SS-Grade

Software Design Requirements

Date: March 25, 2024

Yi Qian Kang

Table of Contents

1. Introduction

1.1 Purpose

Our team built and designed an online student academic record using PHP, HTML, CSS, JavaScript, and MySQL. Our online SS-Grade provides an easy-to-use user interface that allows customers to quickly register, log in, find student information, update student grades, calculate final grades, and delete student information.

With the advancement of technology, people have begun to reduce the use of paper documents to record student information. Instead, we can manage and record student information through online websites. This method not only reduces ecological pollution, but also makes it easier to save student information, and the most important thing is to be able to effectively manage student information. Enable users to be more efficient when processing student records through the use of automated systems. Therefore, the student academic record system plays a key role in the overall functioning of the education system.

1.2 Scope

This document provides an overview of the technical aspects of the SS-Grade project and the techniques used to develop and implement the application as well as the overall functionality, constraints, and specifications. By describing use cases through system architecture, diagrams, and tables, readers can better understand how applications are developed and implemented. It is a web-based application compatible with all modern browsers.

1.3 Project Summary

SS-Grade is a system that facilitates users to manage personal information and course information of student information. The software allows users to easily and intuitively find, add, delete or update students' academic information and visually displays students' personal information, grades and corresponding course codes to users. SS-Grade is a complete web-based student information management system for any user who wishes to manage their student's academic information through an interactive and intuitive user interface (UI) that is created for user comfort while providing Problems provide simple and clear solutions. SS-Grade provides a home page that includes a register page, login page, add page, search page, update page, delete page, and final grade page

1.4 Document Objective

This document will cover all the requirements required by the software. It will also describe all its features and how it works. The target user would be any user who wants to manage student information and course information. They will use this document to remind them of the project's requirements. This document will also be used as the basis for this project.

1.5 References

Description of a student record system : Building An Automated Record System. (n.d.). https://nces.ed.gov/pubs2000/building/desc_system.asp

2. Design Considerations

2.1 Constraints

SS-Grade applications use a web server to program some of these elements. HTML and CSS will display the pages of our website. SQL will be used to retrieve, add, insert, select, and update the database. PHP will serve as the backend for submitting SQL commands. MySQL acts as a database and is subject to the features, functionality, and limitations of the database. This can impact application performance if a large number of users and data are stored.

2.2 Environment

SS-Grade developed using the following technology stack:

- JavaScript is the development language used for both the frontend and backend, connecting the client and database.
- MySQL is used for database storage and manipulation.
- Apache - Open source web server.
- Visual Studio Code - Source code editor.
- CSS(HTML) - style sheet language used to display the website.
- PHP - Server-side programming language

2.3 Methodology

Our APP utilizes the agile development methodology in which we split into sub-teams and divide ourselves to work on the frontend, backend and database design. Each team is responsible for creating weekly sprints and development goals to help brainstorm new features based on backlog and user stories. Once a week, we form a meeting and review what has been completed and what has to be done for the upcoming weeks. The team communicates via WeChat, and members are encouraged to express the issues they are facing. As a result, we can recognize and solve these problems.

2.4 Main Feature

For SS-Grade projects, the main features can be broadly categorized to cover all the key features that such an application should provide. These features are designed to create a comprehensive, user-friendly and secure environment for efficient access and management of student academic data.

2.4.1 Student file management

- Updates: Allows student records to be updated to reflect changes in personal information, course enrollment, and academic performance.
- Delete: Provides the ability to securely delete student records from the system, with appropriate checks and balances to prevent accidental data loss.
- Add: Allow users to add student information and course information from the system.
- Select: Users can query student performance information by entering student ID and course code.

2.4.2 Final grade calculation

- Grade Assignment: Allows users to update students' grades on tests and exams.

- Final grade generation: summarizes students' grades.

2.4.3 Security and privacy

- Data encryption: Ensure that sensitive data stored and transmitted by the system is encrypted to protect user privacy.

3. Architectural Design

3.1 System Design and Decomposition

The structure of our application consists of three main parts: Frontend and Backend and Database. The client is a web browser that interacts with the server through HTTP requests. When the server receives the request, it will use Apache to execute the corresponding PHP file and can interact with the MySQL database to store data and perform operations. The results of the operation are sent back to the PHP script, after the script's response to the operation, it will sent back to the client's web browser. Finally, the client's web browser for user viewing and interaction.

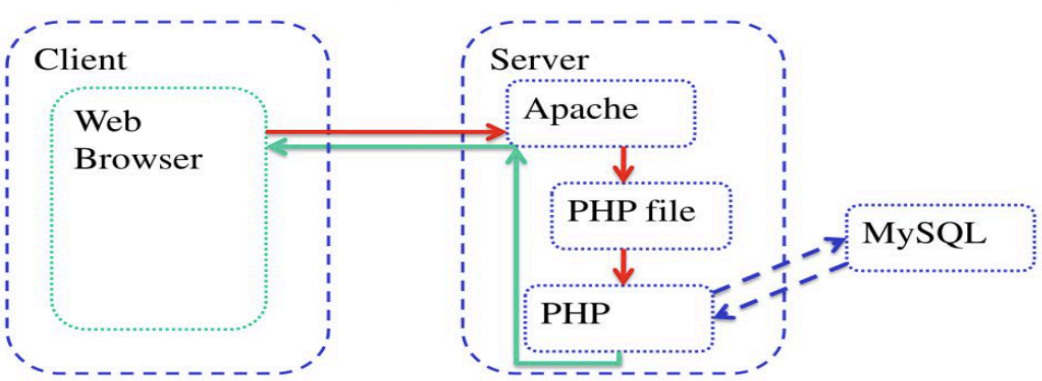


Image_1: System Design

3.2 System Architecture Design Overview

3.2.1. User interface layer (front-end)

- Web pages/forms: HTML forms with CSS styling provide a user interface for entering, displaying, and managing data. JavaScript can be used for form validation and enhanced user interaction.
- Add information page: enter new student information and course information.
- View/query information page: Convenient to query and display student information.
- Update information page: Update course information (grades) for existing students.
- Delete information page: delete student records (grades, student personal information).
- Final Grade Page: Calculate the student's final grade.

3.2.2. Application layer (backend)

- Server-side scripting (PHP): handles user requirements, form submission, and database interaction.
- Form handlers (add_data.php, data.php, update_data.php, deleteinfo.php): specific scripts that handle user-submitted forms to add, update, or delete records.
- Session Management: Manage user sessions for authentication and maintain user state across requests.

3.2.3. Data access layer

- Database connection (connect.php): manages the connection with the database system. (Included in other scripts to facilitate database operations)

- SQL operations: Use the PDO method, Prepare a statement for SQL to perform SQL operations: add, update, delete or query.

3.2.4. Database layer

- Database Management System (DBMS): Stores and manages all student personal information, course information, and user registration data.

- Table: Name_Table: Student name and ID. Course_Table: course code, student ID, student score. Register_Table: username and password registered by the user.

- Referential integrity: Ensures relationships between administrative systems, such as linking student IDs to corresponding course registrations and grades.

3.2.5. Security measures

- Data Validation: All input received through forms is properly validated on both the client side (using JavaScript) and the server side (using PHP) to prevent SQL injection and security threats.

- Password management: Securely store user passwords using hashes.

4. Detail Design

4.1 Data Description

The data stored consists of 3 primary tables (known as “collections” in MySQL):

Name_Table: User collection stores user information such as Student_ID(primary key) and Student_Name

Course_Table: Course collection stores course information, such as Student_ID(Primary key), Course_Code, Test_1, Test_2, Test_3, and Final_Exam. Each Test weighs 20% and the final exam weighs 40%.

Users_Table: Id, username, password (hash form).

Table 1: Name_Table

Field	Type	Description
Student_Id	Varchar(9) PK, NN, UQ	Unique identifier for each student, consisting of a 9-digit number. Is the primary key.
Student_Name	Varchar(50) NN, UQ	Student name for each student

Table 2: Course_Table

Field	Type	Description
Course_Code	Varchar(10) NN	Identifier for each course record.
Student_id	Varchar(9) NN	The student's unique identifier, linking to the Student_ID in Name_Table. Foreign key.
Test_1	DECIMAL(4,1)	Grade for Test 1. default 0, with one digital after the dot.
Test_2	DECIMAL(4,1)	Grade for Test 1. default 0, with one digital after the

		dot.
Test_3	DECIMAL(4,1)	Grade for Test 1. default 0, with one digital after the dot.
Final_Exam	DECIMAL(4,1)	Grade for Test 1. default 0, with one digital after the dot.

Table 3: User_Table

Field	Type	Description
ID	INT PK, NN, AI	Auto_increment ID for each user
username	Varchar(255) NN, UQ	A unique identifier for each user. This serves as the primary key.
password	Varchar(255) NN	The user password for each user will be hash form in the database for security

4.2 Data flow and operations

4.2.1. Add student information

- Enter data through the add.php form, including student ID, name, course code and grades.

