**Lorem Ipsum – Assignment 2:**

**Student Conduct Tracker**

Skyler Almarales

Student ID: 816034038

GitHub Username: Zurun-Rikuthin

**–**

Kadie Bisnath

Student ID:816036090

GitHub Username: KeraGold

**–**

Jaheim Caesar

Student ID:816035438

GitHub Username: TheLegend1011101

**–**

Ronaldo Walker:

Student ID: 816036438

GitHub Username: ronaldowalker

St. Augustine Campus, University of the West Indies

COMP 3613: Software Engineering (Semester I – 2024/2025)

Mr. Nicholas Mendez

October 18, 2024

# Table of Contents

[**Table of Contents 2**](#_b4ovbkj5m2b3)

[**Links 4**](#_c4hjr3pbiiad)

[**Feature List 4**](#_jpin5jpz3e04)

[**Member Contributions 5**](#_14sr5yazkvja)

[**Use Case Diagram (Updated) 6**](#_m3nm29r8h8dg)

[**Model Diagram (Updated) 7**](#_atfw8vuz3nya)

[**Wireframes 8**](#_rgj8zl3lqh9g)

[General Schema 8](#_mnltuyw5rizu)

[Individual Frames 9](#_vsofvr38etya)

[Landing and Login Page 9](#_60ylyibey7a7)

[Home Page 10](#_fb1e8yc11q37)

[Add Student 11](#_nqhilv5cqduf)

[Search Student 12](#_hw78vrb0e5ji)

[Search Results 13](#_c62jrcf6ifk3)

[Student Details 14](#_fjll592938bu)

[Review Student 15](#_n2rg06v4wakv)

[View All Reviews 16](#_v1yop0lsdbwh)

[Search Reviews 17](#_ma1dghbn7yd)

[Student Reviews 18](#_mln2sj2dea9x)

[User Profile (Upper) 19](#_bgmw11d2xor9)

[User Profile (Lower) 20](#_mxxd9vajctxp)

[**Test Plans 21**](#_4xamvep7sdwj)

[User Acceptance Tests 21](#_lzrleui6prmx)

[User Account Creation 21](#_j6w8tevc167c)

[User Login 23](#_ci5xp7lx0w5f)

[User Logout 24](#_89nra733zguh)

[Add Student Profile 25](#_okwy9mysfpni)

[Search Student 27](#_1dnsuolbbhse)

[View Student Profile 28](#_ym8ol12mcbnb)

[Review Student 29](#_7i4fe9mxmy18)

[Search Reviews 30](#_l2we6pb75vfv)

[View Review 31](#_c6dwhjbikfjq)

[Unit Tests 32](#_17i7gsw8psew)

[Integration Tests 34](#_hnxy4rqoqlyd)

[User Operations 34](#_m59oq3pk14ny)

[Student Operations 35](#_gdx1knz0xttd)

[Review Operations 35](#_8fglc1dqootx)

[API Tests 37](#_5ch123qs8wqc)

[User Account Creation 37](#_j8cryaixtcsn)

[User Login 38](#_2xvzm82pm6dt)

[User Logout 39](#_2f62t2ljjlj6)

[Add Student Profile 40](#_939cjt9wh52d)

[Search Student 42](#_777xog2gqhsz)

[View Student Profile 43](#_c5dx7cvqdy2d)

[Review Student 43](#_6um8pwjkq2q1)

[Search Reviews 45](#_570uq8pz1qyg)

[View Review 46](#_3v2y9zkj80kq)

# 

# Links

GitHub Repo:

* https://github.com/Lorem-Ipsum-Scholars/COMP-3613-Assignment-2

# Feature List

Most Critical Features:

* Add Student
* View Student
* Review Student
* View Student Reviews

Sub-Features:

* Add Student
  + Edit Student
* View Student
  + Search Student
    - Search by ID
    - Filter search
  + View the profile of a specific student
  + List all students in the database
* Review Student
  + Edit Review
* View Student Reviews
  + List all reviews in the database
    - Sort reviews
  + List all reviews for an individual student
    - (Also falls under *View Student* as a given student’s reviews can be listed directly under their profile.)
  + View a specific student review from the list

(Assume an authentication page as the landing page for logged out users.)

# Member Contributions

Skyler Almarales:

* General Planning
* Test Plans (User Acceptance, Unit, Integration, API)
* <Add more here where needed>

Kadie Bisnath:

* General Planning
* Wireframes
* <Add more here where needed>

Jaheim Caesar:

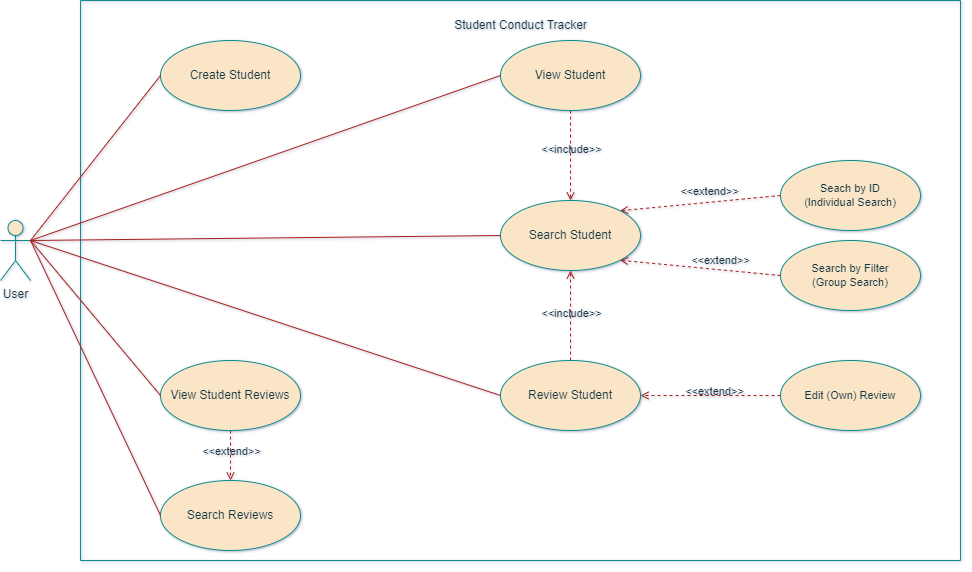
* General Planning
* Views
* Test Plans (Unit)
* Test Implementation (Unit, Integration, API)
* <Add more here where needed>

Ronaldo Walker:

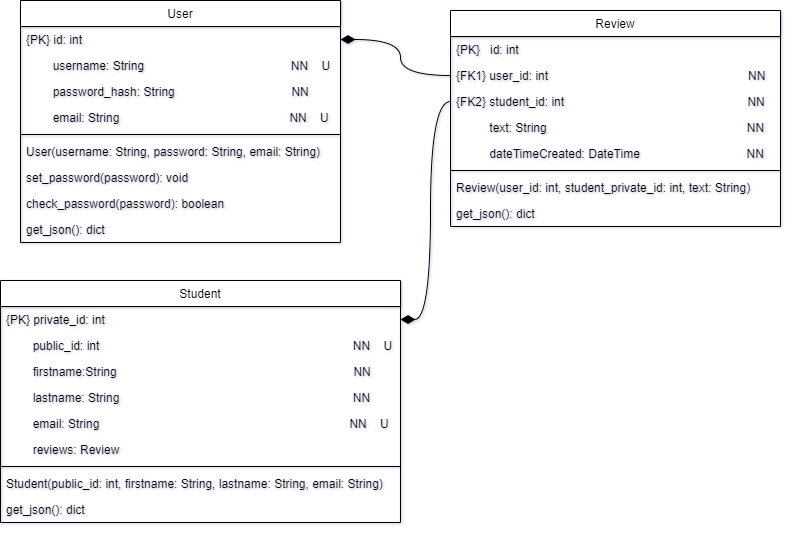
* General Planning
* Views
* Test Implementation
* Test plans (User Acceptance)
* <Add more here where needed>

