Web Platform Development 2

**Coursework 2 - CRUD Website**

Gavin Mackle & Ben Maxwell

We declare that all work submitted for this coursework is the work of Ben Maxwell and Gavin Mackle alone unless stated otherwise.

horizontal line

# Introduction

Our project is a coursework management website that uses CRUD functionality along with a user login and register system to provide students with a software tool that can help them organise their coursework.

# Link Design

The Site structure can be found in Figure 1 showing how a user would navigate the website. The thought process behind designing the website was about simplicity and putting access to functionality in places that make sense for the user which can be further seen when fully analysing the URL’s implemented into the application.

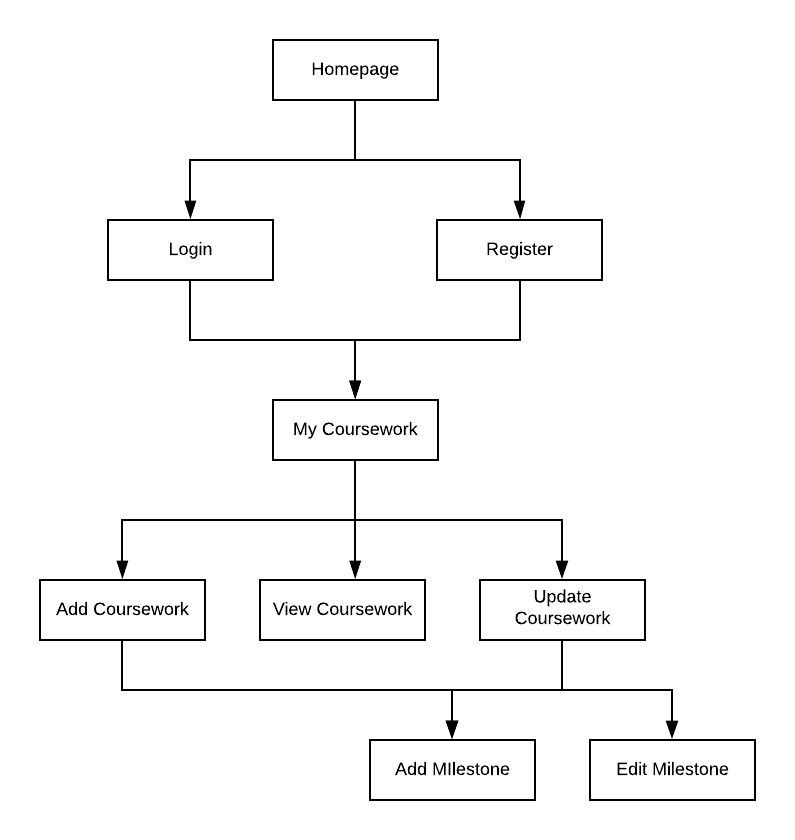


Figure 1 - Website Structure

**URL’s**

When designing the URL’s we were aiming to keep it simple and allow room for expansion or modification of the website upon future development. All URLs discussed can be reached on the website by adding them to the base URL. The base URL is:

scheme://domain:port/

The base URL simply takes the user to the homepage where they can access functionality such as logging in, Registering, logging out and accessing their coursework via the navigation bar. Every page displays the navigation allowing users to navigate easily.

Register - register/

As the name implies, this URL takes the User to a register page where they are presented with a form prompting them to sign up. The form includes error checking to prevent User’s signing up with bad credentials such as missing fields or already taking usernames. Once a user has successfully registered they are redirected to the login URL showing the login page.

Login - login/1

When a user first comes onto the website they can access this URL through the navigation bar or through successfully registering with the website. On the page presented when navigating to this URL is a form allowing a user to enter their login credentials to authorise them allowing them to view their coursework and shared courseworks as well. Upon a successful login, the user is redirected back to the home page.

Login - login/fail

If a user has an unsuccessful login attempt whether that’s due to bad credentials or a missing field they will be redirected back to the login page where an error message will appear notifying the user of the unsuccessful login attempt. Apart from this it functions the exact same as the regular login URL.

