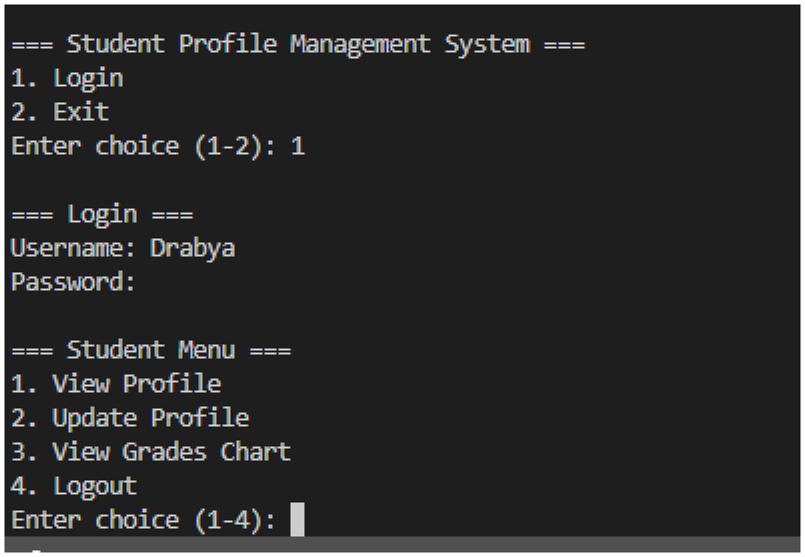
- **Test Cases and Scenarios:**



```
=== Student Profile Management System ===
1. Login
2. Exit
Enter choice (1-2): 1

=== Login ===
Username: admin
Password:

=== Admin Menu ===
1. Add New User
2. Modify Student Record
3. Delete Student
4. Generate Insights
5. Logout
Enter choice (1-5):
```



```
=== Admin Menu ===
1. Add New User
2. Modify Student Record
3. Delete Student
4. Generate Insights
5. Logout
Enter choice (1-5): 1

Add New User
User ID: 201
Username: Drabya
Password: Hi
Full Name: Drabya Hamal
Email: dhamal@gmail.com
Role (admin/student): student
Student user added successfully!
```



```
=== Student Profile Management System ===
1. Login
2. Exit
Enter choice (1-2): 1

=== Login ===
Username: Drabya
Password:

=== Student Menu ===
1. View Profile
2. Update Profile
3. View Grades Chart
4. Logout
Enter choice (1-4):
```

Security Measures

- Password Management: Passwords are stored in a separate file
- User Management: Users are stored in a separate file.
- Grade Management: Grades of the students are stored in a separate file.
- User Authentication: Login credentials are validated before granting access.

Future Enhancements

- Proposed Features

    1. Use of database instead of text files.
    2. Roles for teachers.
    3. GUI-based interface.

System Performance and Scalability

- **Performance Evaluation:** Performance Performs well with a small amount of data.
- **Scalability Analysis:** Performance may degrade with large files; using a database would be better.

User Feedback and Usability Study

- **Feedback Collection Process:** Manual testing by users using printed instructions
- **Usability Study Methodology:** Observational feedback from testers.
- **Analysis of User Feedback:** System is easy to use, but improvements needed in password security and file management.

Legal and Ethical Considerations

- **Compliance with Data Protection Regulation:** The system adheres to data protection regulations, including the General Data Protection Regulation (GDPR) and the Family Educational Rights and Privacy Act (FERPA), to ensure compliance with data privacy and security requirements.
- **Ethical Use of Student Data:** No sensitive information is asked or input. Most of the values are just random values which is inputted by the programmers. User consent is obtained for data collection, processing and sharing, and transparent data handling practices are followed to ensure user trust and confidence in the system.

Maintenance and Support

- A systematic bug tracking and resolution process is implemented to identify, prioritize, and address software defects and issues.
- There is manual testing and debugging.
- **Software Update Process:** Updates involve editing the Python files and re-running tests.

Conclusion

With ongoing development and enhancements, the Student Management System needs to evolve further, incorporating new features, new facilities and more practices to meet the evolving needs of the educational system.

The system fulfills core requirements of student data management with clear potential for upgrades.