- The add_data.php script validates the input and performs two main SQL INSERT operations: one adds the student's personal information to the Name_Table, and the other adds the student's information and grades to the Course_Table.

4.2.2. Update student information

- Facilitate updates via an update form, similar to add.php.

- The update_data.php script processes the form, validates the input, and executes the SQL UPDATE statement to update the records in Course_Table based on the provided student ID and course code and the grade information you want to change.

4.2.3. Delete student information

- Delete information records for user-specified student IDs.

- The deleteinfo.php script handles the request, first deleting the relevant course and grade records from the Course_Table, and then deleting the student's information from the Name_Table.

4.2.4. Select and display information

- Check student information based on student ID and course code.

- The data.php script handles requests, queries and displays information through SELECT, and uses INNER JOIN to connect Name_Table and Course_Table.

5. Component Design

5.1 Class Diagram

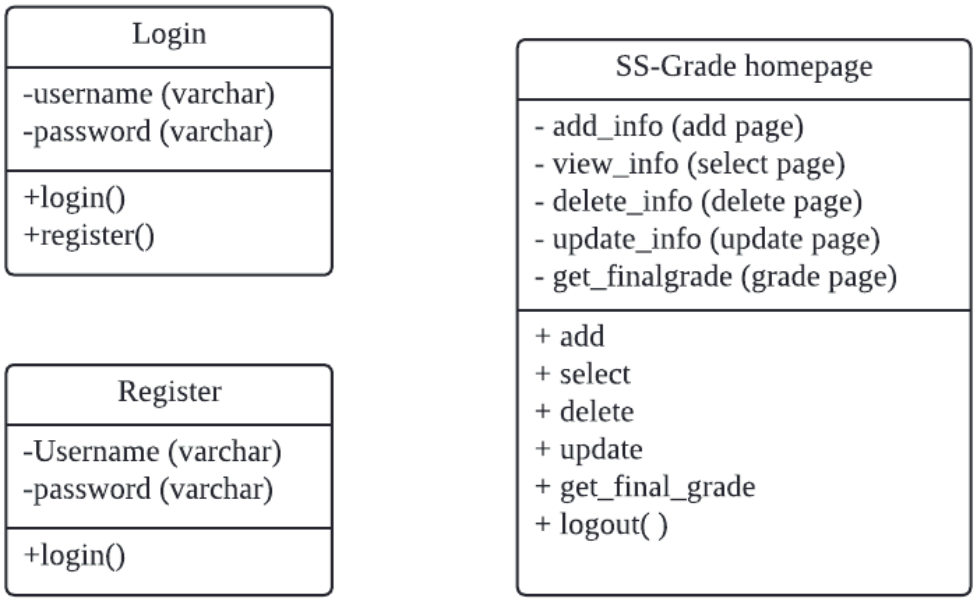


Image_2.1: Class Diagram (classes, attributes, methods)

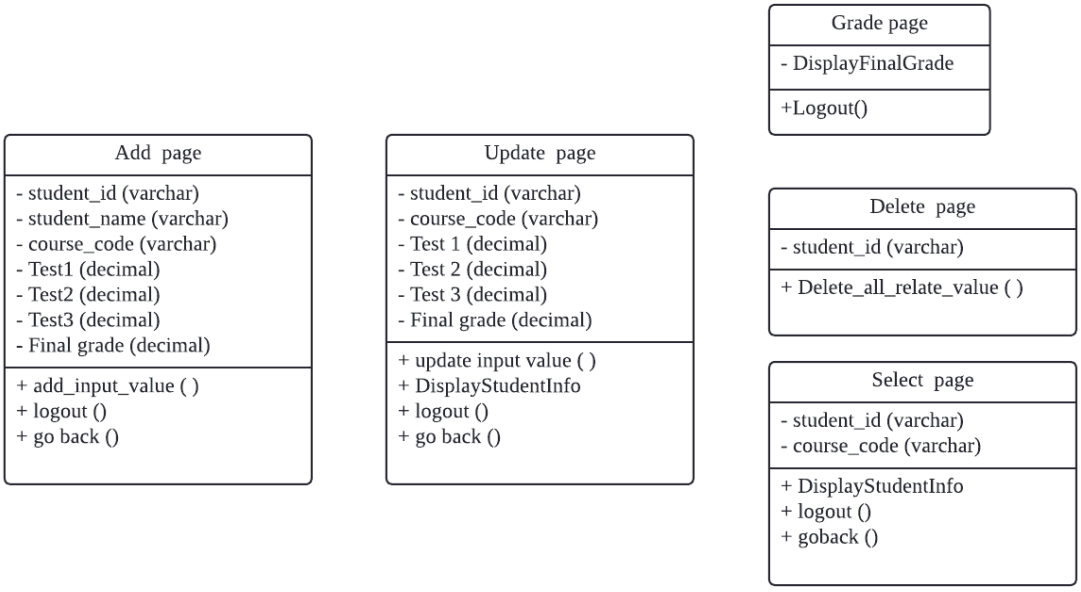


Image_2.2: Class Diagram (classes, attributes, methods)

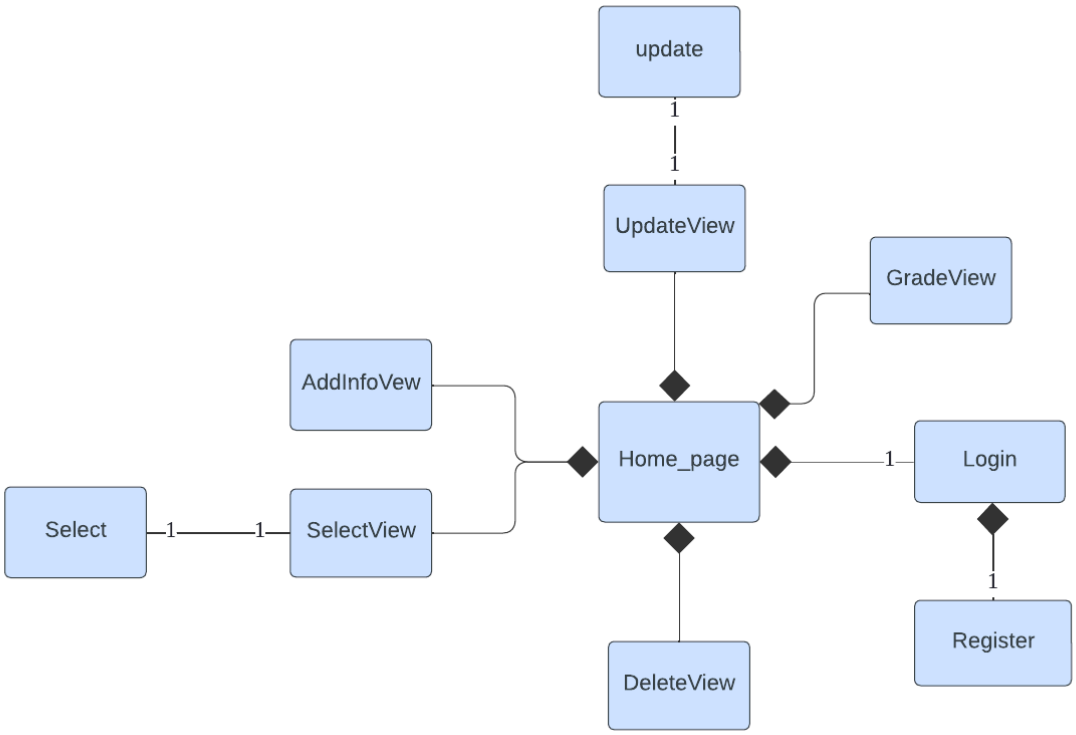


Image_2.3: Class Diagram(associations, dependencies, inheritance)

Explanation:

In the 2.1 class diagram, there are two primary classes: Login and SS-Grade homepage. The Login class contains attributes for 'username' and 'password', and methods to 'login()' and 'register()'. This suggests that the system has a user authentication mechanism that requires users to sign in to

access or register for an account on the SS-Grade system. The SS-Grade homepage class, as its name implies, is the main interface users interact with after logging in. It contains attributes for different operations like 'add_info', 'view_info', and others that correspond to different pages within the system. The methods include operations to 'add', 'select', 'delete', 'update', and 'get_final_grade', describing these are the main functionalities offered by the SS-Grade system for managing academic records.

The 2.2 class diagram shows additional classes for each page of functionality, including Add page, Update page, Delete page, Select page, and Grade page. Each of these classes has specific attributes and methods corresponding to the fields and actions a user can perform on that page. For example, the Add page class has attributes for 'student_id', 'student_name', 'course_code', and test grades, with methods to add input values, log out, or go back, it creates a user interface where new student records can be entered.

The 2.3 diagram represents the navigation flow between the different views within the SS-Grade system. It shows how different actions taken on the homepage can lead to different views, such as 'AddInfoView', 'UpdateView', 'DeleteView', and others. This navigational class diagram may serve as a high-level overview of how users will interact with the system, showing a user journey from logging in to performing various tasks related to student information management.

