Project: Student Course Advising Website – Milestone 2

Student name: Brendan Hearrell

Student UIN: 01219737

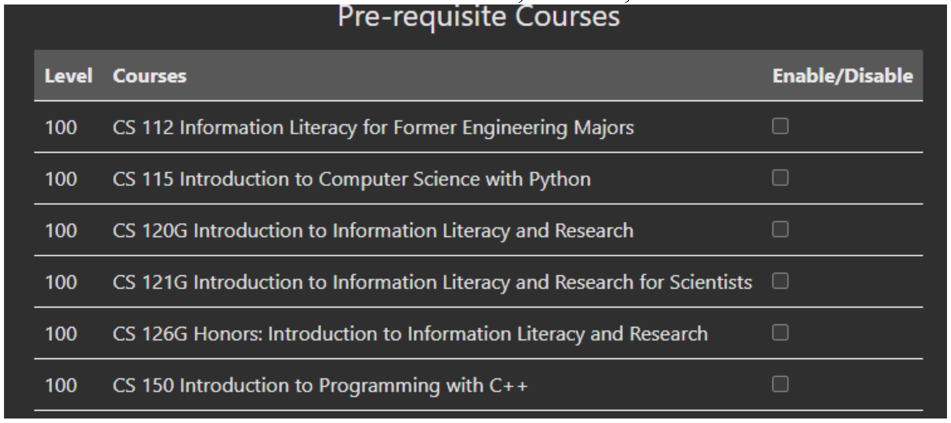
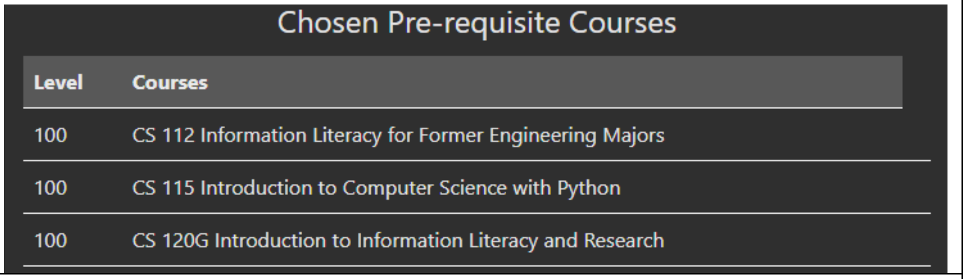
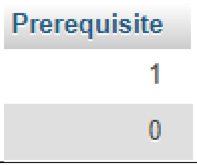
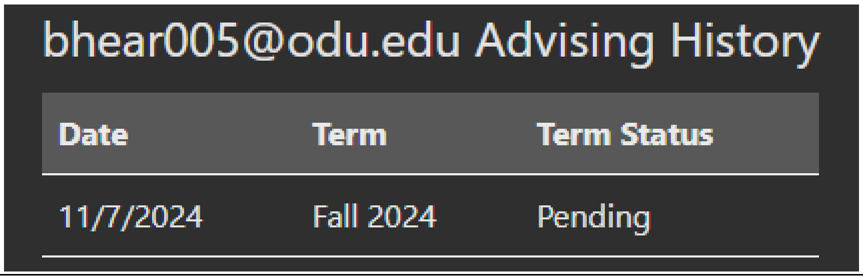
1. Overview