References

- Python Official Documentation
- Matplotlib Documentation
- File Handling in Python
- Object-Orientation Programming Guides
- GeeksforGeeks : https://www.geeksforgeeks.org/python-projects-beginner-to-advanced/?ref=shm
- W3Schools: https://www.w3schools.com/python/
- CodewithHarrry: https://www.youtube.com/@CodeWithHarry

Work Division

1). Structure of the project

The structure of the project was designed by Drabya Hamal. He decided on the structure and made a flowchart as well as documentation of the project. He is responsible for the errors and the explanation of the program.

2)OOP

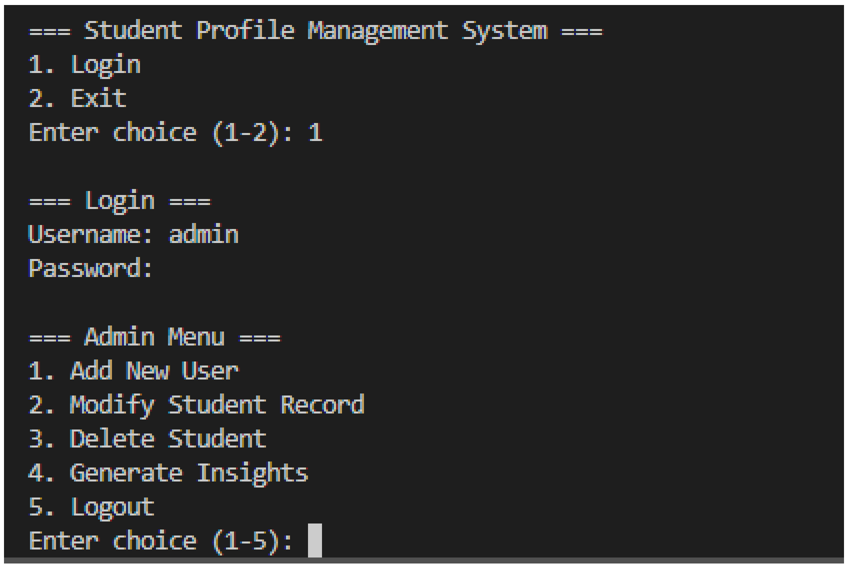
Classes and Objects with Definitions were decided by Lokesh Bhatta. The variables and functions were all looked at and researched by him. He is also responsible for the different function names used in the program.

3) File handling

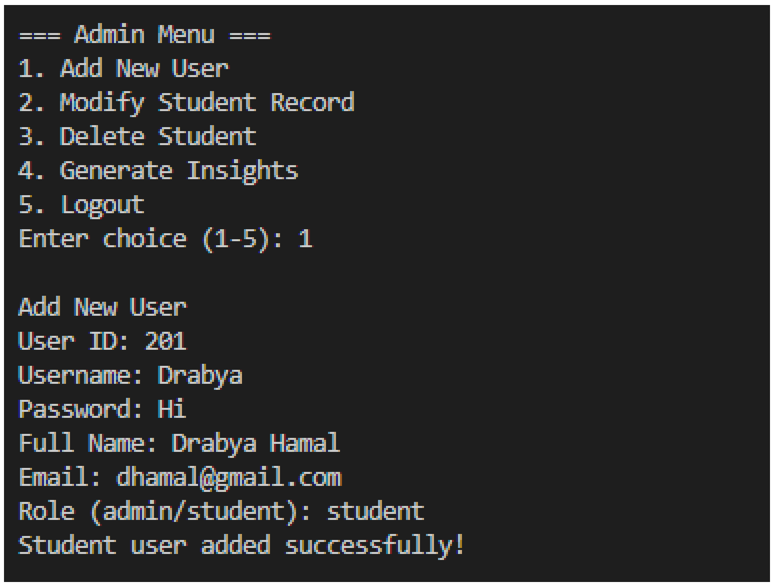
File handling was all handled by Reshav Karna. Task related to store, retrieval and managing all the important data was done by him. Similarly, the different files were also made by him, and he is responsible for all the changes in the files. He is also responsible for the presentation.

Although the tasks were divided, the work was done with one another sitting next to one another, helping each other complete the task.