Together, we successfully created and implemented our requirements for the project by visual blueprint of the SS-Grade system, detailing its functionality, user interaction, and data management strategies to support the management and recording of student academic information in an online environment

5.2 Sequence Diagrams

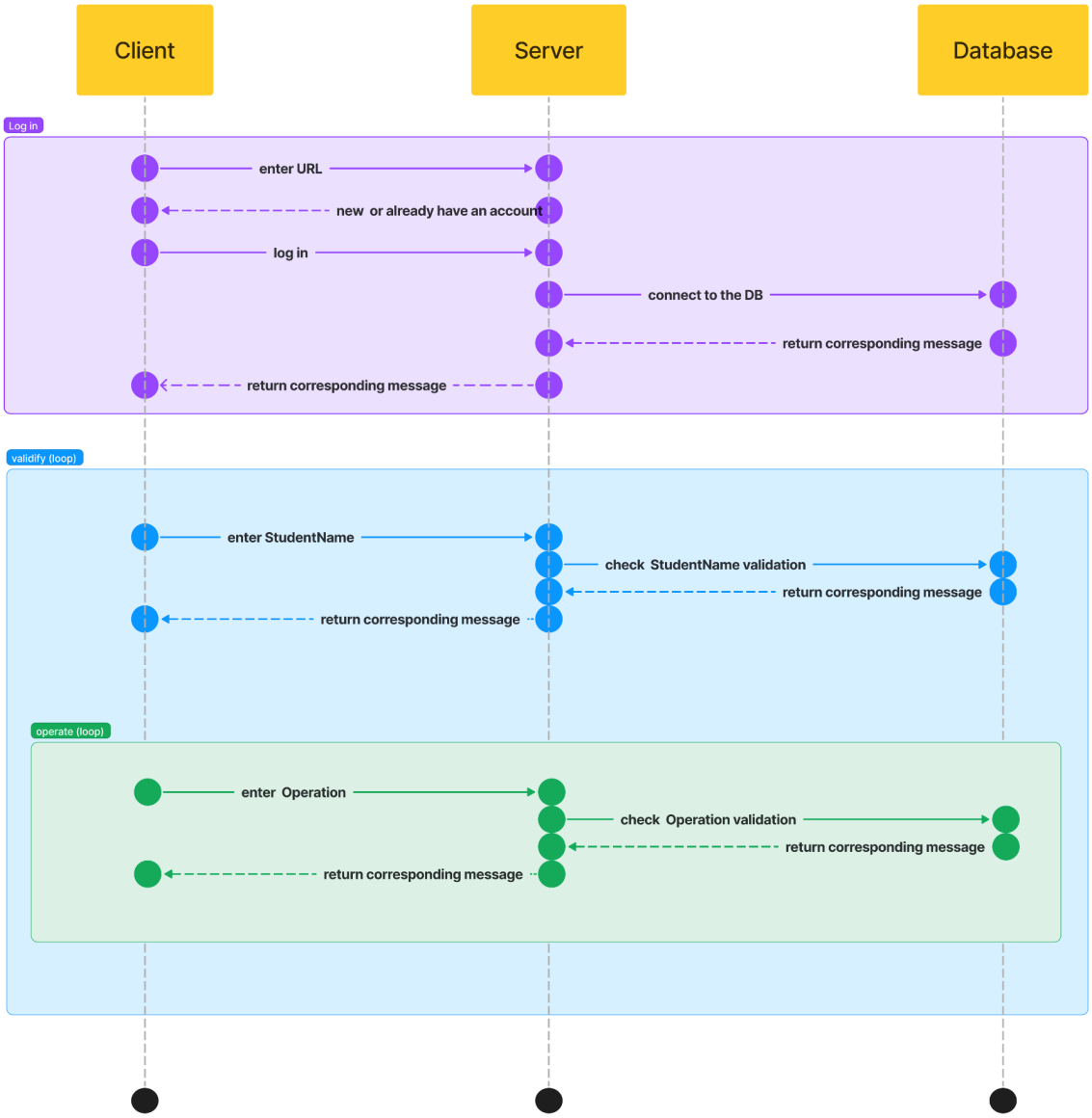


Image_3: Sequence Diagram

To effect the expected interaction between the actor (user) and the system a variety of sequence diagrams are created to illustrate the particular sequence of events that occurs during a set use case.

5.3 Activity Diagrams

In addition to the sequence diagrams to illustrate use cases for the application, activity diagrams are used to model dynamic user behaviour within the application. The main user interactions within the application are: checking students' info, adding info, deleting info, and updating info.

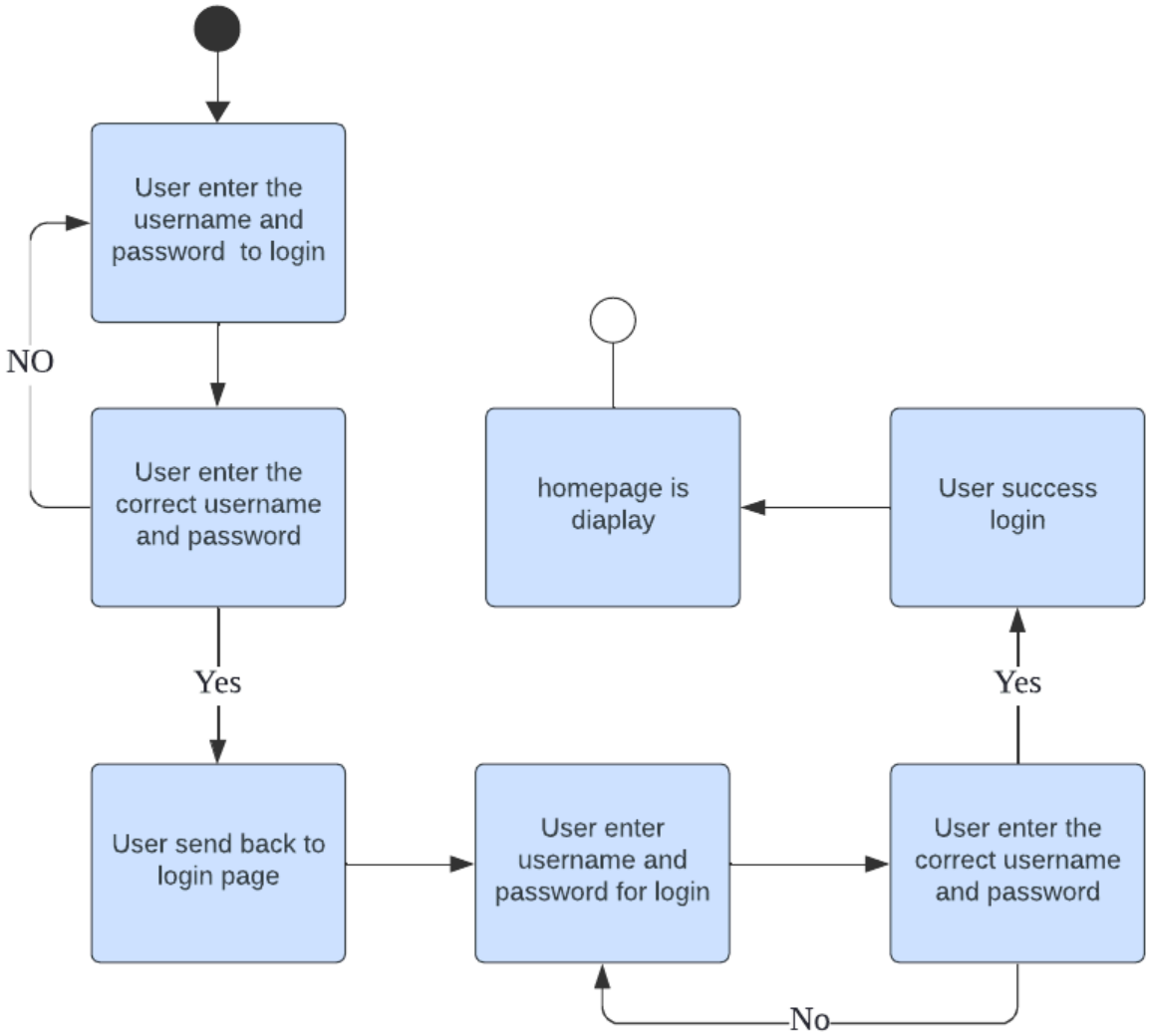


Image_4.1: Activity Diagram (login)