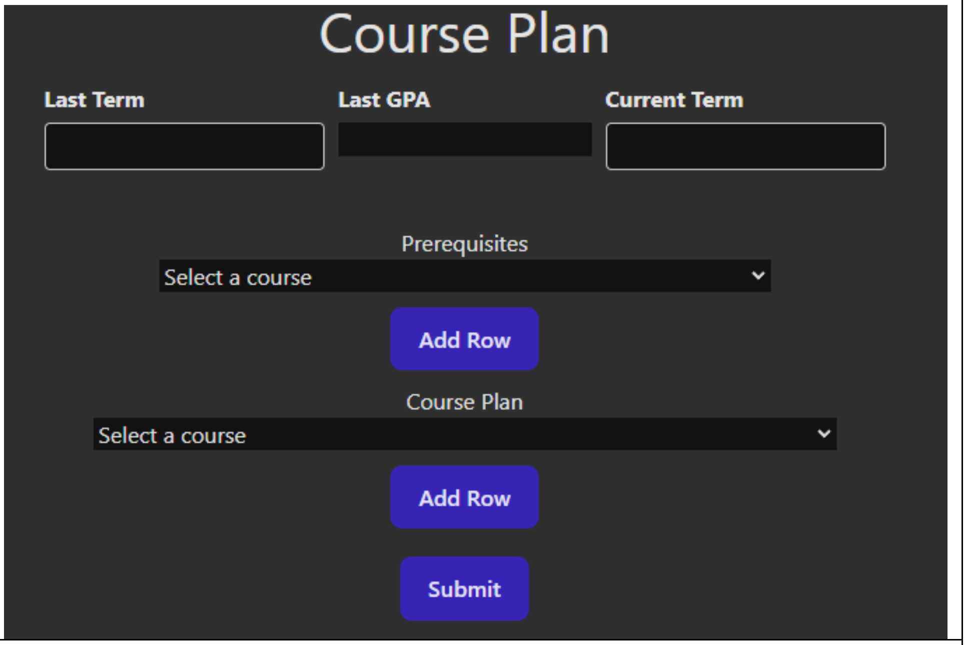
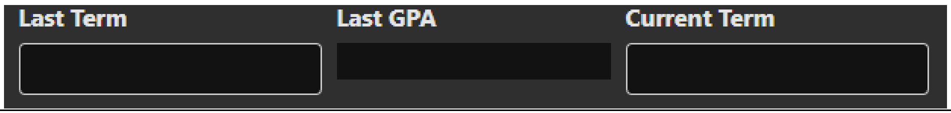
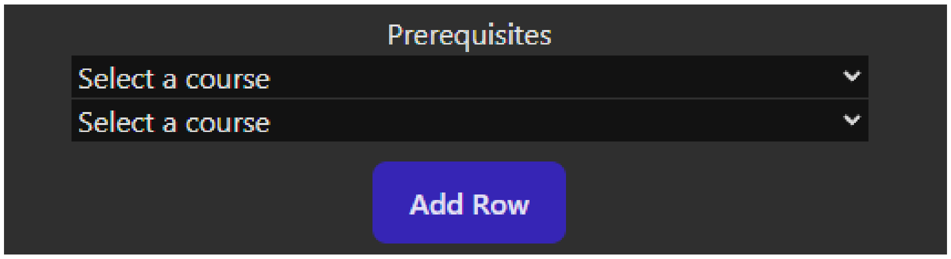
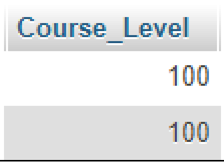
The website is an advising tool for students attending college. The first portion of the project covers account creation, email validation, user login, two factor authentication, account editing, and differentiating between normal user accounts and administrator accounts. The language used for the project will be JavaScript which will utilize the React library and Vite for local development. Node.js will be used as the runtime environment for JavaScript and Express.js will be used for the back-end framework.