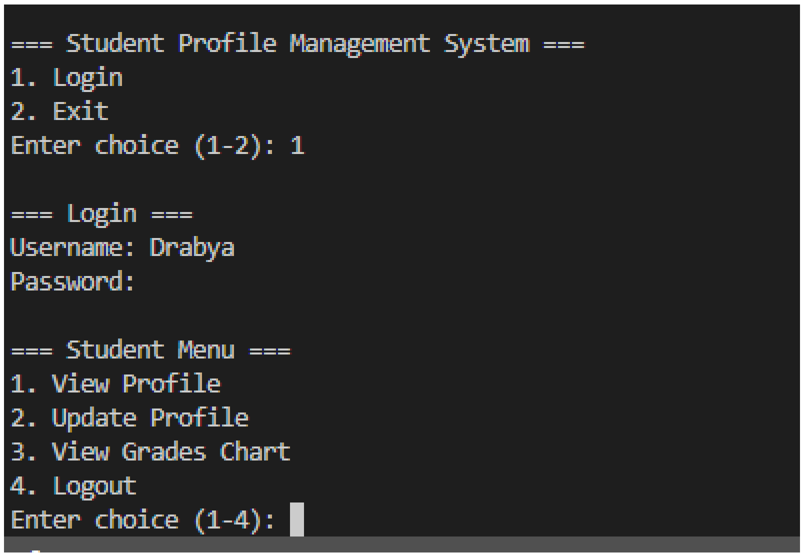
Output



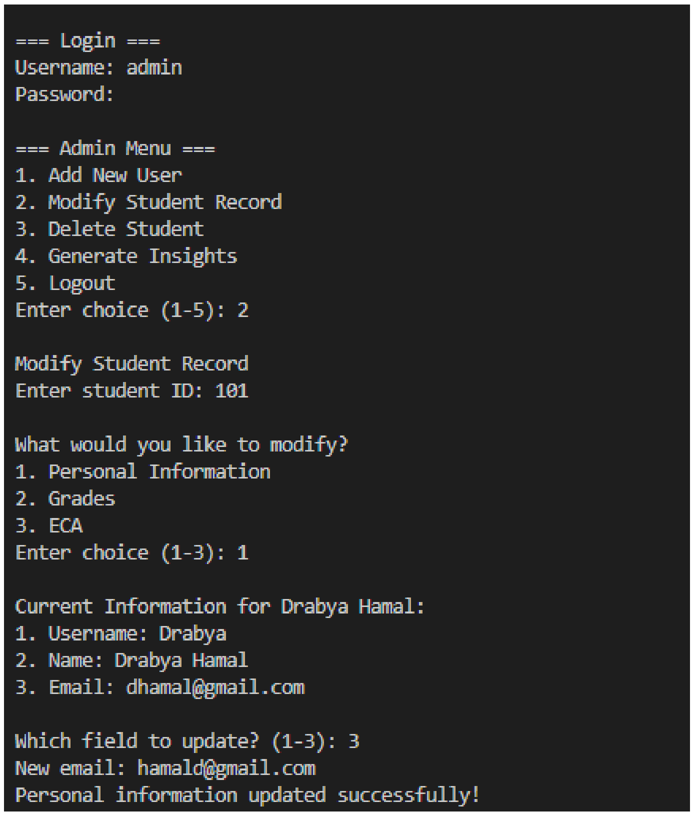
```
=== Student Profile Management System ===
1. Login
2. Exit
Enter choice (1-2): 1

=== Login ===
Username: admin
Password:

=== Admin Menu ===
1. Add New User
2. Modify Student Record
3. Delete Student
4. Generate Insights
5. Logout
Enter choice (1-5):
```