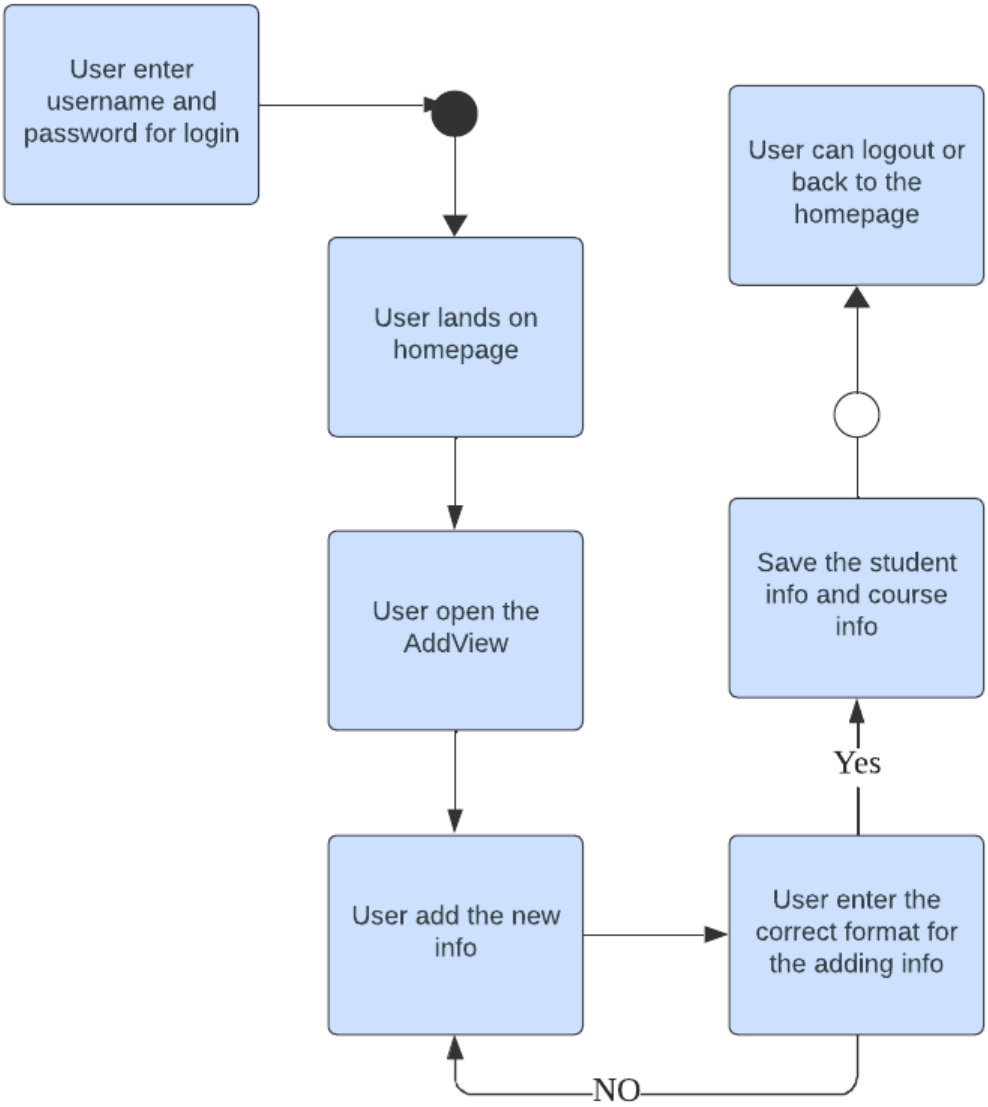


Image_4.2: Activity Diagram (add)

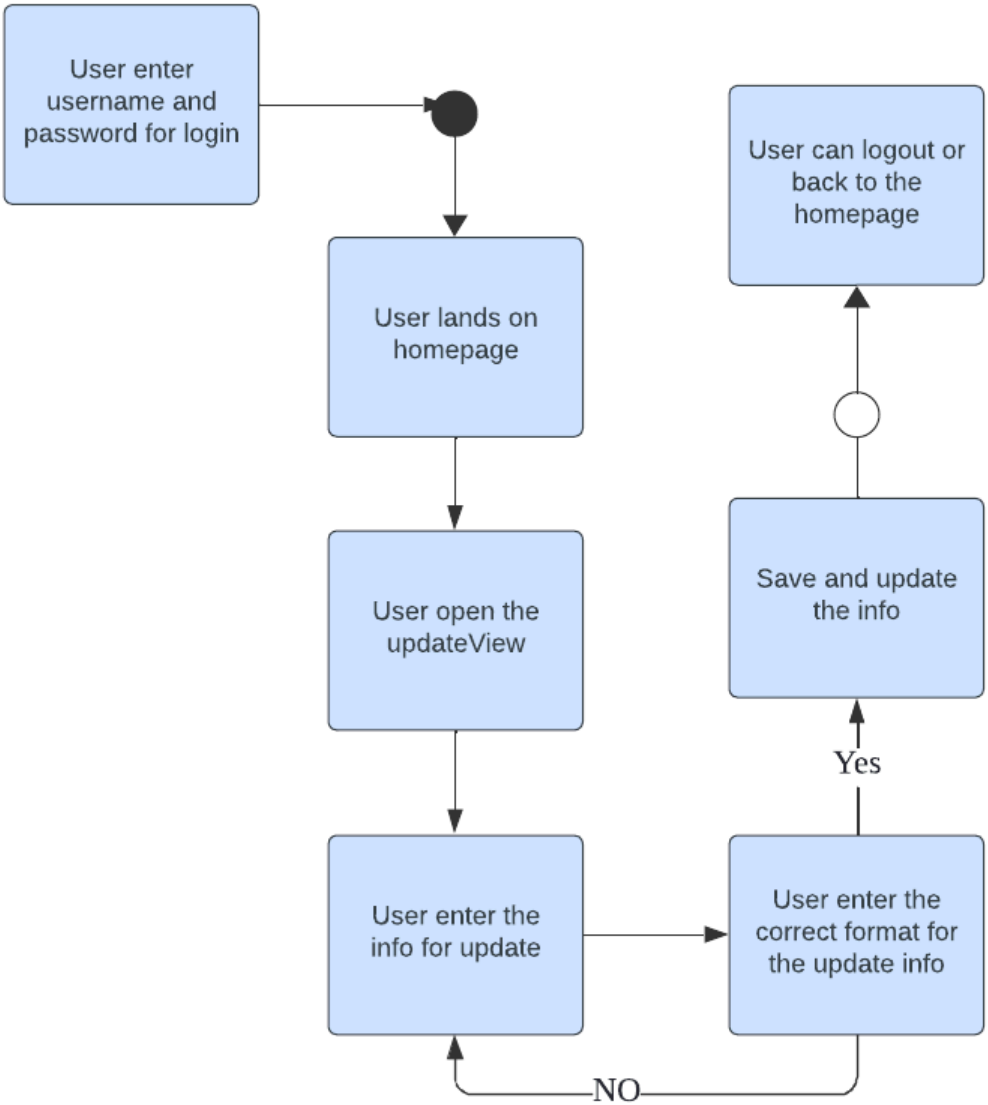


Image_4.3: Activity Diagram (update)

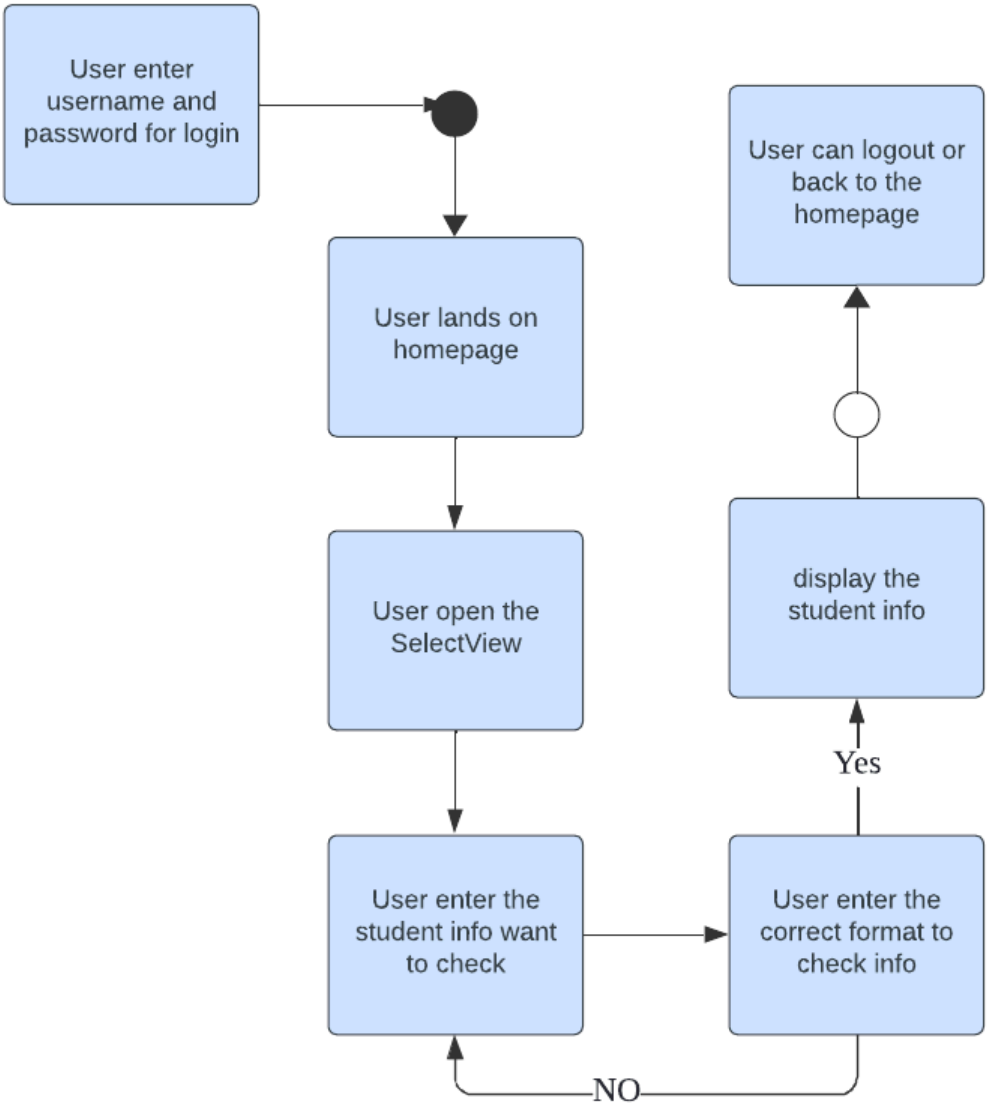


Image_4.4: Activity Diagram (select)