# 

# Use Case Diagram (Updated)



# Model Diagram (Updated)



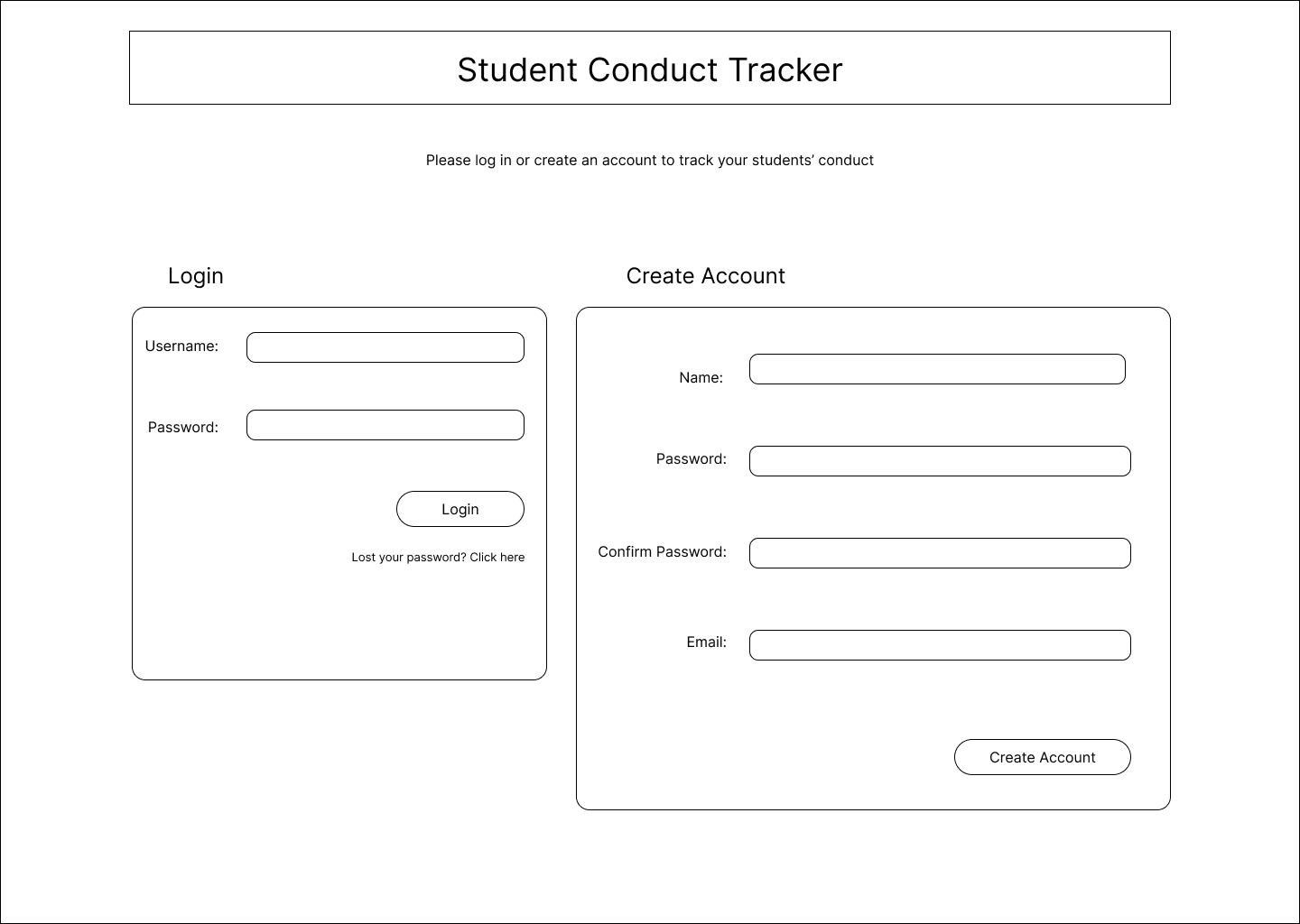
# Wireframes

## General Schema

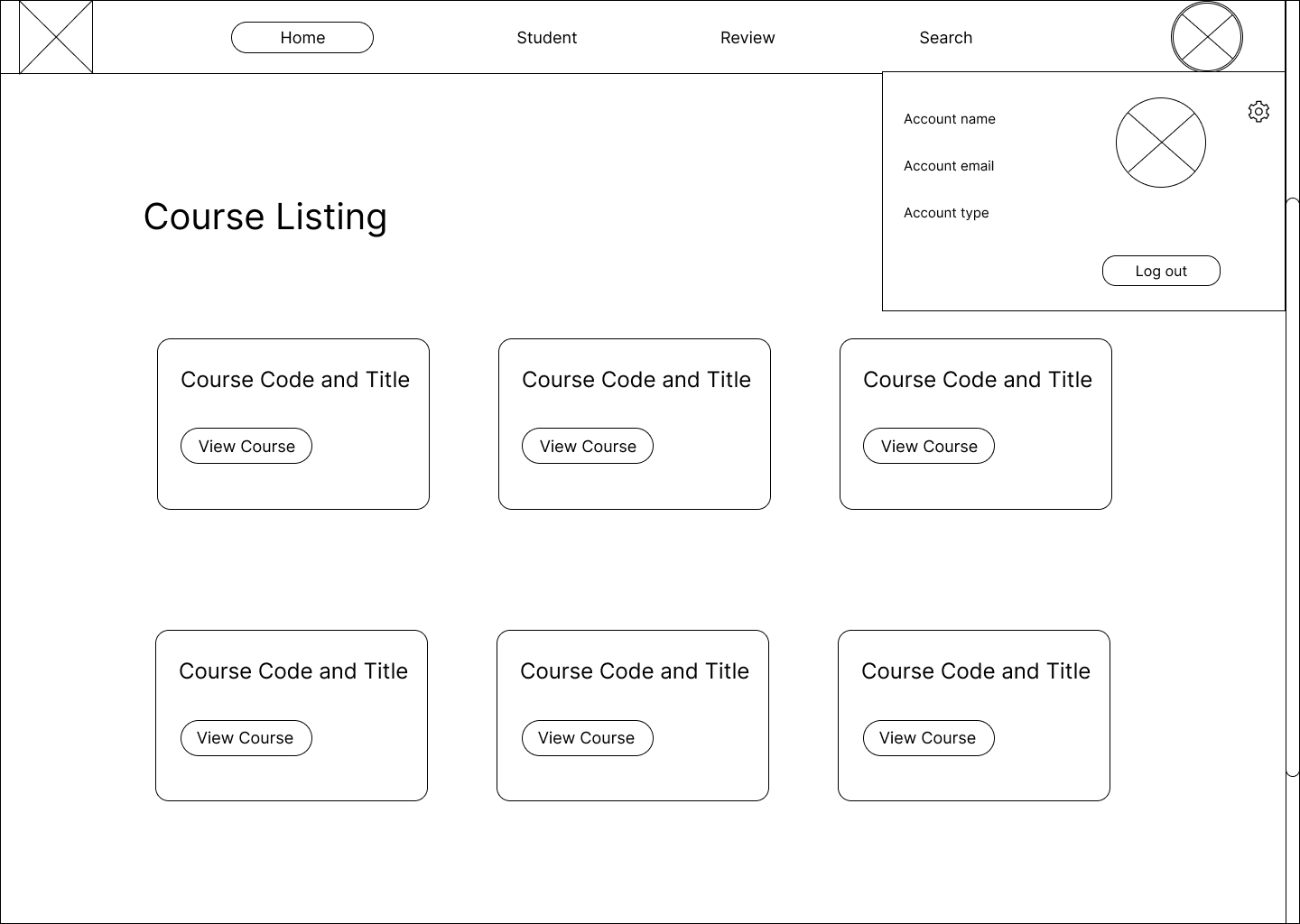
 

## Individual Frames

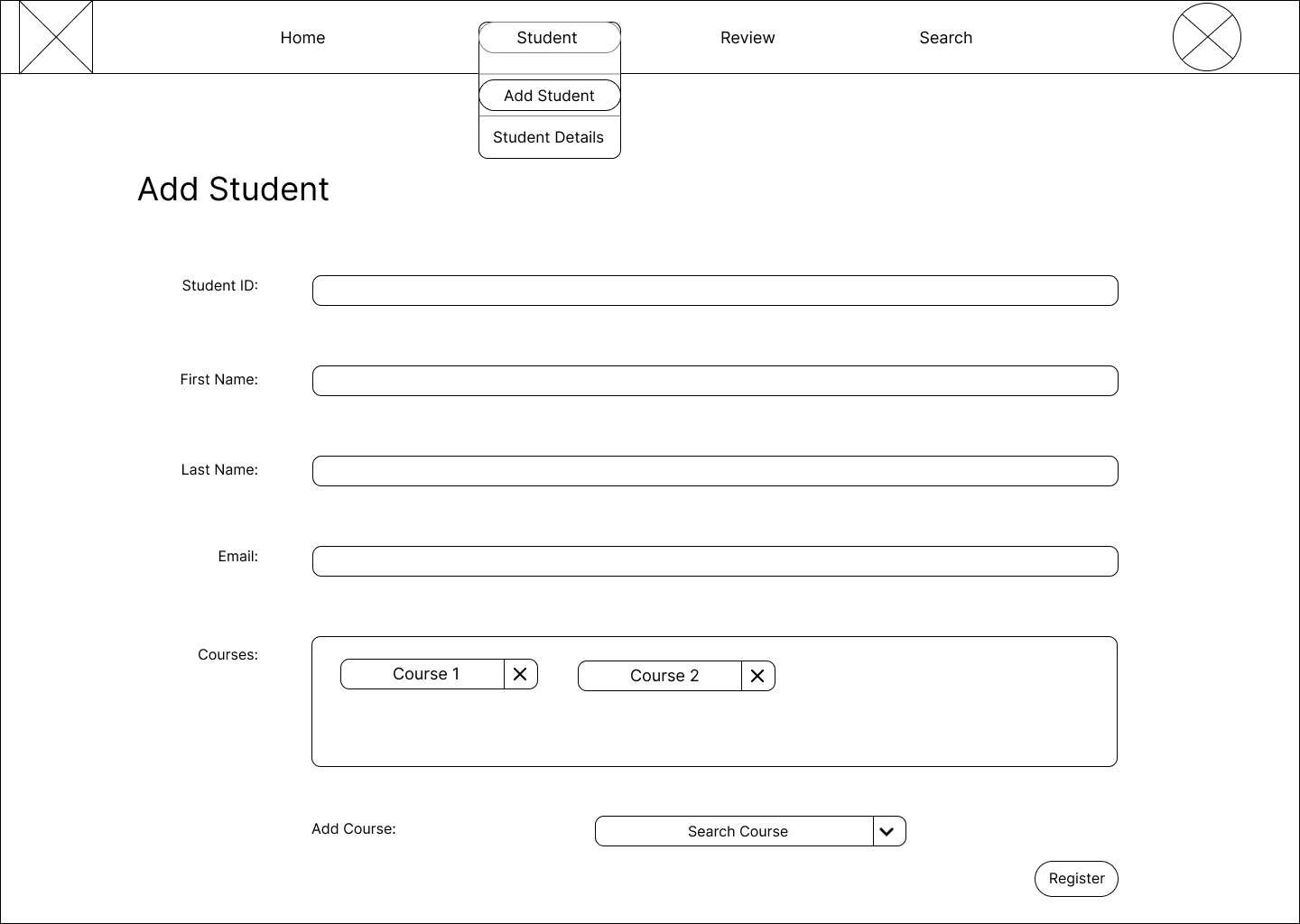
### Landing and Login Page



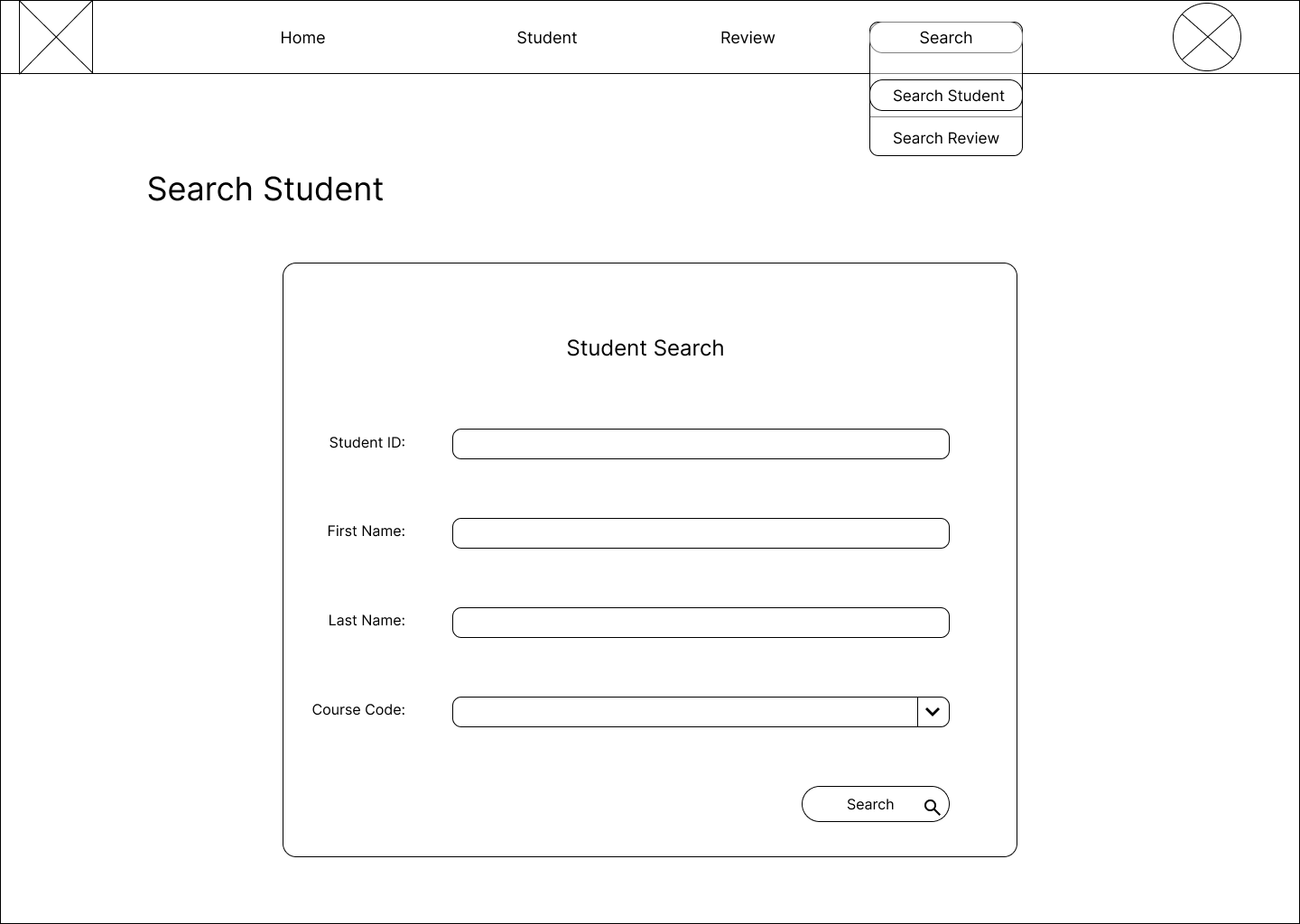
### Home Page



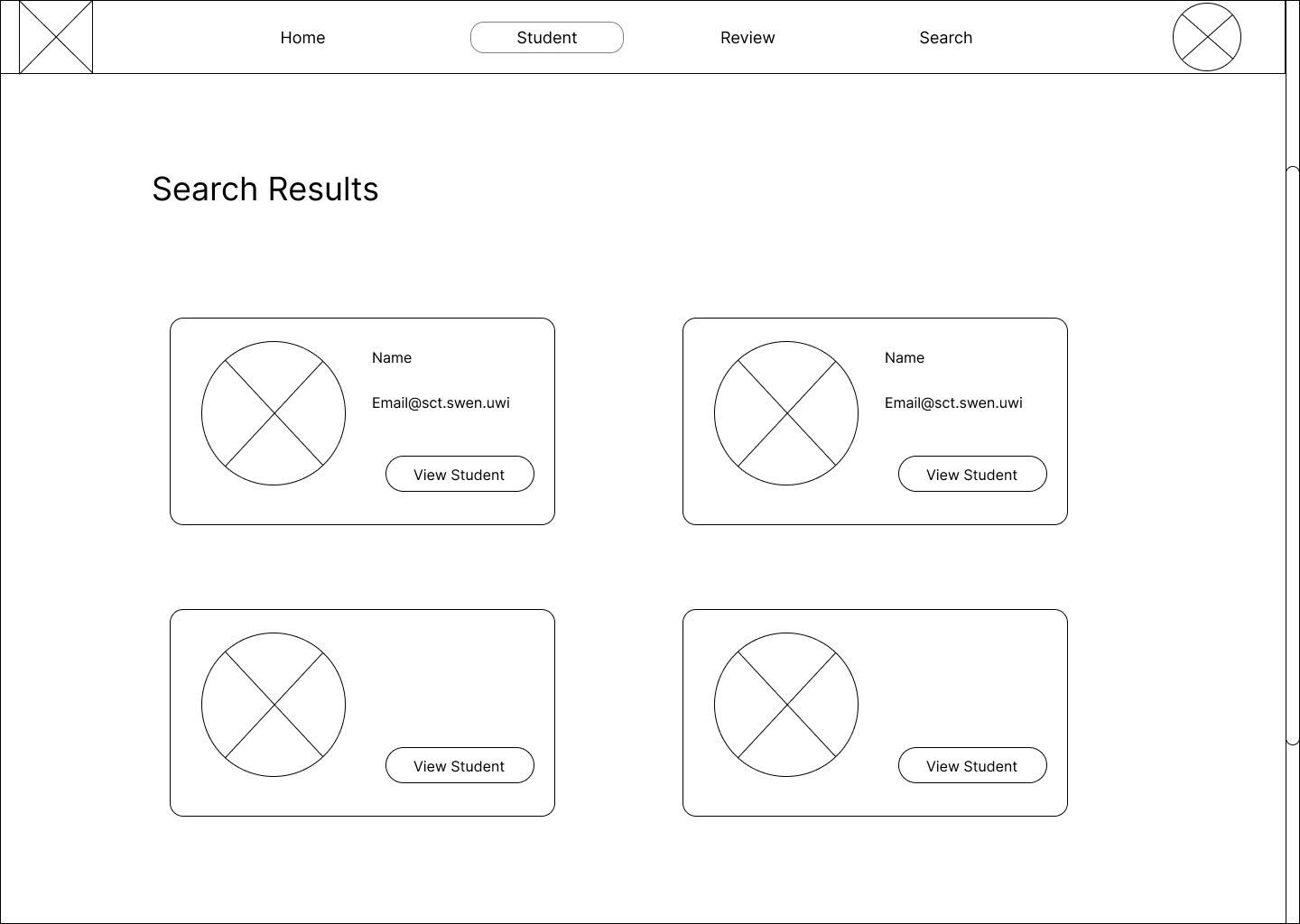
### Add Student



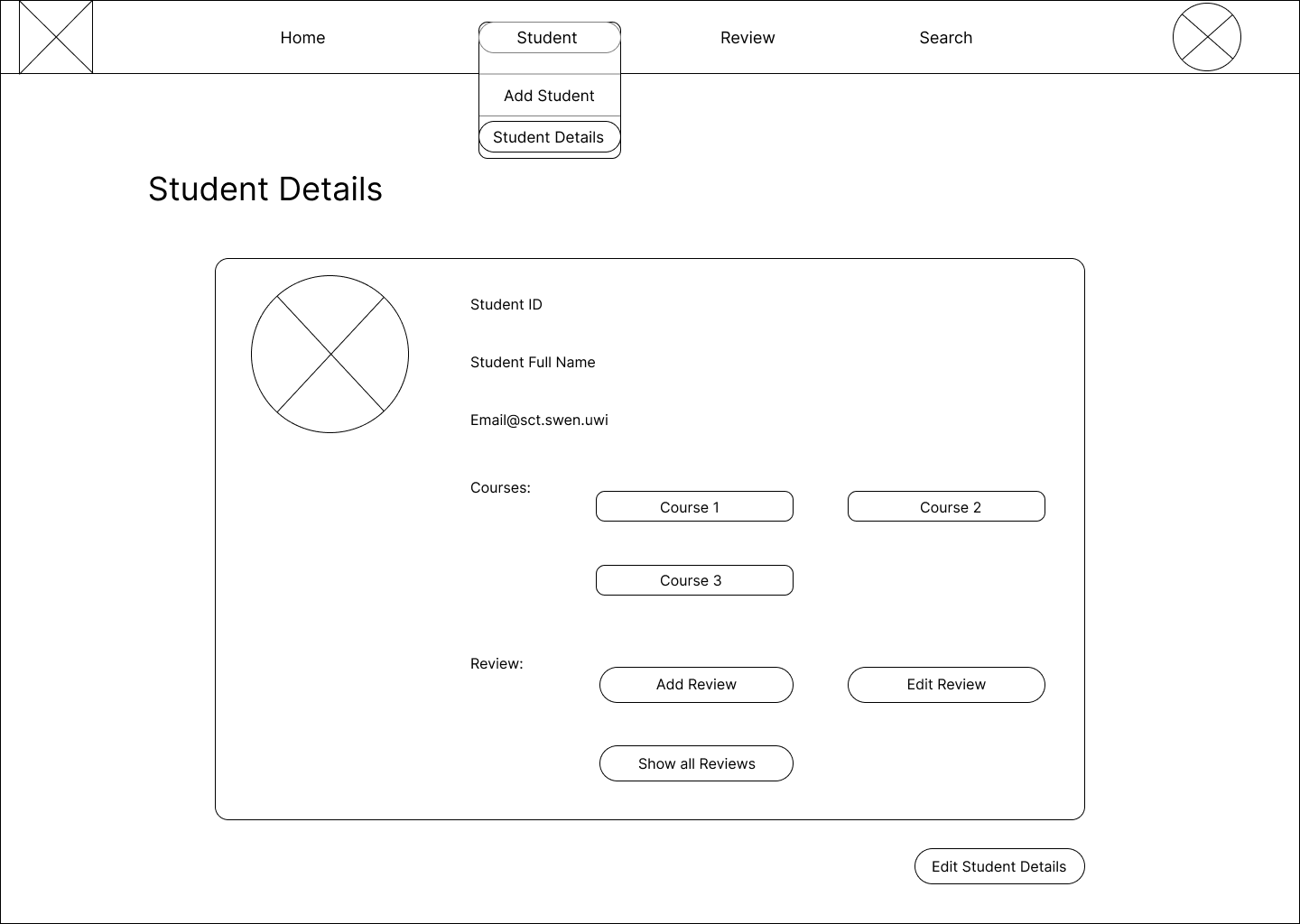
### Search Student



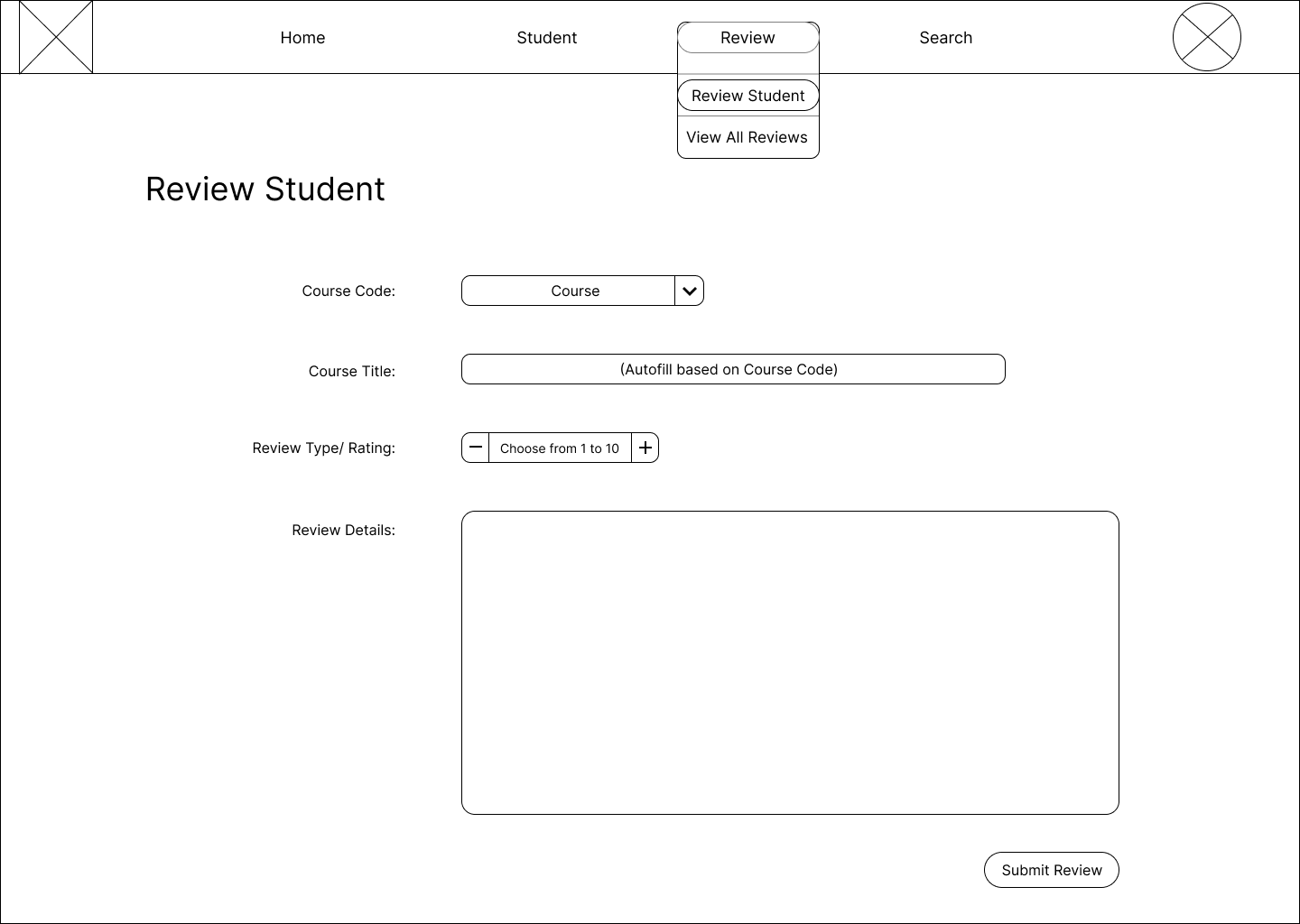
### Search Results



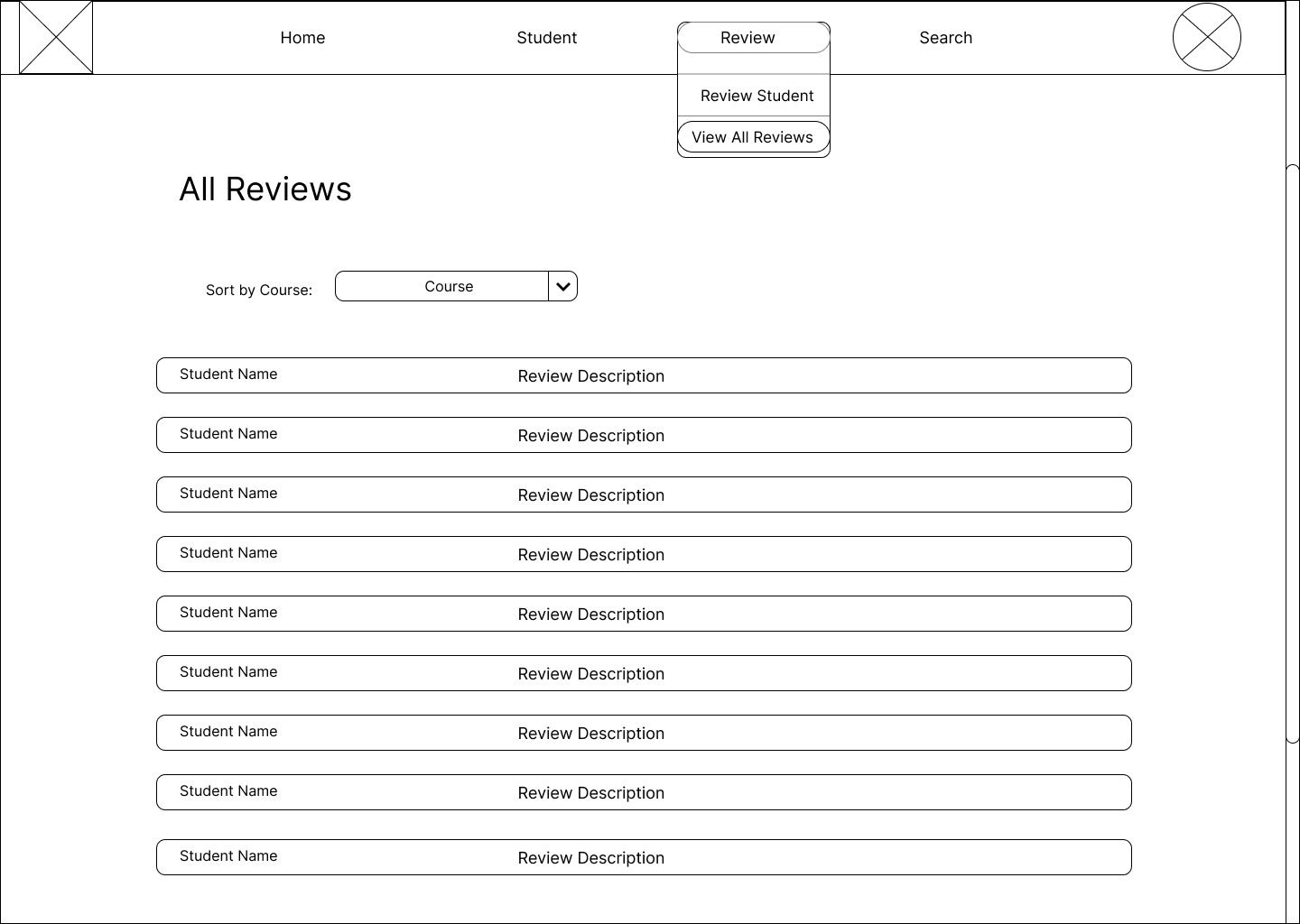
### Student Details



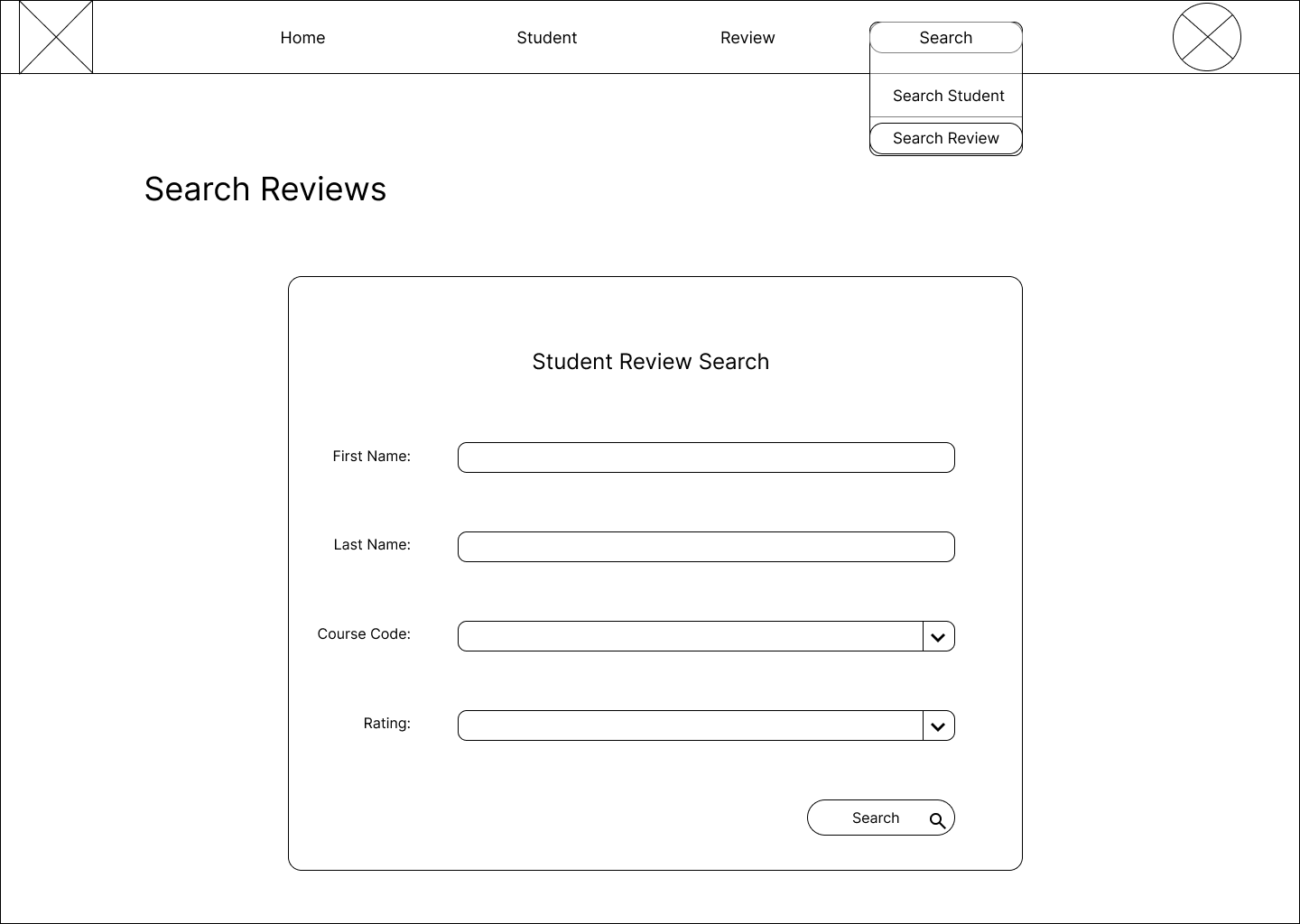
### Review Student



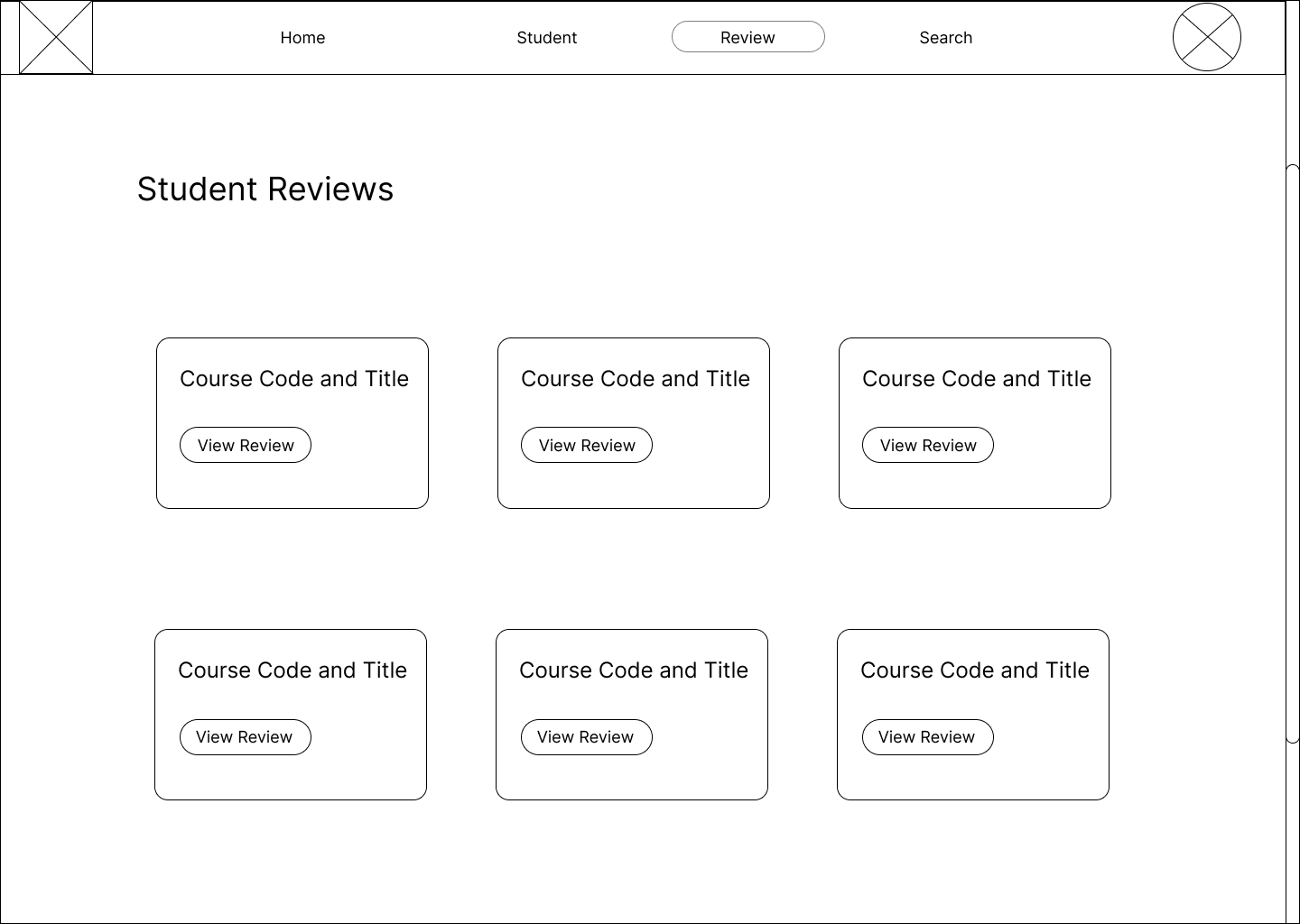
### View All Reviews



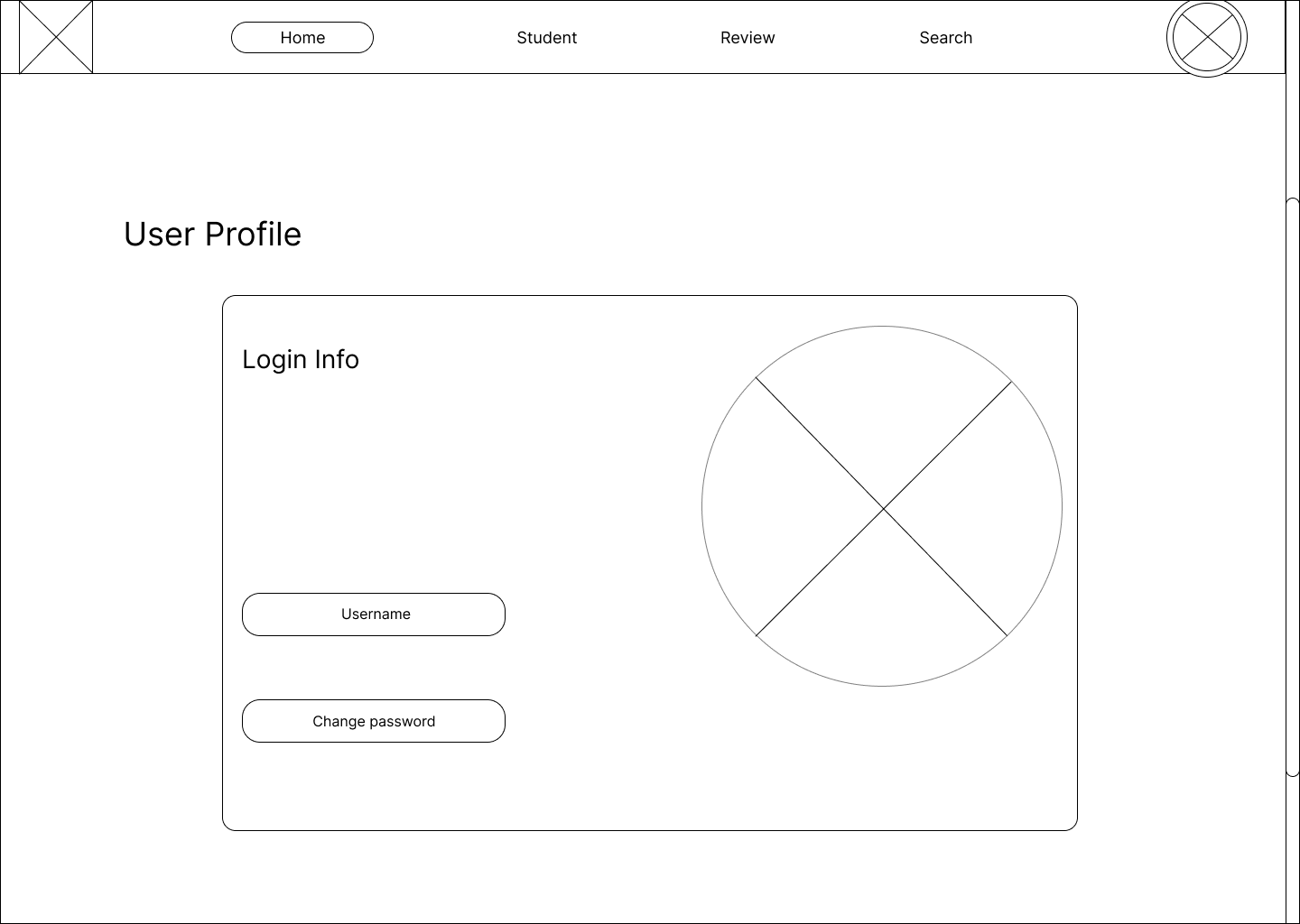
### Search Reviews



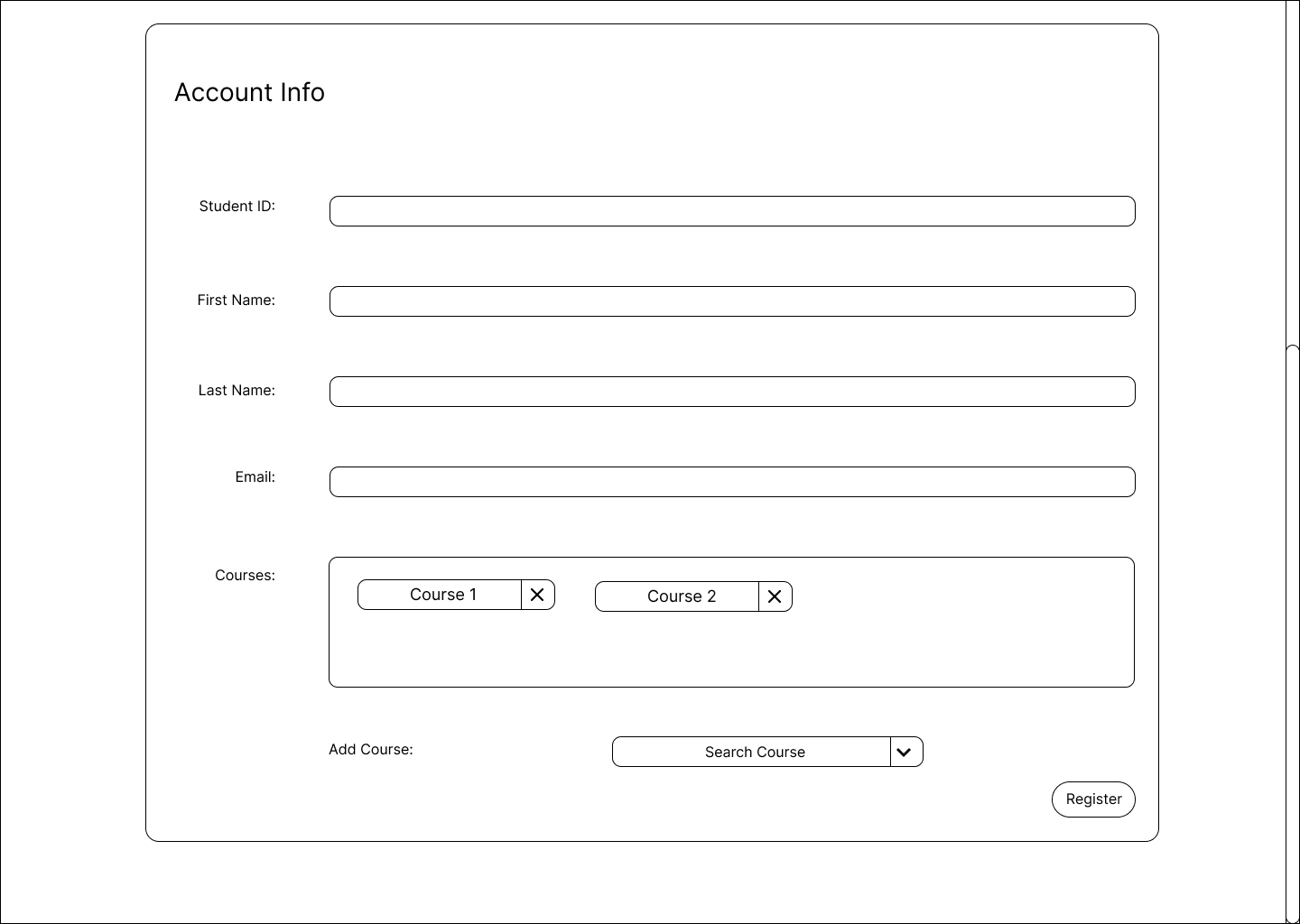
### Student Reviews



### User Profile (Upper)



### User Profile (Lower)



# Test Plans

## User Acceptance Tests

### User Account Creation

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful*** | * None | * Access the authentication page via URL or redirect. * Fill in the sign-up form’s required fields, including a unique username and email. * Submit the form. | * Alert the user to a successful account creation. * Redirect the user back to the authentication page to login to their new account. |
| ***Missing required fields*** | * None | * Access the authentication page via URL or redirect. * Partially fill in the sign-up form’s required fields. * Submit the form. | * Alert the user to a failed sign up due to incomplete data. * Redirect the user back to the authentication page to try again. |
