

# Project Report for Milestone 3

Project: Course Advising Portal

Student Name: Quhura Fathima

Student UIN: 01275914

## Project Description:

The **Course Advising Portal** is a sophisticated web application designed to streamline the academic advising process for students and administrators at Old Dominion University. By combining dynamic user interfaces, robust backend logic, and secure database management, the portal delivers a seamless and secure experience for users.

**Milestone 3** introduces several new features, including advanced advising functionalities, enhanced security measures, and comprehensive testing frameworks, to ensure the portal's usability, reliability, and robustness.

## Key Features Added in Milestone 3:

### 1. Admin Advising Portal:

- Developed a screen to display advising sheets submitted by students in the CS department, showing key details like Student Name, UIN, Term, and Status.
- Created a detailed view for individual advising records, enabling administrators to approve or reject records with feedback.

### 2. Dynamic Status Updates and Notifications:

- Implemented dynamic updates to advising records upon admin approval or rejection.
- Integrated email notifications to inform students about their updated record status and admin feedback.

### 3. Enhanced Student Portal:

- Provided students with the ability to view the updated status of their advising sheets on the **Course Advising History** form.

### 4. Security Enhancements:

- **reCAPTCHA Verification:** Integrated Google's reCAPTCHA to prevent automated bots from accessing the system.

- **Clickjacking Protection:** Implemented middleware to prevent the portal from being embedded in iframes, ensuring clickjacking protection.
- **Password Rules:** Enforced strong password policies requiring uppercase letters, lowercase letters, numbers, special characters, and a minimum length of 8 characters.

#### 5. User Interface Enhancements:

- Added a favicon to enhance branding and improve the portal's overall user interface design.

#### 6. Testing Framework:

- Introduced a comprehensive testing suite to validate backend functionalities, user authentication, advising form submissions, and email notifications.
- Installed and utilized testing tools such as **Mocha**, **Chai**, and **Supertest** for writing and executing test cases.

#### 7. Comprehensive Test Cases:

- Created test cases to validate user signup, login, and advising functionality, ensuring robust system behavior.
- Verified the reCAPTCHA integration, advising status updates, and email notifications as part of automated testing.

### Technologies Used:

#### 1. Frontend:

- **React.js:** Powers the dynamic and responsive user interface.
- **Material-UI:** Ensures accessibility and maintains a modern design standard.

#### 2. Backend:

- **Node.js** and **Express.js:** Serve as the backbone for server-side logic and API management.
- **Nodemailer:** Facilitates email communications for OTP verification and notifications.

#### 3. Database:

- **MySQL:** Manages data persistence securely, storing user details and advising records.

#### 4. Deployment:

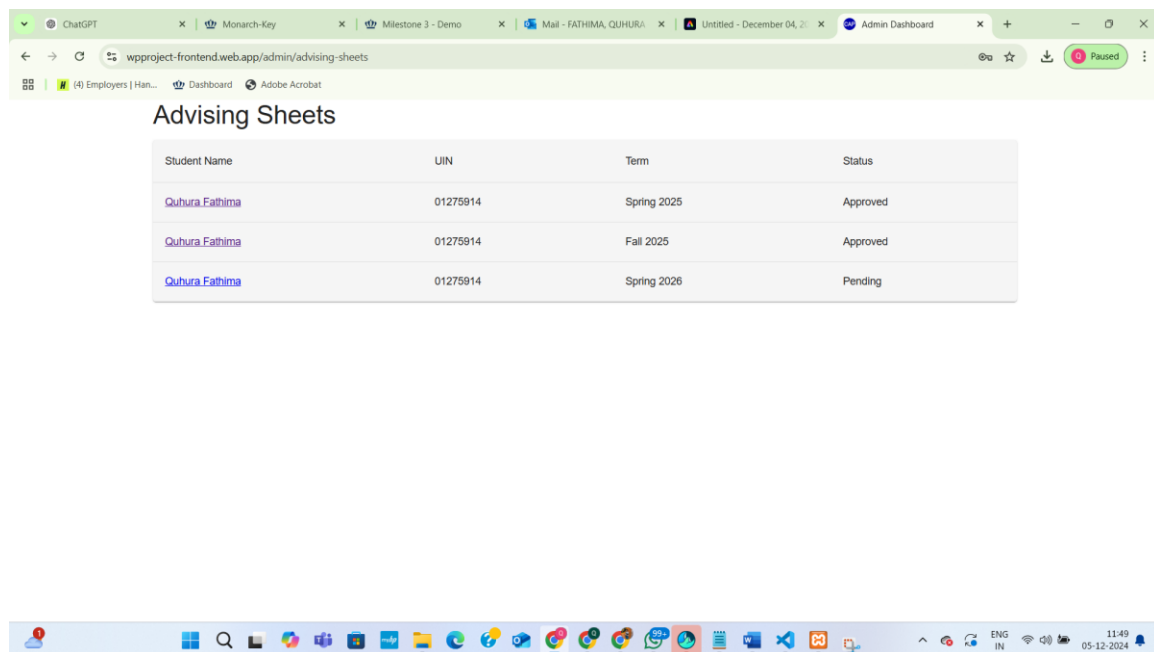
- **Render:** Hosts the backend services with automated deployment capabilities.
- **Firebase:** Hosts the frontend application with SSL encryption and global content delivery.