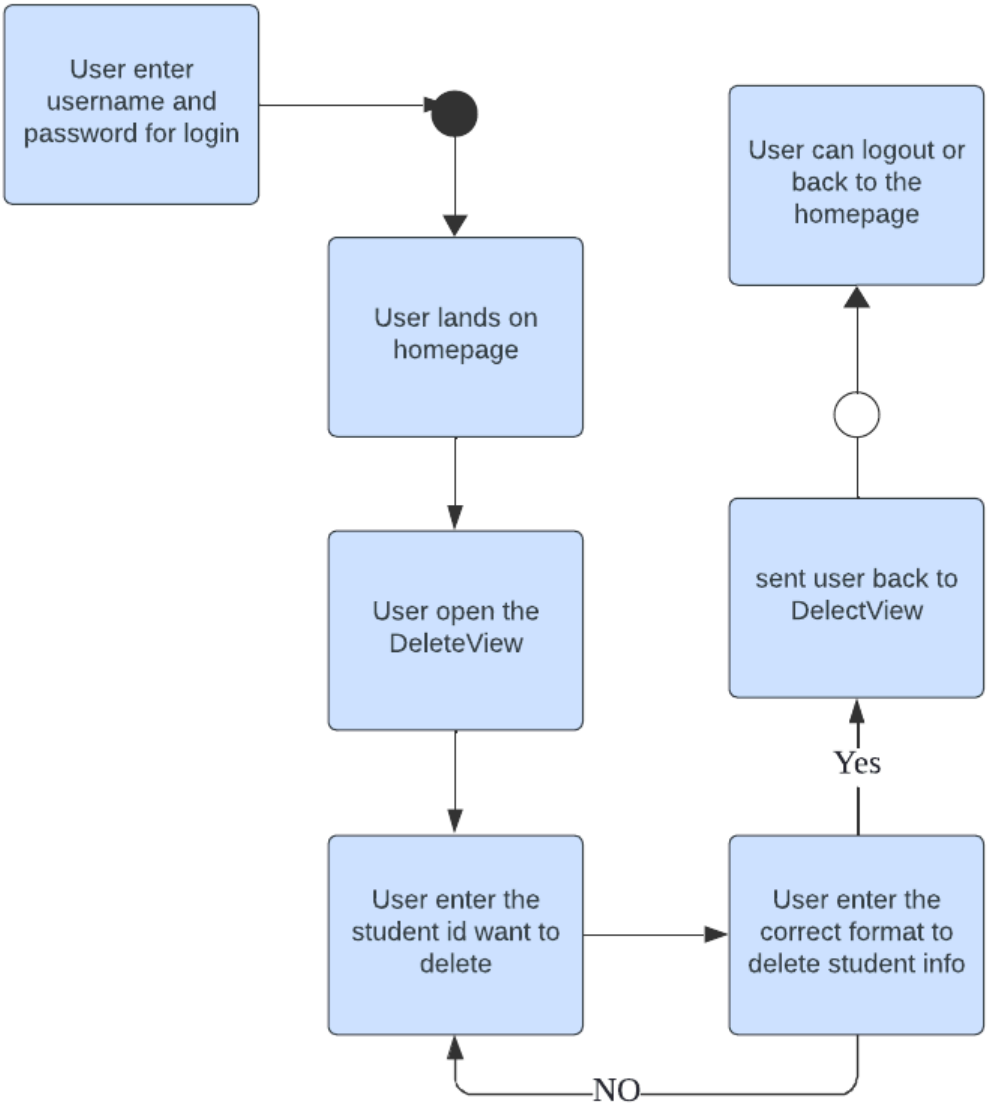


Image_4.5: Activity Diagram (delete)

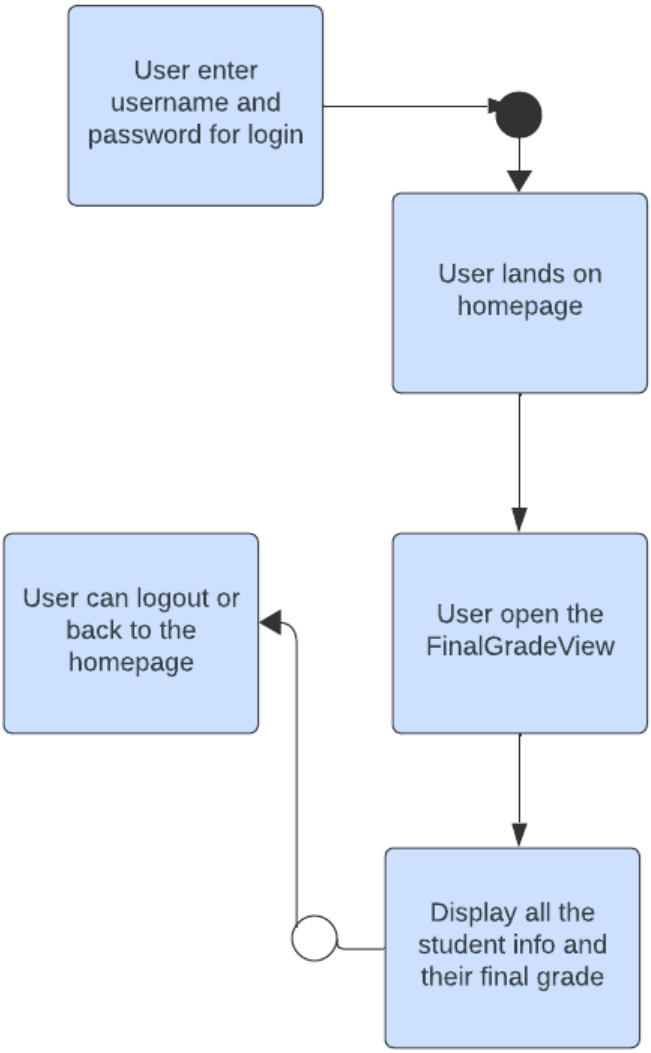


Image_4.6: Activity Diagram (grade)

6. User Interface Design

6.1 User Interface Design Screenshots

The following are screenshots of the UI of the application and how it would look when a user is interacting with the application.

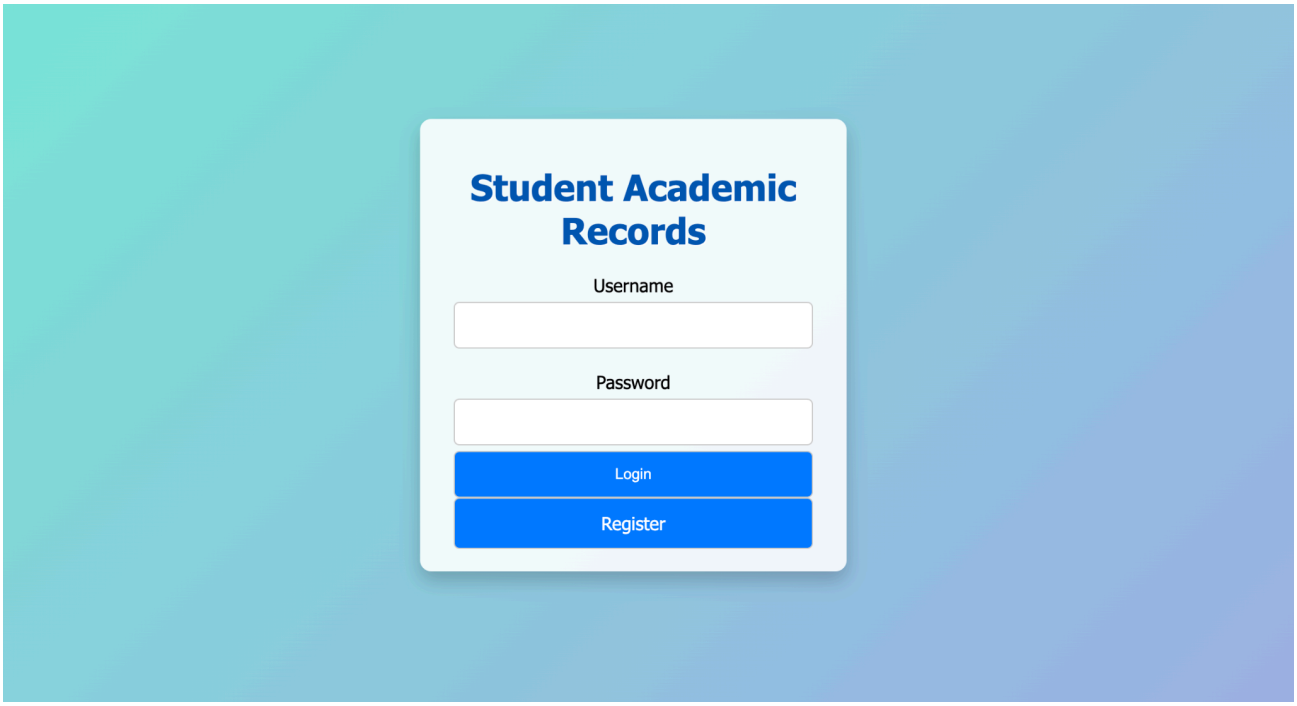


Image_5.1: UI (login)