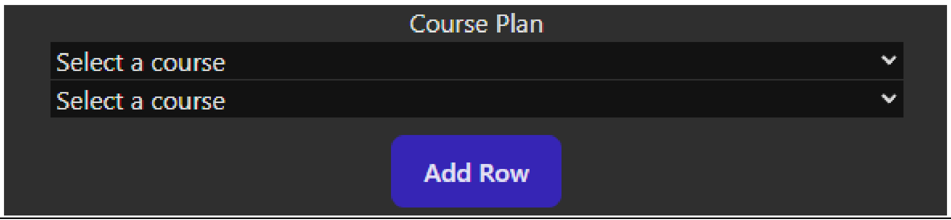
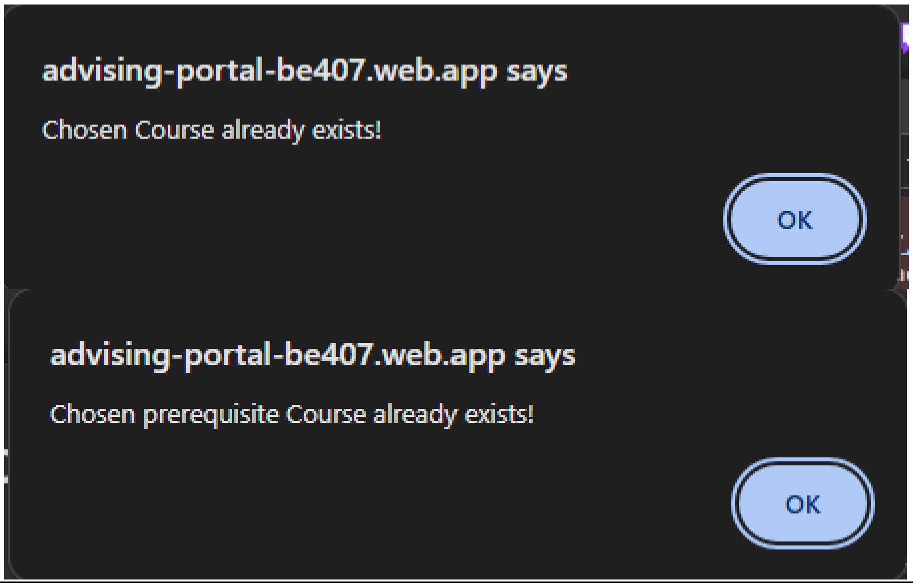
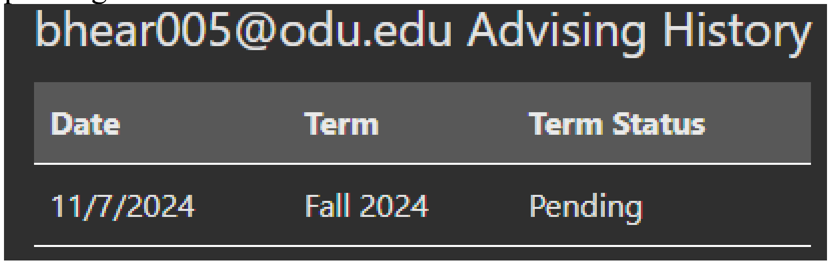
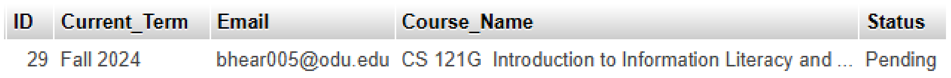
The second portion of the project introduces course advising features to both the admin and the student user. The admin now has the ability to determine prerequisite courses for students and search the advising history of students by email. The student user now has the ability to sign up for classes in the upcoming term and submit them for approval. They will also enter, previous term, current term, and current GPA to be stored for student and admin to view. When the courses are submitted the "header" field along with the courses are stored with a "pending" message, implying that the submission is pending admin approval. The student can also view advising history and all information relating to each term.

[Page intentionally left blank.]

2. Milestone Accomplishments

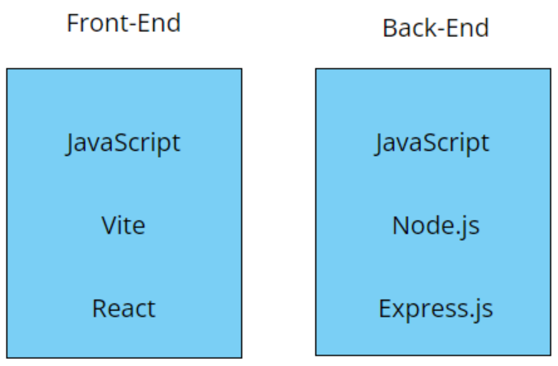
Fulfilled	Feature	Specification
Yes	1	Design and create the prerequisite form for admin, displaying courses from levels 100 to 499 with fields: level, Course, Enable/Disable.  Pre-requisite Courses Level — Courses — Enable/Disable 100 — CS 112 Information Literacy for Former Engineering Majors — ☐ 100 — CS 115 Introduction to Computer Science with Python — ☐ 100 — CS 120G Introduction to Information Literacy and Research — ☐ 100 — CS 121G Introduction to Information Literacy and Research for Scientists — ☐ 100 — CS 126G Honors: Introduction to Information Literacy and Research — ☐ 100 — CS 150 Introduction to Programming with C++ — ☐  Chosen Pre-requisite Courses Level — Courses 100 — CS 112 Information Literacy for Former Engineering Majors 100 — CS 115 Introduction to Computer Science with Python 100 — CS 120G Introduction to Information Literacy and Research
Yes	2	Update database based on the admin's selection of prerequisites.  Prerequisite 1 0
Yes	3	Implement an Advising menu accessible upon student login. Course Sign Up
Yes	4	Design and create the "Course Advising History" form… Advising History  bhear005@odu.edu Advising History Date — Term — Term Status 11/7/2024 — Fall 2024 — Pending