| ***Invalid username format*** | * None | * Access the authentication page via URL or redirect. * Use a non-alphanumeric username for the sign-up form. * Submit the form. | * Alert the user to a failed sign up due to an invalid username. * Redirect the user back to the authentication page to try again. |
| ***Invalid email format*** | * None | * Access the authentication page via URL or redirect. * Use an invalid email address for the sign-up form. * Submit the form. | * Alert the user to a failed sign up due to an invalid email address. * Redirect the user back to the authentication page to try again. |
| ***Duplicate username*** | * A valid user profile that shares the target username must already exist. | * Access the authentication page via URL or redirect. * Use a pre-existing username for the sign-up form. * Submit the form. | * Alert the user to a failed sign up due to a duplicate username. * Redirect the user back to the authentication page to try again. |
| ***Duplicate email address*** | * A valid user profile that shares the target email address must already exist. | * Access the authentication page via URL or redirect. * Use a pre-existing email for the sign-up form. * Submit the form. | * Alert the user to a failed sign up due to a duplicate email. * Redirect the user back to the authentication page to try again. |
| ***Password and password confirmation do not match*** | * None | * Access the authentication page via URL or redirect. * Enter a password pre-existing username for the sign-up form. * Submit the form. | * Alert the user to a failed sign up due to a duplicate username. * Redirect the user back to the authentication page to try again. |