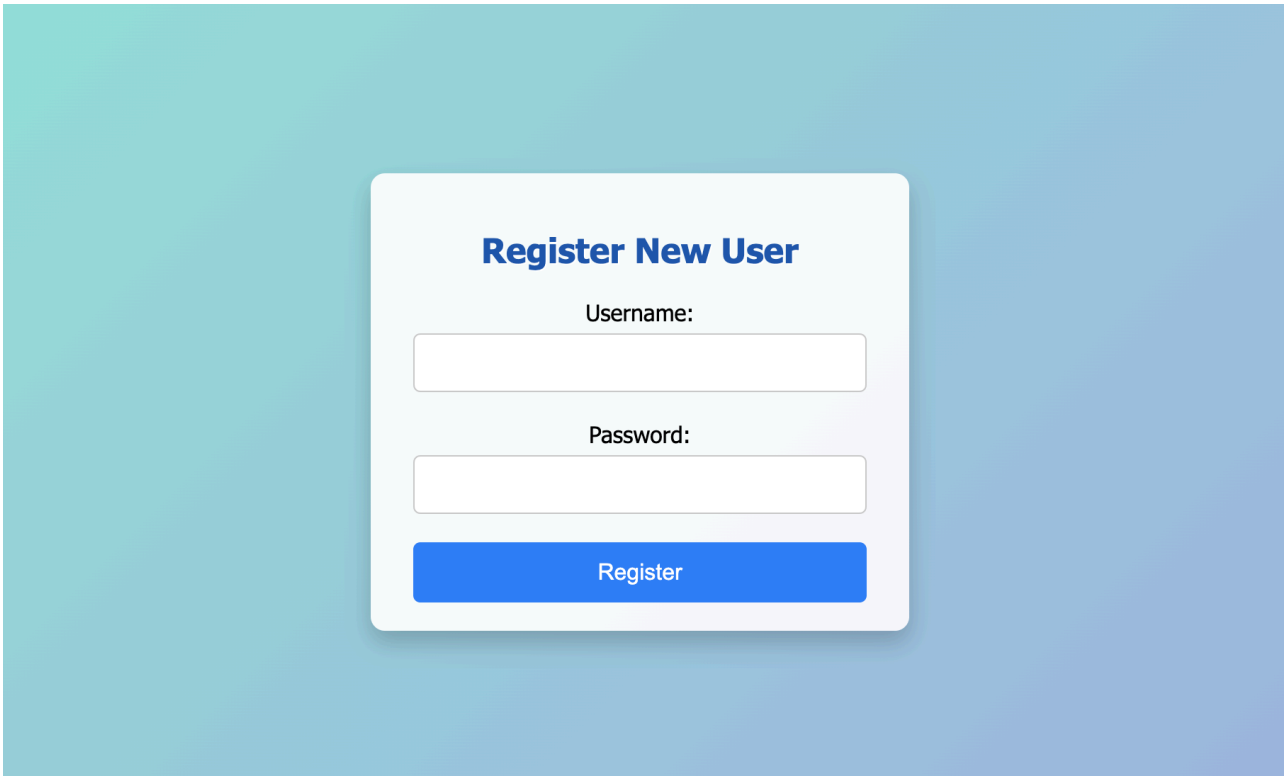


Image_5.2: UI (register)

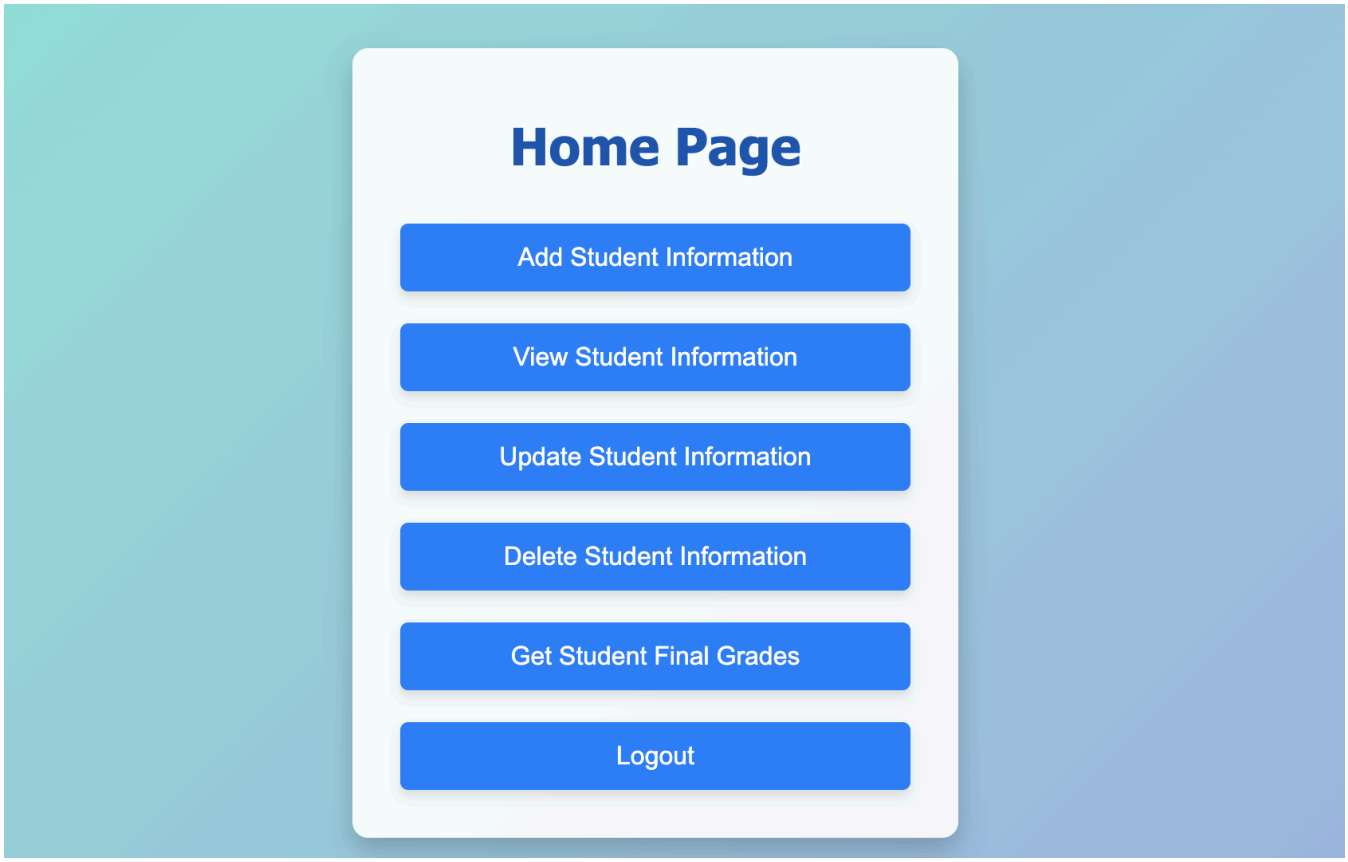


Image_5.3: UI (home)

Final Grades

Go Back Logout

Student ID	Student Name	Course Code	Final Grade
187509717	Ameena Khan	CP202	78.0
187509717	Ameena Khan	ST490	79.4
251173274	Xiao Qiang	PS275	66.6
251173274	Xiao Qiang	EC140	63.8
256047895	Lori Donovan	MA222	79.0
256047895	Lori Donovan	EC140	60.8
280587734	Kendra Paul	PS272	70.4
280587734	Kendra Paul	CH202	75.8
293688639	Dominique Lovel	CH120	81.8
293688639	Dominique Lovel	BU121	82.4

Image_5.4: UI (DB)