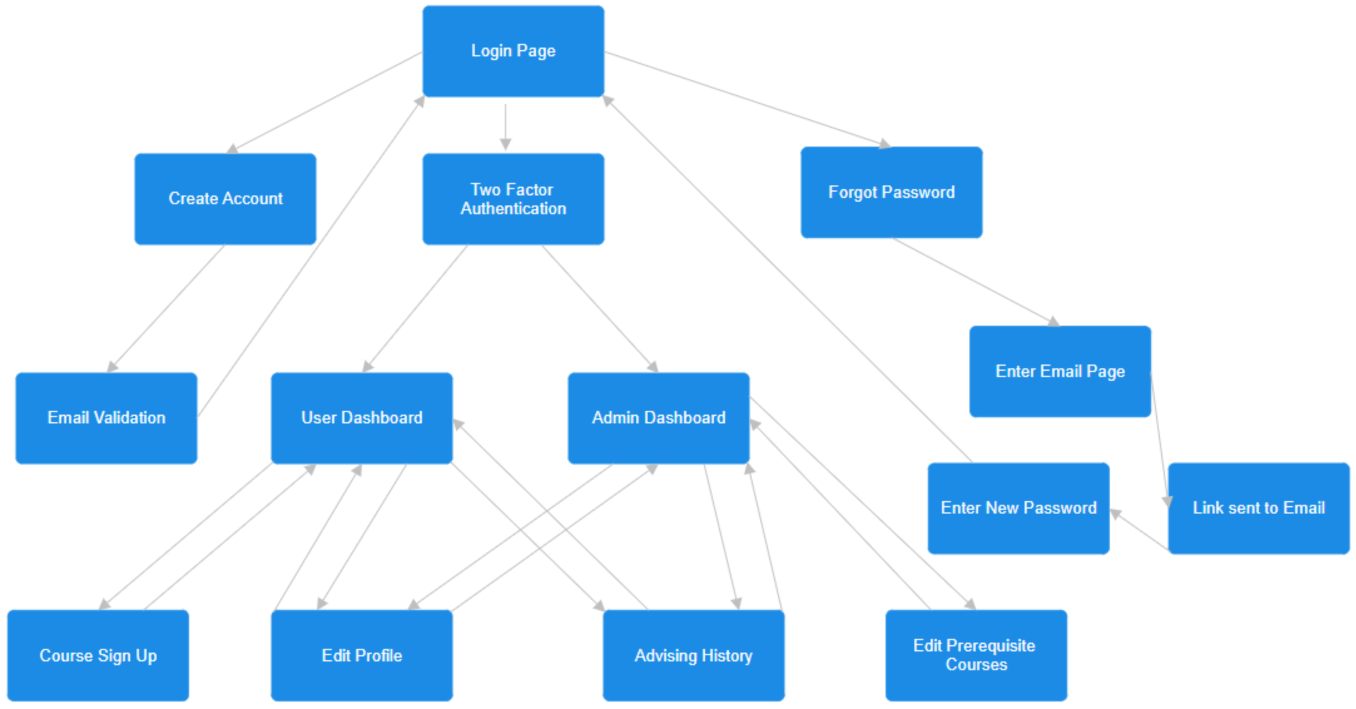
Yes	5	Develop a form for creating new “Course Advising” form with three sections: History, prerequisites, and course plan.  Course Plan Last Term Last GPA Current Term Prerequisites Select a course Add Row Course Plan Select a course Add Row Submit
Yes	6	**Implement the History section with fields: Last Term, Last GPA, Advising Term.** 
Yes	7	Enable dynamic addition of rows for the course section with fields: Level Course, Name.  Prerequisites Select a course Select a course Add Row I forgot about the “level” portion of this requirement but everything else works properly. I also created a Course_Level section in the database for this to be implemented.  Course_Level 100 100