#### 5. Testing Framework:

- **Mocha:** A feature-rich JavaScript test framework for asynchronous testing.
- **Chai:** An assertion library offering BDD/TDD style assertions.
- **Supertest:** Enables HTTP assertions for testing REST APIs.

This milestone ensures the **Course Advising Portal** is not only feature-rich but also reliable, secure, and user-friendly, meeting the diverse needs of both students and administrators.

#### Admin Advising Sheets page



Student Name	UIN	Term	Status
Quhura Fathima	01275914	Spring 2025	Approved
Quhura Fathima	01275914	Fall 2025	Approved
Quhura Fathima	01275914	Spring 2026	Pending

## Admin Advising Form Page

wpproject-frontend.web.app/admin/advising-form/74

1. Last Name

2. Last GPA

3. Advising Term

Pre-requisites

Pre-requisites

Course Plan

Course Plan

Admin Decision

Approved

Message

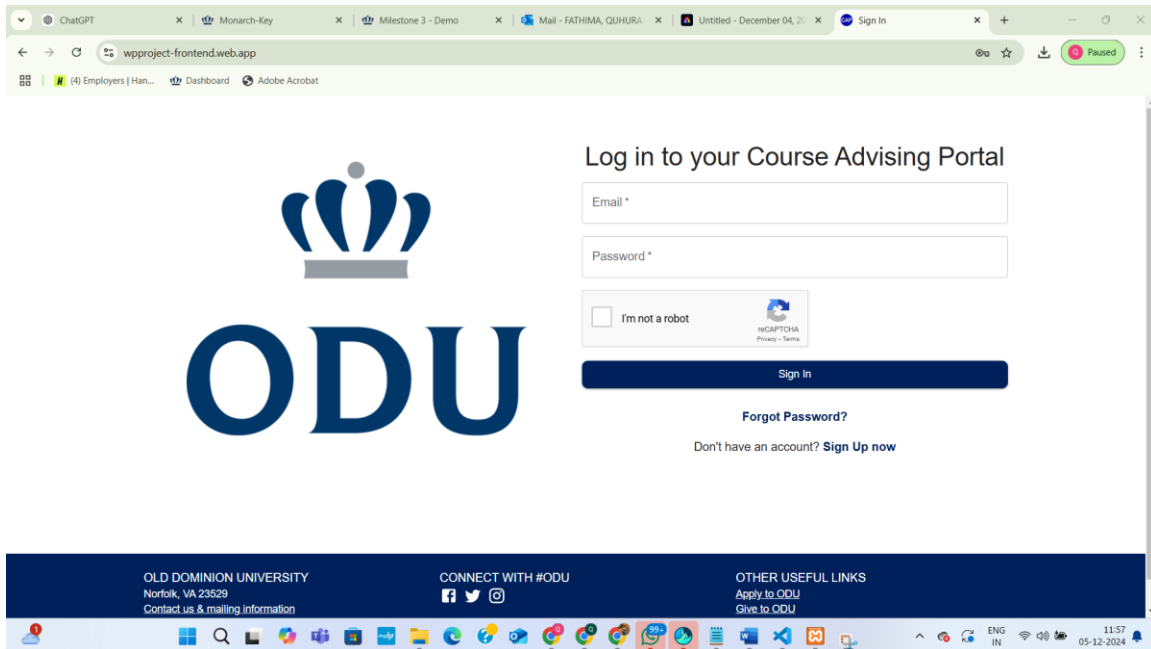
Submit Decision

## Student Course Advising History Page

Course Advising History

Date	Term	Status
2024-12-03T00:00:00.000Z	Spring 2025	Approved
2024-12-03T00:00:00.000Z	Fall 2025	Approved
2024-12-03T00:00:00.000Z	Spring 2026	Pending

## Re-captcha Implementation



## Favicon



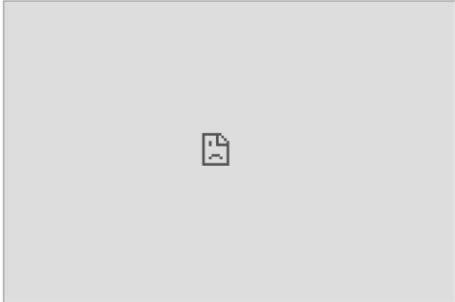
## Click Jacking Test Page

Browser tabs: ChatGPT, Monarch-Key, Milestone 3 - Demo, Mail - FATHIMA, Q... , Untitled - December, Sign In, Clickjacking Test

Address bar: wproject-frontend.web.app/test-files/clickjacking-test.html

Page content:

### Clickjacking Test Page



If your clickjacking protection works, the content above should not be displayed.


Taskbar: Windows Start button, Search, File Explorer, Microsoft Edge, and various application icons. System tray shows ENG IN, Wi-Fi, and the time 11:59 on 05-12-2024.

## Password Rule

Browser tabs: ChatGPT, Monarch-Key, Milestone 3 - Demo, Mail - FATHIMA, Q... , Sign Up, Clickjacking Test