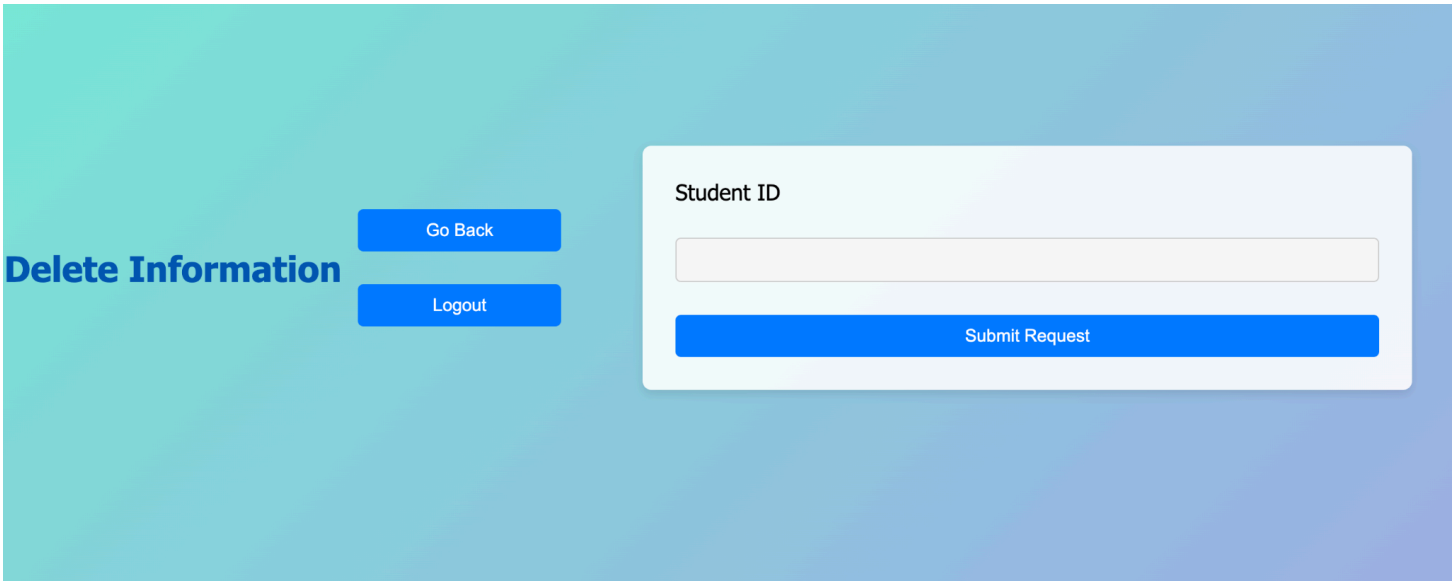


Image_5.5: UI (delete)

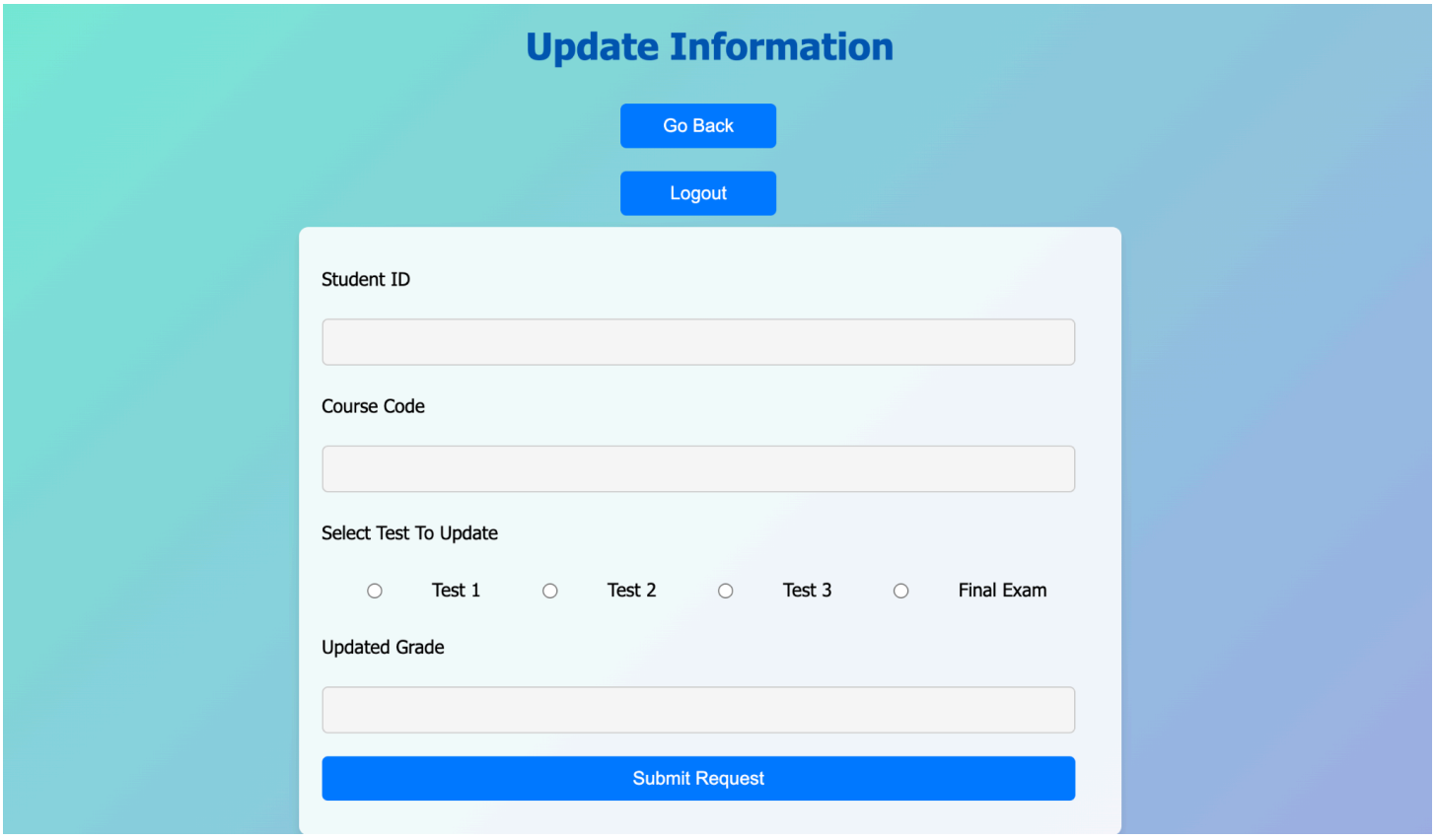


Image_5.6: UI (update)

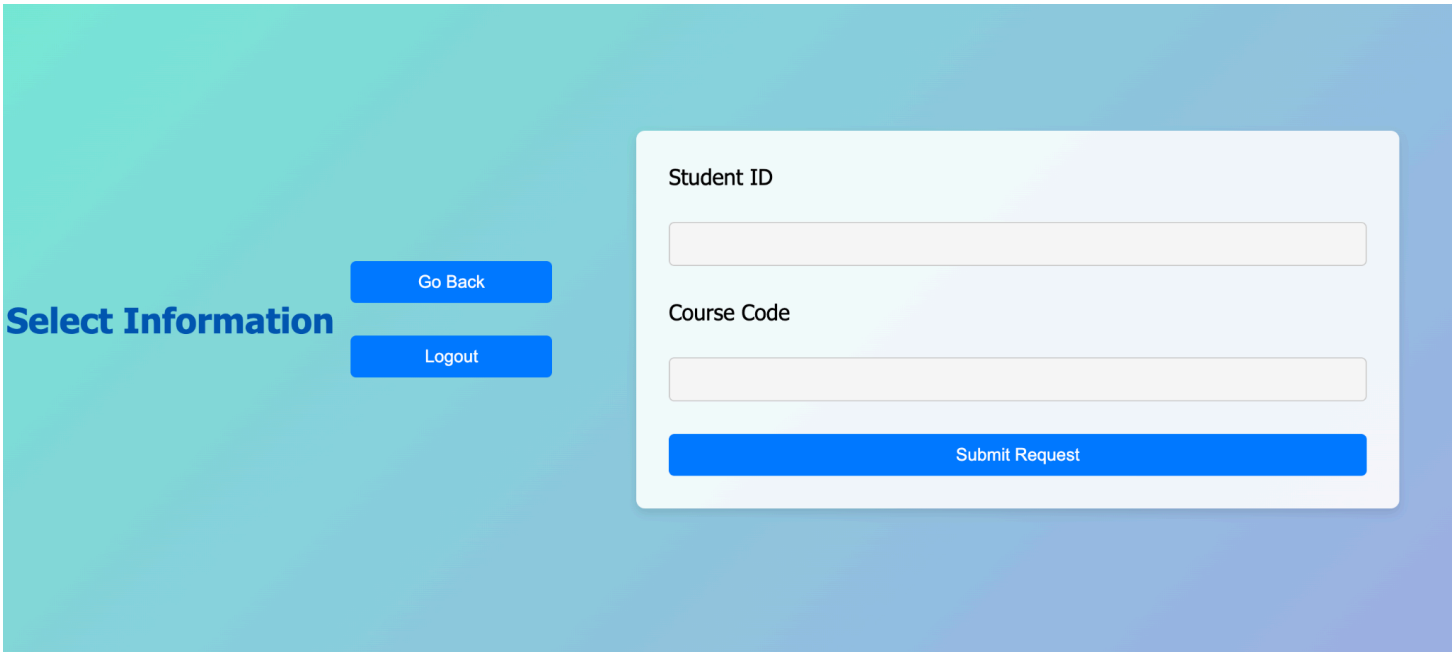


Image_5.5: UI (select)