### 

### User Login

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful*** | * A valid user profile must already exist. | * Access the authentication page via URL or redirect. * Fill in the required fields for the login form. * Submit the form. | * Alert the user to a successful login. * Navbar now displays user profile and logout buttons. * Redirect the user to the home page. |
| ***Missing required fields*** | * A valid user profile must already exist. | * Access the authentication page via URL or redirect. * Partially fill in the form’s required fields. * Submit the form. | * Alert the user to a failed login due to incomplete data. * Redirect the user back to the authentication page to try again. |
| ***Invalid username or password*** | * A valid user profile must already exist. | * Access the authentication page via URL or redirect. * Use an invalid username/password for the login form. * Submit the form. | * Alert the user to a failed login due to invalid data. * Redirect the user back to the authentication page to try again. |
| ***User already logged in*** | * The user is currently logged in to their account. | * Attempt to access the authentication page via URL. | * Redirect the user to the home page. |
| ***Duplicate username*** | * A valid user profile that shares the target username must already exist. | * Access the authentication page via URL or redirect. * Use a pre-existing username for the sign-up form. * Submit the form. | * Alert the user to a failed sign up due to a duplicate username. * Redirect the user back to the authentication page to try again. |
| ***Duplicate email address*** | * A valid user profile that shares the target email address must already exist. | * Access the authentication page via URL or redirect. * Use a pre-existing email for the sign-up form. * Submit the form. | * Alert the user to a failed sign up due to a duplicate email. * Redirect the user back to the authentication page to try again. |

### 

### User Logout

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful*** | * The user is currently logged in to their account. | * Press the logout button in the navbar. | * Alert the user to a successful logout. * Navbar no longer displays user profile and logout buttons. * Redirect the user to the authentication page. |
| ***User not logged in*** | * The user is ***not*** currently logged in to their account. | * Attempt to access the logout page via URL. | * Redirect the user to the authentication page to log in. |

### Add Student Profile

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful*** | * The user is currently logged in to their account. | * Access the student record creation page via the navbar or URL. * Fill in the form’s required fields including both a unique student ID and email. * Submit the form. | * Alert the user to a successful student record creation. * Redirect the user to the new student’s profile page. |
| ***User not logged in*** | * The user is ***not*** currently logged in to their account. | * Attempt to access the student record creation page via URL. | * Redirect the user to the authentication page to log in. |
| ***Missing required fields*** | * The user is currently logged in to their account. | * Access the student record creation page via the navbar or URL. * Partially fill in the form’s required fields. * Submit the form. | * Alert the user to a failed student record creation due to incomplete data. * Redirect the user back to the student record creation page. |
| ***Invalid student ID format*** | * The user is currently logged in to their account. | * Access the student record creation page via the navbar or URL. * Use an invalid student ID for the form. * Submit the form. | * Alert the user to a failed student record creation due to an invalid student ID. * Redirect the user back to the student record creation page. |
| ***Invalid name format*** | * The user is currently logged in to their account. | * Access the student record creation page via the navbar or URL. * Use a non-alphabetic name for the form. * Submit the form. | * Alert the user to a failed student record creation due to an invalid name. * Redirect the user back to the student record creation page. |
| ***Duplicate student ID*** | * The user is currently logged in to their account * A valid user profile that shares the target username must already exist. | * Access the student record creation page via the navbar or URL. * Use a pre-existing student ID for the form. * Submit the form. | * Alert the user to a failed student record creation due to an invalid student ID. * Redirect the user back to the student record creation page. |
| ***Duplicate email*** | * The user is currently logged in to their account * A valid user profile that shares the target email address must already exist. | * Access the student record creation page via the navbar or URL. * Use a pre-existing email for the form. * Submit the form. | * Alert the user to a failed student record creation due to an invalid email address. * Redirect the user back to the student record creation page. |