Address bar: wproject-frontend.web.app/signup

Page content:



### Sign Up for the Course Advising Portal

Sign-up failed: Password must be at least 8 characters long and contain letters and numbers.

First Name *	Quhura		Last Name *	Fathima	
Email *	qfath001@odu.edu		Password *	****	
Department *	CS	Degree/Major *	MS	UIN *	01275914

[Sign Up](#)

Already have an account? [Log in](#)

Footer:

OLD DOMINION UNIVERSITY  
Norfolk, VA 23529  
[Contact us & mailing information](#)

CONNECT WITH #ODU  
[f](#) [t](#) [i](#) [o](#)

OTHER USEFUL LINKS  
[Apply to ODU](#)  
[Give to ODU](#)

Taskbar: Windows Start button, Search, File Explorer, and various application icons. System tray shows ENG IN, Wi-Fi, and the time 12:01 on 05-12-2024.

## Test cases

```
user: { email: 'qfath001@odu.edu', isAdmin: 0 }
}
Login Response: {
  message: 'OTP sent. Please verify to continue.',
  email: 'qfath001@odu.edu',
  isAdmin: 0
}
  ✓ should successfully log in an existing user (1146ms)
Login completed for email: qfath001@odu.edu
Verify Login OTP Response: { message: 'Login successful!', isAdmin: false }
  ✓ should verify OTP and complete login process (113ms)
Skipping reCAPTCHA validation in test mode
Generated OTP for qfath001@odu.edu: 418917
OTP sent to qfath001@odu.edu: 250 2.0.0 OK 1733418534 6a1803df08f44-6d8da6968b1sm8406376d6.39 - gsmtp
Session after setting user: Session {
  cookie: {
    path: '/',
    _expires: 2024-12-06T17:08:53.578Z,
    originalMaxAge: 86400000,
    httpOnly: true,
    secure: false,
    sameSite: 'none'
  },
  user: { email: 'qfath001@odu.edu', isAdmin: 0 }
}
Profile Response: {
  firstName: 'Test',
  lastName: 'User',
  email: 'qfath001@odu.edu',
  department: 'CS',
  degree: 'MS',
  uin: '01275912'
}
  ✓ should fetch user profile details (932ms)

5 passing (3s)
```