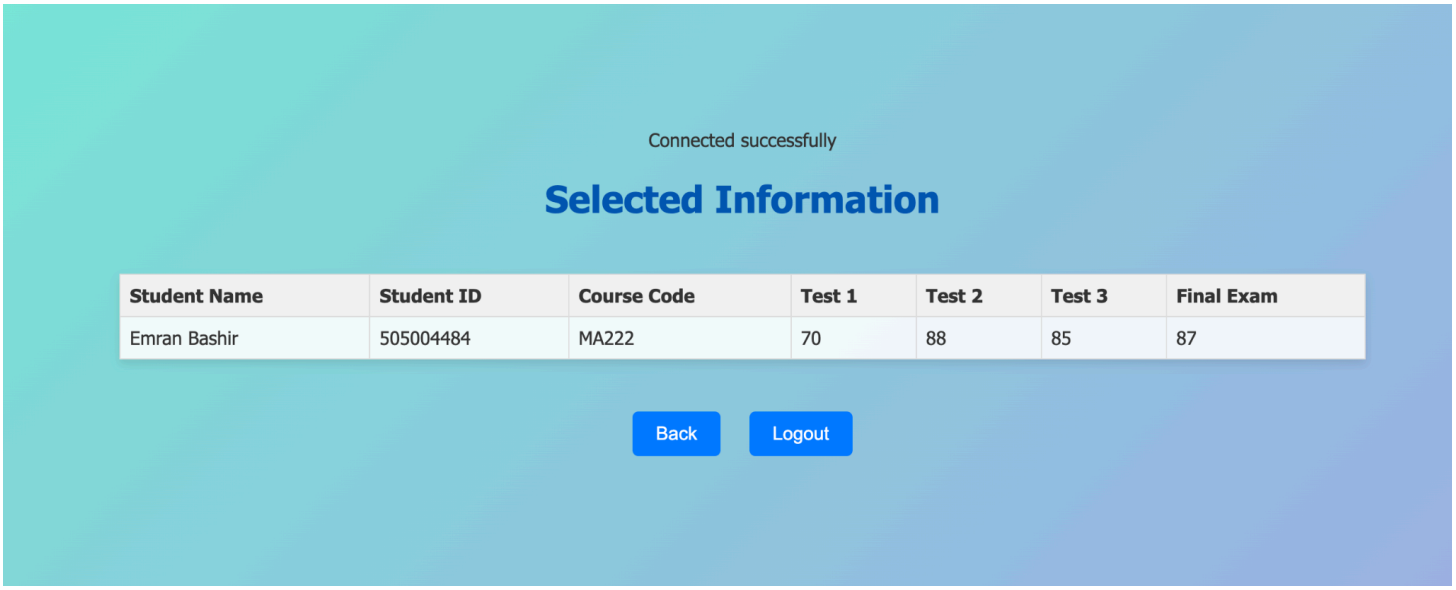


Image_5.6: UI (selected)

7. Appendix

Final Grades

Go Back | Logout

Student ID	Student Name	Course Code	Final Grade
187509717	Ameena Khan	CP202	78.0
187509717	Ameena Khan	ST490	79.4
251173274	Xiao Qiang	PS275	66.6
251173274	Xiao Qiang	EC140	63.8
256047895	Lori Donovan	MA222	79.0
256047895	Lori Donovan	EC140	60.8
280587734	Kendra Paul	PS272	70.4
280587734	Kendra Paul	CH202	75.8
293688639	Dominique Lovel	CH120	81.8
293688639	Dominique Lovel	BU121	82.4

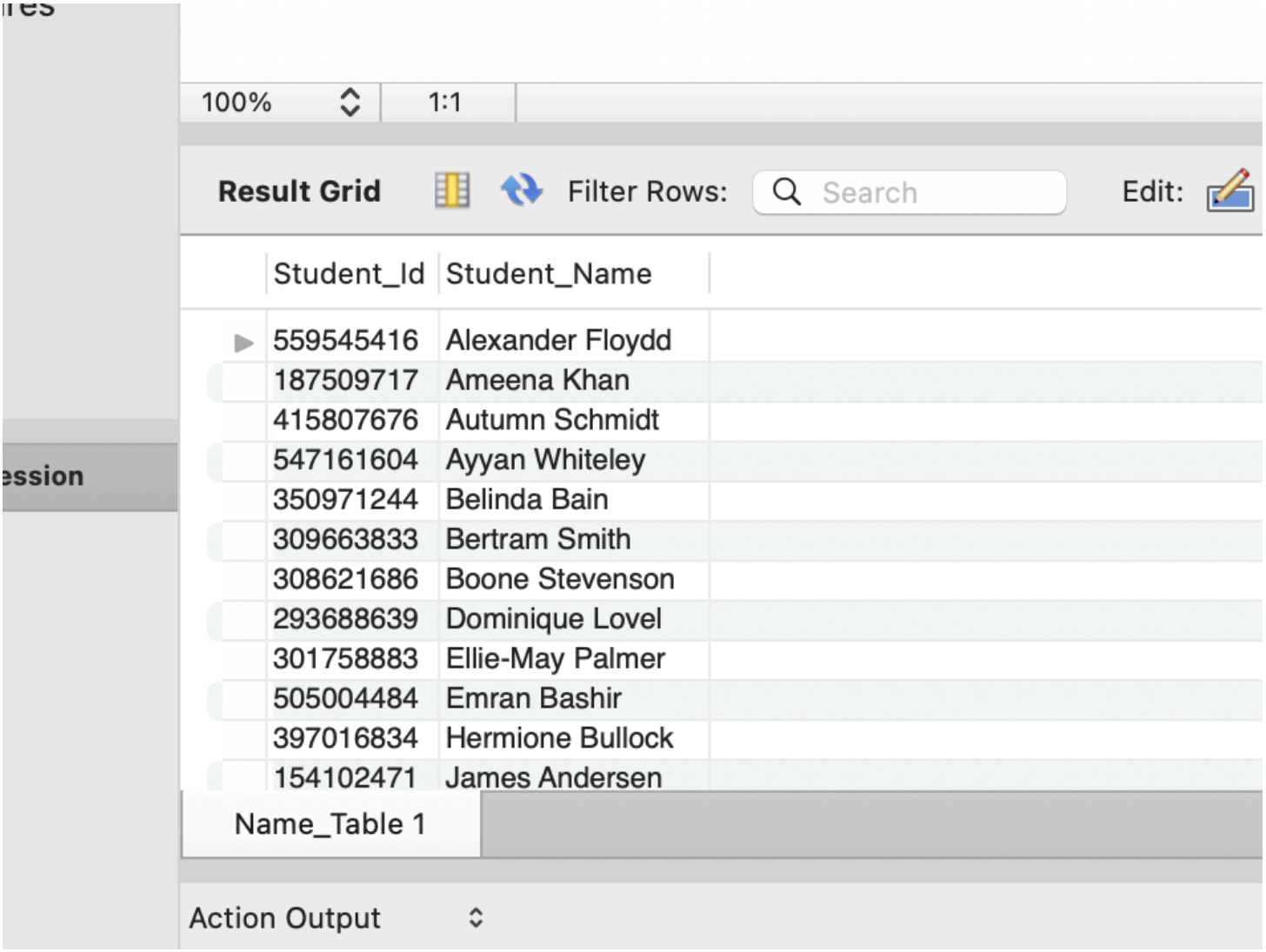
Final Grade output

100% 1:1

Result Grid Filter Rows: Search Export:

Student_Id	Cours...	Test1	Test2	Test3	Final_Exam
280587734	PS272	74.0	98.0	76.0	52.0
280587734	CH202	66.0	82.0	81.0	75.0
256047895	MA222	69.0	80.0	72.0	87.0
154102471	CP465	63.0	82.0	58.0	68.0
187509717	CP202	58.0	98.0	56.0	89.0
503239671	ST262	66.0	84.0	95.0	88.0
448227065	CP465	59.0	69.0	56.0	96.0
429464715	CH120	54.0	93.0	71.0	80.0
627137015	EC140	85.0	56.0	72.0	77.0
415807676	EC140	70.0	89.0	90.0	63.0
293688639	CH120	85.0	80.0	78.0	83.0
256047895	EC140	66.0	53.0	85.0	50.0

Course_Table 1



Name Table and Course Table output