Yes	8	Enable dynamic addition of rows for the course section with fields: Level, Course Name. 
Yes	9	Implement rules for course selection, preventing the addition of courses previously taken in the previous terms.  advising-portal-be407.web.app says Chosen Course already exists! advising-portal-be407.web.app says Chosen prerequisite Course already exists!
Yes	10	Implement functionality to save new entries so that newly created records are displayed in the ‘Course Advising History’ form with a pending status.  bhear005@odu.edu Advising History — Date: 11/7/2024; Term: Fall 2024; Term Status: Pending  ID: 29; Current_Term: Fall 2024; Email: bhear005@odu.edu; Course_Name: CS 121G Introduction to Information Literacy and ...; Status: Pending
No	11	When a user clicks on any record displayed in point 4, they should be redirected to the Course Advising form with the selected record pre-populated. Additionally, if the status of the record is ‘approved’ or ‘rejected,’ the record should be frozen and not editable. If the status is ‘pending,’ the user can take changes and save the record.

		I did not finish number 11, but I started to add functionality for this to happen. The table is a button that will lead to the required page with the relevant term saved to access the required data. I just didn't have time to implement the functionality for the listing on the page.
Yes	12	Deploy your Frontend, Backend, and Database on server and your demo should be demonstrated from live server. I did implement this but also found that it broke some functionality within the prerequisite selection page. A cors error has been popping up in the console and also crashing the server. When it isn't on the server the prerequisite page works but I intend to go back and fix this issue.

3. Architecture





4. Database Design

The Classes table contains all listed courses including fields; ID, Course_Level, Course_Name, and Prerequisite. Course_Level indicates the level of the course taken for identification. Course_Name contains the entire course name. Prerequisite contains the Boolean value indicating the admin selected prerequisite courses.

Field	Type	Key	Example
ID	Int	Primary	1
Course_Level	Varchar		200
Course_Name	Varchar		CS 250 Programming with C++
Prerequisite	Boolean (tinyint)		1

The course_plan table contains all "general" courses that are not indicated as prerequisite and include the fields; ID, Current_Term, Email, Course_Name, and Status. Current_Term indicates the term that the student requested. Email correlates to the user email. Course_Name indicates the user requested course to be taken. Status indicates the approval status from the admin.

Field	Type	Key	Example
ID	Int	Primary	1
Current_Term	Varchar		Fall 2024
Email	Varchar		Bhear005@odu.edu
Course_Name	Varchar		CS 250 Programming with C++
Status	varchar		Pending

The prerequisite table contains all the same fields as the course_plan table. I am considering deleting this table and creating a Boolean value in course_plan to designate prerequisites instead.

Field	Type	Key	Example
ID	Int	Primary	1
Current_Term	Varchar		Fall 2024
Email	Varchar		Bhear005@odu.edu
Course_Name	Varchar		CS 250 Programming with C++
Status	varchar		Pending

[Page intentionally left blank.]

The advising_history table contains the fields; ID, Email, Submission_Date, Term, Last_Term, Last_GPA, and Term_Status. The Email field corresponds to the user email address. The Submission_Date indicates the date that the request was received, rejected, or approved. The Term field indicates the current term the user is signing up for. The Last_Term field indicates the previous term the user attended. The Last_GPA field indicates the most recent GPA of the user. The Term_Status indicates the admin approval status.

Field	Type	Key	Example
ID	Int	Primary	1
Email	Varchar		Bhear005@odu.edu
Submission_Date	Varchar		11/7/2024
Term	Varchar		Fall 2024
Last_Term	Varchar		Spring 2024
Last_GPA	Float		3.01
Term_Status	Varchar		Pending

5. Implementation

Design and create the prerequisite form for admin, displaying courses from levels 100 to 499 with fields: Level, Course, Enable/Disable

The prerequisite form for the administrator contains all courses loaded dynamically from the database. When the checkbox is checked the input is managed through an API call and the course is added to the bottom of the page. This indicates that the Boolean value was successfully changed in the database to indicate a prerequisite course.

Code contained within: Project -> client -> src -> components -> prerequisite.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> user.js

Update database based on the admin's selection of prerequisites.

Once the checkbox is checked an API call executes changing the Boolean value indicating a prerequisite course to 1. The course is then displayed at the bottom of the screen to indicate a successful execution.

Code contained within: Project -> client -> src -> components -> prerequisite.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> user.js

Implement an Advising menu accessible upon student login.

The menu upon student login includes the link to edit profile, look at advising history, and select new courses to take for the upcoming term.