## 2. Milestone Accomplishments

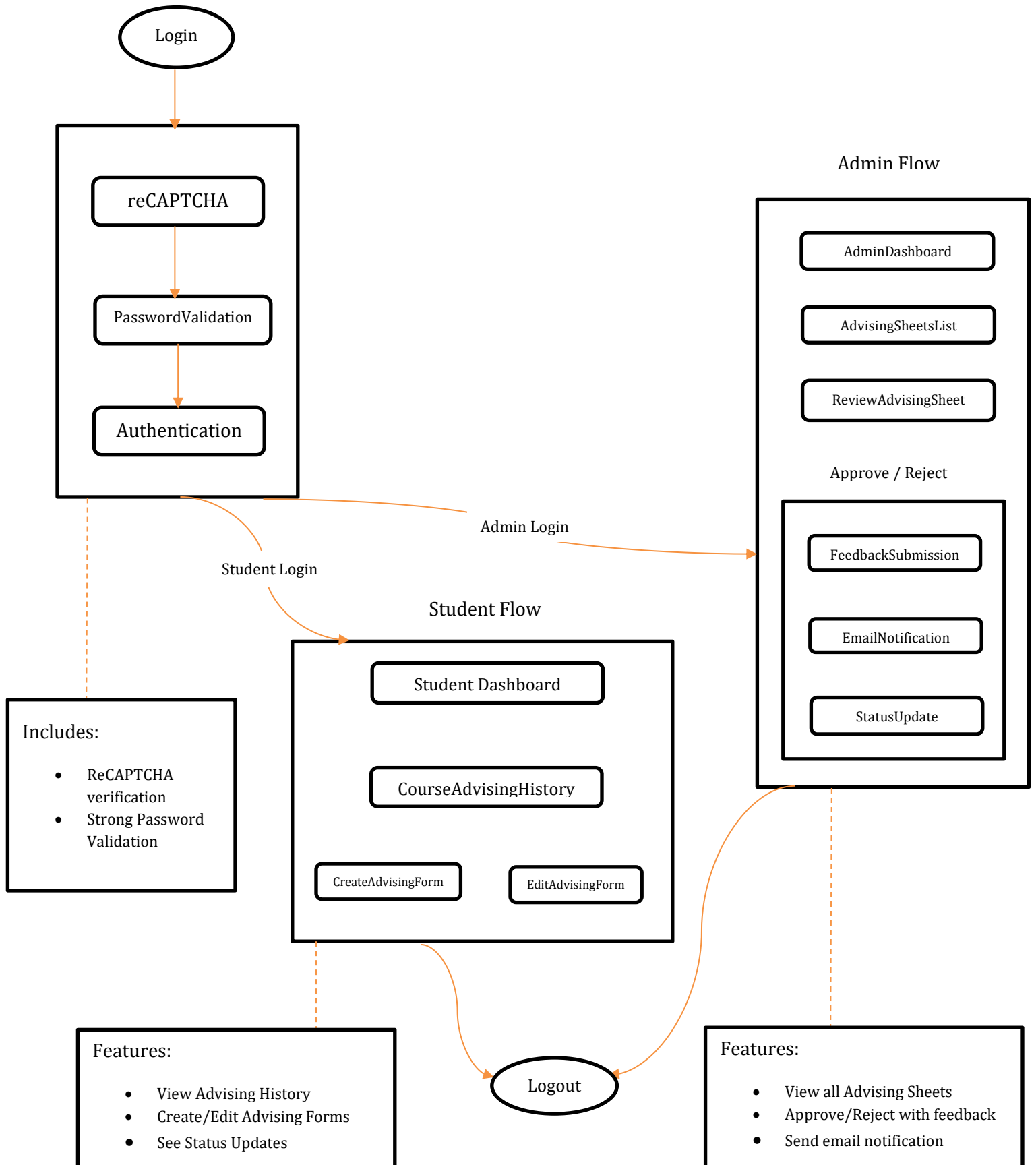
The table below summarizes the accomplishment of all milestone 3 specifications.