### 

### Search Student

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful - Empty Search*** | * The user is currently logged in to their account. | * Access the student search page via the navbar or URL. * Submit the form without providing any filters. | * Redirect the user to the student search result page listing ***all*** students in the database. |
| ***Successful - Student found*** | * The user is currently logged in to their account. * At least one student profile matching the provided filters must already exist. | * Access the student search page via the navbar or URL. * Provide the desired filters in the search form. * Submit the form. | * Redirect the user to the student search result page listing all student profiles that match the provided filters. |
| ***Successful - Student not found*** | * The user is currently logged in to their account. * No student profiles matching the provided filters currently exist. | * Access the student search page via the navbar or URL. * Provide the desired filters in the search form. * Submit the form. | * Redirect the user to the student search result page stating there were no results found. |
| ***User not logged in*** | * The user is ***not*** currently logged in to their account. | * Attempt to access the student search/search results page via URL. | * Redirect the user to the authentication page to log in. |
| ***Invalid student ID format*** | * The user is currently logged in to their account. | * Access the student record creation page via the navbar or URL. * Provide a non-numeric student ID. * Submit the form. | * Alert the user to a failed student record creation due to an invalid student ID. * Redirect the user back to the student search page. |
| ***Invalid name format*** | * The user is currently logged in to their account. | * Access the student record creation page via the navbar or URL. * Provide a non-alphabetic first/last name. * Submit the form. | * Alert the user to a failed student record creation due to an invalid name. * Redirect the user back to the student search page. |

### 

### View Student Profile

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful*** | * The user is currently logged in to their account. * The target student profile must already exist. | * Access a student’s profile (e.g., from search results, student listings, URL). | * Redirect the user to the target student’s profile page. |
| ***User not logged in*** | * The user is ***not*** currently logged in to their account. | * Attempt to access a student’s profile page (e.g., from search results, student listings, URL). | * Redirect the user to the authentication page to log in. |
| ***Student not found*** | * The user is currently logged in to their account. | * Attempt to directly access a non-existent student profile page via URL. | * Redirect the user to the “404 - Not Found” page. |

### Review Student

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful*** | * The user is currently logged in to their account. * A valid student profile must already exist. | * Access the student review creation page via the given student’s profile or URL. * Fully complete the form with valid data (including both a unique student ID and email). * Submit the form. | * Alert the user to a successful review creation. * Redirect the user to the new review’s page. |
| ***User not logged in*** | * The user is ***not*** currently logged in to their account. | * Attempt to access the student review creation page via URL. | * Redirect the user to the authentication page to log in. |
| ***Missing required fields*** | * The user is currently logged in to their account. | * Access the student record creation page via the navbar or URL. * Partially fill in the form’s required fields. * Submit the form. | * Alert the user to a failed student record creation due to incomplete data. * Redirect the user back to the student record creation page. |
| ***Student not found*** | * The user is currently logged in to their account. | * Attempt to access the review page for a non-existent student via URL. | * Redirect the user to the “404 - Not Found” page. |

### 

### Search Reviews

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful -***  ***Empty Search*** | * The user is currently logged in to their account. | * Access the review search page from the navbar or via URL. * Submit the form without providing any filters. | * Redirect the user to the reviews list page, listing all reviews in the database. |
| ***Successful -***  ***(At least one review has been made)*** | * The user is currently logged in to their account. * At least one (1) valid student profile must already exist. * At least one (1) valid student review for any given student must already exist. | * Access the review search page from the navbar or via URL. * Fill in the desired fields of the filter search form. * Submit the form. | * Redirect the user to the reviews list page, listing all reviews in the database that match the given filters. |
| ***Successful -***  ***(No reviews have been made)*** | * The user is currently logged in to their account. * No valid student reviews have yet been made. | * Access the review search page from the navbar or via URL. * Fill in the desired fields of the filter search form. * Submit the form. | * Redirect the user to the reviews list page, with a message saying no reviews currently exist. |
| ***User not logged in*** | * The user is ***not*** currently logged in to their account. | * Attempt to access the reviews search/search results page via URL. | * Redirect the user to the authentication page to log in. |

### View Review

| **Test Case** | **Pre-conditions** | **Test Steps** | **Test Criteria** |
| --- | --- | --- | --- |
| ***Successful*** | * The user is currently logged in to their account. * A valid student profile must already exist. * A valid review for the given student must already exist. | * Access a specific student review from the student’s profile, search results, or via URL. | * Redirect the user to review for a given student. |
| ***User not logged in*** | * The user is ***not*** currently logged in to their account. | * Attempt to access the review page via URL. | * Redirect the user to the authentication page to log in. |
| ***Review not found*** | * The user is currently logged in to their account. | * Attempt to access a non-existent review page via URL. | * Redirect the user to the “404 - Not Found” page. |

# 

## Unit Tests

| **Unit Test** | **Module Tested** | **Expected Output** |
| --- | --- | --- |
| **test\_create\_user** | User(“bob”, “bobpass”, bob.smith@mail.com) | * user.username == “bob” * user. email == “bob.smith@mail.com” |
| **test\_user\_check\_password** | User.check\_password(“bobpass”) | * user.check\_password(“bobpass”) == True * user.check\_password(“FAKEpass”) == False |
| **test\_user\_set\_password** | User.set\_password(“newpass”) | * user.password\_hash != generate\_password\_hash(“bobpass”) |
| **test\_user\_get\_json()** | User.get\_json() | {  “id”: 1,  “username”: “bob”  “email”: “bob.smith@mail.com”  } |
| **test\_create\_student()** | Student(816012345, “jane”, “smith”, jane@mail.com) | * student.public\_id == 816012345 * student.firstname == “jane” * student.lastname == “smith” * student.email == “jane@mail.com” |
| **test\_ student\_to\_ json()** | Student.to\_json() | {  “private\_id”: None,  “public\_id”: 816012345  “firstname”: “jane”,  “lastname”: “smith”,  “email”: “jane@mail.com”  “reviews”: []  } |
| **test\_create\_review()** | Review(1, “Test Review”, 1) | * review.user\_id == 1 * review.student\_private\_id == 1 * review.text == “Test Review” |
| **test\_review\_to\_json()** | Review.to\_json() | {  user\_id: 1,  student\_private\_id: 1,  'text': "Test Review"  } |

## Integration Tests

### User Operations

| **Test** | **Dependencies** | **Description** |
| --- | --- | --- |
| **test\_create\_user** | * create\_user() * get\_user() | Creates a user record in the database and ensures it has the correct values. |
| **test\_get\_user()** | * create\_user() * get\_user\_by\_id() | Ensures that a user record with a specific ID can be successfully retrieved from the database. |
| **test\_get\_user\_by\_username()** | * create\_user() * get\_user\_by\_username() | Ensures that a user record *with a specific username* can be successfully retrieved from the database. |
| **test\_user\_to\_json()** | * create\_user() * get\_user\_json()   + User.to\_json() | Ensures that a given user record can be properly converted to JSON format. |
| **test\_get\_all\_users()** | * create\_user() * get\_all\_users() | Ensures that the data of all user records in the database can be retrieved. |
| **test\_get\_all\_users\_json()** | * create\_user() * get\_all\_users\_json()   + User.to\_json() | Ensures that all user records in the database can be collectively converted to JSON format. |

### Student Operations

| **Test** | **Dependencies** | **Description** |
| --- | --- | --- |
| **test\_create\_student()** | * create\_student() * get\_student() | Creates a student record in the database and ensures it has the correct values. |
| **test\_ student\_to\_ json()** | * get\_student\_json()   + Student.to\_json() | Ensures that a given student record can be properly converted to JSON format. |
| **test\_search\_student\_no\_parameters()** | * create\_student() * search\_student() | Verifies that searching with no parameters returns all students in the system. |
| **test\_search\_student\_by \_public\_id()** | * create\_student() * search\_student() | Ensures that the public ID search filter returns matching values. |
| **test\_search\_student\_by\_first\_name()** | * create\_student() * search\_student() | Ensures that the first name search filter returns matching values. |
| **test\_search\_student\_by\_last\_name()** | * create\_student() * search\_student() | Ensures that the last name search filter returns matching values. |

### Review Operations

| **Test** | **Dependencies** | **Description** |
| --- | --- | --- |
| **test\_create\_review** | * create\_user() * create\_student() * get\_student() * create\_review() * get\_review() | Ensure the review record in the database has the correct values. |
| **test\_review\_to\_json()** | * create\_review() * get\_review\_json()   + Review.to\_json() | Ensure a given review record can be correctly converted to JSON format.  } |
| **test\_search\_review\_no\_parameters()** | * create\_user() * create\_student() * get\_student() * create\_review() * get\_review() | Verifies that searching with no parameters returns all students in the system. |
| **test\_search\_student\_by \_user\_id()** | * create\_student() * search\_student() | Ensures that the public ID search filter returns matching values. |
| **test\_search\_student\_by\_student\_public\_id()** | * create\_student() * search\_student() | Ensures that the first name search filter returns matching values. |
| **test\_search\_student\_by\_last\_name()** | * create\_student() * search\_student() | Ensures that the last name search filter returns matching values. |