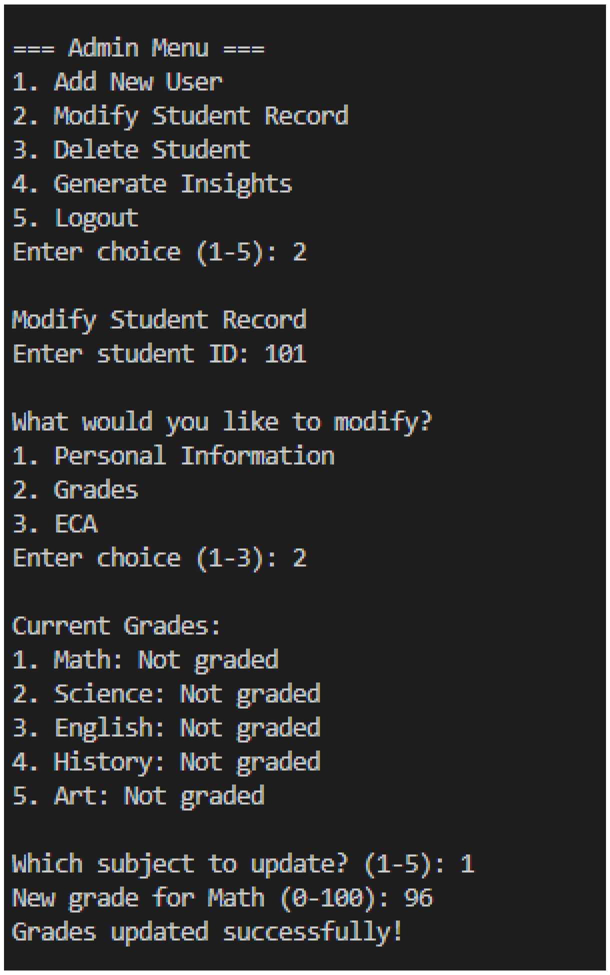
```
=== Admin Menu ===
1. Add New User
2. Modify Student Record
3. Delete Student
4. Generate Insights
5. Logout
Enter choice (1-5): 1

Add New User
User ID: 201
Username: Drabya
Password: Hi
Full Name: Drabya Hamal
Email: dhamal@gmail.com
Role (admin/student): student
Student user added successfully!
```



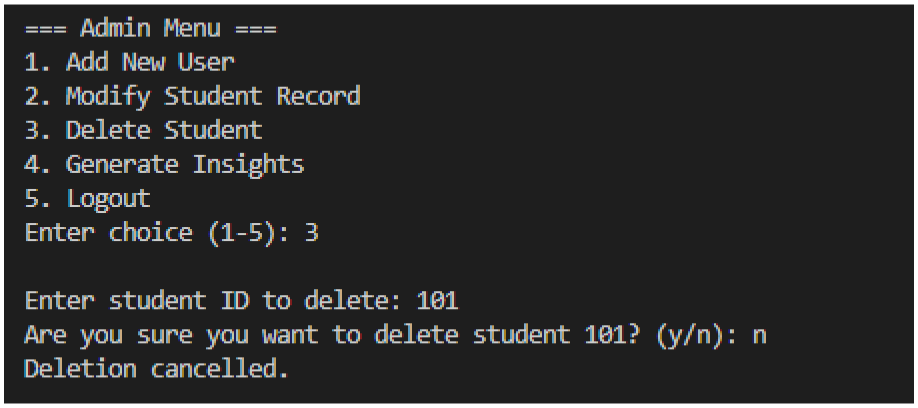
```
=== Student Profile Management System ===
1. Login
2. Exit
Enter choice (1-2): 1

=== Login ===
Username: Drabya
Password:

=== Student Menu ===
1. View Profile
2. Update Profile
3. View Grades Chart
4. Logout
Enter choice (1-4):
```



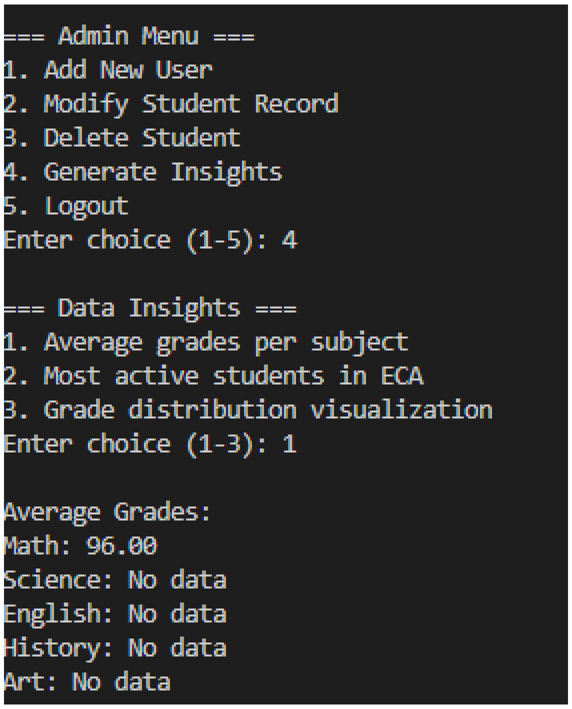
```
=== Login ===
Username: admin
Password:

=== Admin Menu ===
1. Add New User
2. Modify Student Record
3. Delete Student
4. Generate Insights
5. Logout
Enter choice (1-5): 2

Modify Student Record
Enter student ID: 101

What would you like to modify?
1. Personal Information
2. Grades
3. ECA
Enter choice (1-3): 1

Current Information for Drabya Hamal:
1. Username: Drabya
2. Name: Drabya Hamal
3. Email: dhamal@gmail.com

Which field to update? (1-3): 3
New email: hamald@gmail.com
Personal information updated successfully!
```