Fulfilled	Feature#	Specification
Yes	1	Develop a screen to display advising sheets submitted by CS department students.
Yes	2	Clicking on a student's name will redirect to a page displaying the student-submitted record. On this page, there will be options to approve or reject the record. When the admin submits their decision, they must also provide a text message with their feedback on the advising sheet. After clicking the submit button, the system will redirect to the advising sheet form (as described in point 1), and the new status

		of the student's record will be updated accordingly
Yes	3	Implement status update of student records upon submission of approval or rejection
Yes	4	Upon submission, student will receive an email where they can see the status and message provided by admin
Yes	5	Now student will be able to see the updated status of their advising sheet on Course Advising History form
Yes	6	Add reCAPTCHA to the login page. Verify the reCAPTCHA before login. Once the reCAPTCHA is verified then user can logged in into the system
Yes	7	Prevent your application from clickjacking attack. Implement the prevention of click jacking. Show the clickjacking prevention by using <iframe> in .html form
Yes	8	Add a favicon to the website
Yes	9	Add a password rule requiring a mix of capital letters, lowercase letters, special characters, and numbers (implement regex for all password fields in the application). The password length should be at least 8 characters.
Yes	10	Create test cases and execute in your BE application (Create at least 3 test cases



### 3. Architecture



## Flow of architecture

### 1. Login Flow:

- Users (both students and admins) start by accessing the login page.
- The system verifies the **reCAPTCHA** to ensure that the login request is not automated.
- Upon successful reCAPTCHA verification, user credentials are authenticated against the database.
- If authentication succeeds:
  - Students are directed to the **Student Dashboard**.
  - Admins are directed to the **Admin Flow** for managing advising records.

### 2. Student Flow:

- **Student Dashboard:**
  - Features available:
    - **Course Advising History:** Displays previously submitted advising records, including their status (Pending, Approved, or Rejected).
    - **Create/Edit Advising Forms:**
      - Students can create new advising forms or edit pending ones.
      - Each form includes sections for History, Prerequisites, and Course Plan.
    - **Status Updates:** Students can view the updated status of their advising records.
- **Logout Option:**
  - Allows students to securely exit the system, clearing session data.

### 3. Admin Flow:

- **Admin Dashboard:**
  - Displays all advising sheets submitted by students, including key details (e.g., Student Name, Term, Status).
  - Clicking on a record directs the admin to the **Approve/Reject Flow**.

- **Approve/Reject Flow:**
  - Admins can review student submissions in detail.
  - Features available:
    - Approve or reject the record.
    - Provide feedback or comments for the student.
    - Update the record's status in the database.
    - Trigger an email notification to the student, informing them of the updated status.

#### 4. Security and Features:

- **Implemented Security Enhancements:**
  - **reCAPTCHA Verification:** To prevent bot attacks during login.
  - **Strong Password Validation:** Ensures user passwords meet complexity requirements (uppercase, lowercase, special characters, numbers).
- **Session Management:**
  - User sessions are securely managed using express-session with sessions stored in MySQL.
- **Email Notifications:**
  - Admin feedback and status updates are sent to students via email.

#### 5. Key Functionalities:

- **Student Functionalities:**
  - View advising history.
  - Create or edit advising forms.
  - View status updates for advising records.
- **Admin Functionalities:**
  - View all submitted advising sheets.
  - Approve or reject forms with detailed feedback.
  - Send email notifications for status updates.