Code contained within: Project -> client -> src -> components -> dashboard.jsx

Design and create the "Course Advising History" form to display previously submitted records or indicate no records. Records will show in the list. You must show the below columns.

The Course Advising History contains Date, Term, and Term Status. Date is either the submitted date by the user or the approval/rejection date from the admin. Term is the current term. Term Status is the state that the user request is in, either Pending, Rejected, or Accepted.

Code contained within: Project -> client -> src -> components -> A_H_StudentView.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> user.js

Develop a form for creating new "Course Advising" form with three sections: History, prerequisites, and course plan.

The Course Advising form is comprised of History which contains term data and user GPA, prerequisites which are determined by the admin, and course plan which contains general courses for the user to sign up for.

Code contained within: Project -> client -> src -> components -> StudentCourseEntry.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> studentadvising.js

Implement the History section with fields: Last Term, Last GPA, Advising Term.

The history fields in the Course Advising form contain Last Term, Current GPA, and Advising Term. These values are stored in the advising_history table and will be used to generate the courses taken previously for the history page.

Code contained within: Project -> client -> src -> components -> StudentCourseEntry.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> studentadvising.js

<u>Enable dynamic addition of rows for prerequisites section with fields: Level, Course Name.</u>

There is a button titled Add Row which can dynamically add rows if the user choses to sign up for more courses. The level dropdown contains 100, 200, 300, and 400 all correlating to the level of the course the user is searching for. The Course Name dropdown contains the course name that corresponds to the level dropdown. Because these are the dropdown menus for prerequisites it is very likely that the higher-level courses will not be contained within.

(I have not implemented the level dropdown, but I intend to.)

Code contained within: Project -> client -> src -> components -> StudentCourseEntry.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> studentadvising.js

<u>Enable dynamic addition of rows for the course section with fields: Level, Course Name.</u>

There is a button titled Add Row which can dynamically add rows if the user choses to sign up for more courses. The level dropdown contains 100, 200, 300, and 400 all correlating to the level of the course the user is searching for. The Course Name dropdown contains the course name that corresponds to the level dropdown.

(I have not implemented the level dropdown, but I intend to.)

Code contained within: Project -> client -> src -> components -> StudentCourseEntry.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> studentadvising.js

<u>Implement rules for course selection, preventing the addition of courses previously taken in the previous terms.</u>

If the user tries to sign up for a class that they have taken previously either in the prerequisite field or general course field, they will be notified via alert stating that they have already taken the prerequisite or general course.

Code contained within: Project -> client -> src -> components -> StudentCourseEntry.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> studentadvising.js

<u>Implement functionality to save new entries so that newly created records are displayed in the 'Course Advising History' form with a Pending status.</u>

When a user successfully signs up for the courses without any conflicts, they will receive a successful submission alert, and their history will be stored in the database with the pending status for the admin to approve. The user can also view or change the fields within the pending form so long as there are no conflicting courses.

(I have yet to implement accessing the pending form for editing but intend to.)

Code contained within: Project -> client -> src -> components -> StudentCourseEntry.jsx

Project -> server -> routes -> classes.js

Project -> server -> routes -> studentadvising.js

<u>When a user clicks on any record displayed in point 4, they should be redirected to the Course Advising form with the selected record pre-populated. Additionally, if the status of the record is 'approved' or 'rejected,' the record should be frozen and not editable. If the status is 'pending,' the user can make changes and save the record.</u>

This is not yet implemented. However, I intend to navigate back to the Course Advising form with all fields filled with relevant information. If the status of the advising history is Accepted or Rejected the student will not be able to change any fields contained in the form. If the status is pending the student will be able to change the form and resubmit for approval.

Code contained within: Project -> client -> src -> components -> StudentCourseEntry.jsx

Project -> client -> src -> components -> CurrentTerm.jsx

(I will likely change the name of CurrentTerm.jsx)

Project -> server -> routes -> classes.js

Project -> server -> routes -> studentadvising.js

<u>Deploy your Frontend, Backend and Database on server and your demo should be demonstrate from live server.</u>

The database is deployed on clever-cloud.com. The frontend is deployed on firebase.google.com. The backend is deployed on render.com. All are currently functional. The .env files within client and server contain data relevant to the deployment of the website.