## 

## API Tests

### User Account Creation

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Account Creation: *Successful*** | POST | /auth | None | {  “username”: “bob”,  “password”: “bobpass”,  “confirm\_password”: “bobpass”,  “email”: “bob.smith@mail.com”  } | {  “message”: “account created”  } | 201 | User is redirected to the **/home** page. |
| **Test Account Creation: *Missing Required Fields*** | POST | /auth | None | {  “username”: “bob”,  “password”: None,  “confirm\_password”: “bobpass”,  “email”: “bob.smith@mail.com”  } | {  “error”: “missing required fields”  } | 400 | User is redirected back to the **/auth** page. |
| **Test Account Creation: *Invalid Username*** | POST | /auth | None | {  “username”: “b21f@!ob}”,  “password”: “bobpass”,  “confirm\_password”: “bobpass”,  “email”: “bob.smith@mail.com”  } | {  “error”: “invalid username”  } | 400 | User is redirected back to the **/auth** page. |
| **Test Account Creation: *Invalid Email*** | POST | /auth | None | {  “username”: “bob”,  “password”: “bobpass”,  “confirm\_password”: “bobpass”,  “email”: “2bob.mal”  } | {  “error”: “invalid email”  } | 400 | User is redirected back to the **/auth** page. |
| **Test Account Creation: *Duplicate Username*** | POST | /auth | A valid user with the target username exists. | {  “username”: “bob”,  “password”: “bobpass”,  “confirm\_password”: “bobpass”,  “email”: “bob.smith@mail.com”  } | {  “error”: “username already taken”  } | 400 | User is redirected back to the **/auth** page. |
| **Test Account Creation: *Duplicate Email*** | POST | /auth | A valid user with the target email exists. | {  “username”: “jane”,  “password”: “janepass”,  “confirm\_password”: “janepass”,  “email”: “bob.smith@mail.com”  } | {  “error”: “email already taken”  } | 400 | User is redirected back to the **/auth** page. |

### 

### User Login

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Login: *Successful*** | POST | /auth | A valid user exists. | {  “username”: “bob”,  “password”: “bobpass”  } | {  “message”:”login successful”,  “access\_token”: <jwt\_token>  } | 200 | User is redirected to the **/home** page. |
| **Test Login: *Missing Required Fields*** | POST | /auth | A valid user exists. | {  “username”: “bob”,  “password”: None  } | {  “error”: “missing required fields”  } | 400 | User is redirected back to the **/auth** page. |
| **Test Login: *Invalid Username or Password*** | POST | /auth | A valid user exists. | {  “username”: bob,  “password”: janepass  } | {  “error”: “invalid credentials”  } | 401 | User is redirected back to the **/auth** page. |
| **Test Login: *User Already Logged In*** | GET | /auth | User is already logged in. | Authorization: Bearer <jwt\_token> | {  “error”: “invalid email”  } | 302 | User is redirected to the **/home** page. |

### 

### User Logout

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Logout: *Successful*** | POST | /auth/logout | User is logged in. | Authorization: Bearer <jwt\_token> | {  “message”:”logout successful”,  “access\_token”: <jwt\_token>  } | 200 | User is redirected to the **/auth** page for login. |
| **Test Logout: *Still Logged In*** | POST | /auth/logout | User is logged in but the server fails to log them out. | Authorization: Bearer <jwt\_token> | {  “error”: “logout failed”  } | 500 | User is redirected back to the **/home** page. |

### Add Student Profile

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Add Student: *Successful*** | POST | /students/add | **Pre-condition:** User is logged in. | Authorization: Bearer <jwt\_token>  {  “public\_id”: 816012345,  “firstname”: “jane”,  “lastname”: “smith”,  “email”: “jane.smith@mail.com”  } | {  “message”: “student created”  } | 201 | User is redirected to the new student’s **/students/<public\_id>** page. |
| **Test Add Student: *Missing Required Fields*** | POST | /students/add | User is logged in. | Authorization: Bearer <jwt\_token>  {  “public\_id”: 816012345,  “firstname”: “”,  “lastname”: “smith”  } | {  “error”: “missing required fields”  } | 400 | User is redirected back to the **/students/add** page. |
| **Test Add Student: *Invalid Student ID Format*** | POST | /students/add | User is logged in. | Authorization: Bearer <jwt\_token>  {  “public\_id”: “bad\_id”,  “firstname”: “jane”,  “lastname”: “smith”,  “email”: “jane.smith@mail.com”  } | {  “error”: “invalid student id format”  } | 400 | User is redirected back to the **/students/add** page. |
| **Test Add Student: *Invalid Name Format*** | POST | /students/add | User is logged in. | Authorization: Bearer <jwt\_token>  {  “public\_id”: 816012345,  “firstname”: “14jane”,  “lastname”: “s}mi\_+th”,  “email”: “jane.smith@mail.com”  } | {  “error”: “invalid name format”  } | 400 | User is redirected back to the **/students/add** page. |
| **Test Add Student: *Duplicate Student ID*** | POST | /students/add | User is logged in. | Authorization: Bearer <jwt\_token>  {  “public\_id”: 816012345,  “firstname”: “alice”,  “lastname”: “smith”,  “email”: “alice.smith@mail.com”  } | {  “error”: “student id already taken”  } | 400 | User is redirected back to the **/students/add** page. |
| **Test Add Student: *Duplicate Email*** | POST | /students/add | User is logged in. | Authorization: Bearer <jwt\_token>  {  “public\_id”: 816054321,  “firstname”: “alice”,  “lastname”: “smith”,  “email”: “jane.smith@mail.com”  } | {  “error”: “email already taken”  } | 400 | User is redirected back to the **/students/add** page. |

### Search Student

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Search Student: *Empty Search*** | GET | /search/students | User is logged in; no filters are provided. | Authorization: Bearer <jwt\_token> | List all students in the database.  Include links to individual profiles: **/students/profile/<public\_id>** | 200 | User is redirected to the search results page. |
| **Test Search Student: *Student Not Found*** | GET | /search/students?q={filters} | User is logged in; no students match the given filters. | Authorization: Bearer <jwt\_token>  {<filters>} | {  “message”: “no results found”  } | 200 | User is redirected to the search results page. |
| **Test Search Student: *Student Found*** | GET | /search/students?q={filters} | User is logged in; at least one student matches the given filters. | Authorization: Bearer <jwt\_token>  {<filters>} | List all students that match the given filters.  Include links to individual profiles: **/students/profile/<public\_id>** | 200 | User is redirected to the search results page. |
| **Test Search Student: *User Not Logged In*** | GET | /search/students?q={filters} | User is not logged in. | None | {  “error”: “authentication required”  } | 302 | User is redirected back to the **/auth** page. |

### View Student Profile

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test View Student Profile: *Success*** | GET | /students/{student\_public\_id} | User is logged in. | Authorization: Bearer <jwt\_token>  {} | (Student profile data) | 200 | User is redirected to the student’s profile page. |
| **Test View Student Profile: *User Not Logged In*** | GET | /students/{student\_public\_id} | User is not logged in. | None | {  “error”: “authentication required”  } | 302 | User is redirected to the **/auth** page. |
| **Test View Student Profile: *Student Not Found*** | GET | /students/{invalid\_student\_public\_id} | User is logged in. | Authorization: Bearer <jwt\_token>  {} | {  “error”: “student not found”  } | 404 | User is redirected to the **‘404 - Not Found’** page. |

### Review Student

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Review Student: *Success*** | POST | /reviews/add | User is logged in; valid student profile exists. | Headers: Authorization: Bearer <jwt\_token>  Body: {  "student\_id": 1,  "user\_id": 1,  "text": "Excellent performance." } | {  “Message”: “review created”  } | 201 | User is redirected to the new review’s page. |
| **Test Review Student: *User Not Logged In*** | GET | /reviews/add | User is not logged in. | None | {  “error”: “authentication required”  } | 302 | User is redirected to the **/auth** page. |
| **Test Review Student: *Missing Required Fields*** | POST | /reviews/add | User is logged in; valid student profile exists. | Authorization: Bearer <jwt\_token>  {  "student\_id": 1,  "user\_id": None,  “text”: None  } | {  “error”: “missing required fields”  } | 400 | Redirect the user back to **/reviews/add**. |
| **Test View Student Profile: *Student Not Found*** | POST | /reviews/add | User is logged in; valid student profile exists. | Authorization: Bearer <jwt\_token> | {  “error”: “student not found”  } | 404 | User is redirected to the **‘404 - Not Found’** page. |

### 

### Search Reviews

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Search Reviews: *Empty Search*** | GET | /search/review | User is logged in; no filters are provided. | Authorization: Bearer <jwt\_token> | List all reviews in the database.  Include link to individual reviews: **/reviews/<review\_id>** | 200 | User is redirected to the search results page. |
| **Test Search Reviews: *Reviews Not Found*** | GET | /search/review?q={filters} | User is logged in; no reviews match the given filters. | Authorization: Bearer <jwt\_token> | {  “message”: “no results found”  } | 200 | User is redirected to the search results page. |
| **Test Search Reviews: *Review Found*** | GET | /search/review?q={filters} | User is logged in; at least one review matches the given filters. | Authorization: Bearer <jwt\_token>  {<filters>} | List all reviews that match the given filters.  Include link to individual reviews: **/reviews/<review\_id>** | 200 | User is redirected to the search results page |
| **Test Search Review: *User Not Logged In*** | GET | /search/review?q={filters} | User is not logged in. | **{**<filters>**}** | {  “error”: “authentication required”  } | 302 | User is redirected to the **/auth** page. |

### View Review

| **Test Case** | **Method** | **URL** | **Pre-**  **condition:** | **Request Headers, Body** | **Response Body** | **Status** | **UI Flow** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test View Review: *Success*** | GET | /reviews/view/{review\_id} | User is logged in; valid student and review exist. | Authorization: Bearer <jwt\_token> | (Review data) | 200 | User is redirected to the review page. |
| **Test View Review: *User Not Logged In*** | GET | /reviews/view/{review\_id} | User is not logged in. | None | {  “error”: “authentication required”  } | 302 | User is redirected to the **/auth** page. |
| **Test Search Reviews: *Review Not Found*** | GET | /search/review/{invalid\_review\_id} | User is logged in; review does not exist | Authorization: Bearer <jwt\_token> | {  “message”: “no results found”  } | 200 | User is redirected to the search results page. |