## 6. Logout Flow:

- Both students and admins can securely log out, ending their session and returning to the login page.

## 4. Implementation

### 1. Develop a screen to display advising sheets submitted by CS department students

- **File(s):** AdminAdvisingSheets.js, adminRoutes.js
  - **Frontend:**
    - AdminAdvisingSheets.js:
      - Displays a table with columns: Student Name, UIN, Term, and Status.
      - Fetches advising sheets from the backend using Axios GET requests.
      - Links each student's name to their detailed advising sheet form.
  - **Backend:**
    - Route: /admin/advising-sheets (GET) in adminRoutes.js.
      - Fetches advising sheets from the advising\_history table and joins with users table for student details.
- 

### 2. Redirect to a page for approving or rejecting a student's advising sheet

- **File(s):** AdminAdvisingForm.js, adminRoutes.js
- **Frontend:**
  - AdminAdvisingForm.js:
    - Displays detailed student advising data including history, prerequisites, and course plans.
    - Provides dropdowns to approve/reject and a text area for admin feedback.
    - Submits decisions via Axios PUT requests.
- **Backend:**
  - Route: /admin/advising-sheet/:id (GET) in adminRoutes.js.

- Fetches a specific advising record by ID.
  - Route: /admin/advising-sheet/:id (PUT) in adminRoutes.js.
    - Updates the status of the advising sheet and stores the admin's feedback.
    - Sends an email notification to the student using Nodemailer.
- 

### 3. Implement status update of student records

- **File(s):** AdminAdvisingForm.js, adminRoutes.js
  - **Frontend:**
    - Updates the status dynamically after admin submission.
    - Redirects back to the advising sheet list.
  - **Backend:**
    - Updates the status field in the advising\_history table for the given record ID.
- 

### 4. Email notification for status and feedback

- **File(s):** adminRoutes.js
  - **Backend:**
    - Uses Nodemailer to send an email to the student upon status update.
    - Email includes the updated status and admin feedback.
- 

### 5. Display updated advising sheet status on Course Advising History

- **File(s):** CourseAdvisingHistory.js, advisingRoutes.js
- **Frontend:**
  - Fetches advising history using Axios GET requests.
  - Displays updated statuses dynamically, with links for pending records to edit.
- **Backend:**

- Route: /student/advising-history (GET) in advisingRoutes.js.
    - Retrieves updated advising history for the logged-in student.
- 

## 6. Add reCAPTCHA to the login page

- **File(s):** index.html, server.js
  - **Frontend:**
    - Integrated Google reCAPTCHA script in index.html.
    - Sends the reCAPTCHA token during login requests.
  - **Backend:**
    - Verifies reCAPTCHA token in the /login route in server.js.
- 

## 7. Implement clickjacking prevention

- **File(s):** server.js, clickjacking-test.html
  - **Backend:**
    - Added headers in middleware to set X-Frame-Options: DENY and Content-Security-Policy: frame-ancestors 'none'.
  - **Testing:**
    - clickjacking-test.html:
      - Contains an <iframe> test to verify clickjacking protection.
- 

## 8. Add a favicon to the website

- **File(s):** index.html
  - **Frontend:**
    - Added a <link> tag to include the favicon in the header.
- 

## 9. Add a password rule

- **File(s):** server.js
  - **Backend:**
    - Validates passwords using a regular expression that enforces:
      - Minimum 8 characters.
      - At least one uppercase letter, one lowercase letter, one number, and one special character.
- 

## 10. Create and execute test cases

- **File(s):** user.test.js
  - **Testing Framework:**
    - **Tools:** Mocha, Chai, Supertest.
    - Implemented test cases for:
      - User signup and login.
      - OTP verification for signup and login.
      - Profile fetching and advising form submission.
- 

## Deployment

- **Frontend:**
  - Deployed on Firebase Hosting for secure and fast content delivery.
- **Backend:**
  - Deployed on Render with MySQL database connections configured.
- **Testing:**
  - Tests executed locally using npm test and verified against the local server.