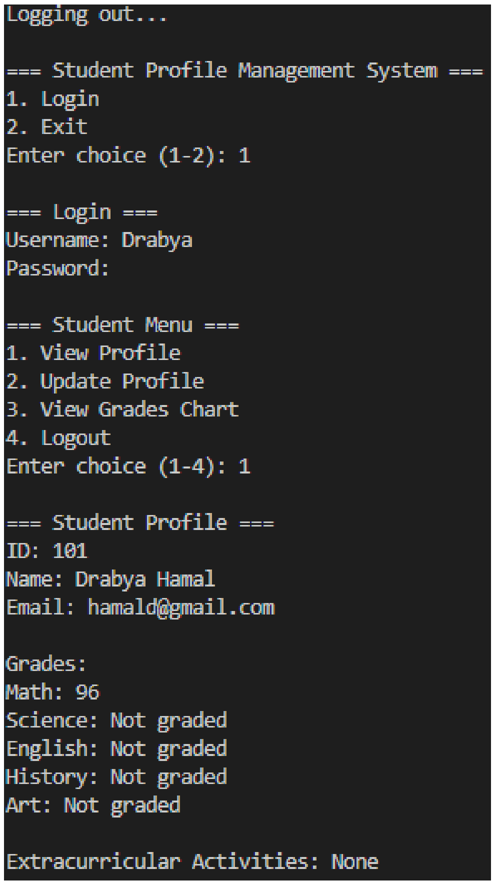
```
=== Admin Menu ===
1. Add New User
2. Modify Student Record
3. Delete Student
4. Generate Insights
5. Logout
Enter choice (1-5): 2

Modify Student Record
Enter student ID: 101

What would you like to modify?
1. Personal Information
2. Grades
3. ECA
Enter choice (1-3): 2

Current Grades:
1. Math: Not graded
2. Science: Not graded
3. English: Not graded
4. History: Not graded
5. Art: Not graded

Which subject to update? (1-5): 1
New grade for Math (0-100): 96
Grades updated successfully!
```



```
=== Admin Menu ===
1. Add New User
2. Modify Student Record
3. Delete Student
4. Generate Insights
5. Logout
Enter choice (1-5): 3

Enter student ID to delete: 101
Are you sure you want to delete student 101? (y/n): n
Deletion cancelled.
```



```
=== Admin Menu ===
1. Add New User
2. Modify Student Record
3. Delete Student
4. Generate Insights
5. Logout
Enter choice (1-5): 4

=== Data Insights ===
1. Average grades per subject
2. Most active students in ECA
3. Grade distribution visualization
Enter choice (1-3): 1

Average Grades:
Math: 96.00
Science: No data
English: No data
History: No data
Art: No data
```



```
Logging out...

=== Student Profile Management System ===
1. Login
2. Exit
Enter choice (1-2): 1

=== Login ===
Username: Drabya
Password:

=== Student Menu ===
1. View Profile
2. Update Profile
3. View Grades Chart
4. Logout
Enter choice (1-4): 1

=== Student Profile ===
ID: 101
Name: Drabya Hamal
Email: hamald@gmail.com

Grades:
Math: 96
Science: Not graded
English: Not graded
History: Not graded
Art: Not graded

Extracurricular Activities: None
```