Login - login/logout

When a user is logged in, they access this URL via the navigation bar which will log them out of the website killing their current authorisation session and redirecting them back to the base URL. If a non authorised user were to access this URL by typing it in, they would be presented with a 401 page.

Authorised URLs

From this point onwards, a user must be at the minimum authorised to access the URL. Some of the URL’s like those for modification of coursework and milestones will also check to make sure the user trying to access the URL is also the author of the resource stored in the database. This will be discussed in each URL individually.

My Coursework - coursework/

When logged in a User can navigate to the coursework URL to see a general overview of all their courseworks. From here, a user can choose to view a specific coursework, update, delete or create a new coursework.

Create Coursework - coursework/create

From the coursework overview page a User can select the option to create a new coursework where they will be taking to this URL where they will be shown a form allowing them to create a new coursework. There is error checking that does not allow a user to enter empty fields apart from the completion date of the coursework. Upon submitting, what the user has entered is inserted into the database and they are redirected to the add milestone URL.

Add Milestone - coursework/addmilestone/id

This URL is accessed either through the creation of a new coursework or the modification of a present coursework. From here the user can add a new milestone to a coursework which is identified as the id of the selected coursework is passed in as a parameter. When on this page the user can also see all the milestones already added to the coursework. When the User is satisfied with the milestone(s) added, they can either navigate to the specific view of this coursework showing all details, the generic overview of all courseworks or the modify coursework page.

View Coursework - coursework/view/id

From the coursework overview page a user can select the option of viewing coursework which will take them to the page of where they can see detailed information about the coursework including milestones and due date. The coursework ID is passed in as part of the url which allows the backend to retrieve all the relevant data regarding the coursework that is stored in the database to present to the user. From here, the author of the coursework can also set the coursework to be viewable by others or to make it private. If made public, any authorised user can view the coursework and it’s details via the link.

Modify Coursework - coursework/modify/id

If a user chooses to update a coursework, they are taking to the modify URL which includes the coursework id so the backend can use it to update the coursework when the user has submitted their changes. The layout is the same as the create coursework page so users will be familiar with the page when they navigate here. To access this page the user must not only be authorised but also be the one who created the coursework. While on this page the user has the chance to add new milestones, modify existing milestones or delete them.

Modify Milestone - coursework/modify/modifymilestone/id

While on the modify milestone page, Users can change the name of a milestone and it’s current status of completion. Again the millstone ID is included in the URL for the backend to be able to update the appropriate record in the database. Upon a user submission with appropriate input they are redirected to the modify coursework page. Like the Modify Coursework page, The user must be authorised and be the author of the milestone to modify it otherwise they will be redirected to a 401 not authorised view.

Delete Milestone - coursework/modify/deletemile/id

From the modify coursework page, Users can select to delete a milestone which sends a request to the back end of the website with the milestone id as part of the url which the back end uses to delete the milestone. Upon completion of this the user’s page is refreshed to show the updated list of milestones.

Share Coursework - coursework/share/id

URL which allows the front of the application to send a request to the backend to make a specific coursework, as designated by the coursework id in the URL, public for other users to view as long as they logged in when attempting to do so. Users can access this functionality by making use of the appropriate button present on the view coursework page which can only be seen and used by the author of the coursework.

Make Coursework Private - coursework/deshare/id

URL which allows Users who’s coursework is currently public to make the coursework private so only they can view it. This works by javascript on the front end sending a request to the backend and including the coursework ID as part of the URL so the appropriate coursework is made private again. This functionality can only be used by the author of the coursework.

Delete Coursework - coursework/delete/id

From the general overview of all user courseworks, a User can select to delete a coursework if they desire. When selected a request is sent to the server containing the coursework ID as part of the URL, again allowing the backend to delete the appropriate coursework and all it’s milestones.

# 

# Persistence

The backend of the website connects to a nedb database for storing all the information regarding coursework,milestones and users. Each entity (User, Coursework, Milestone) has their own database file as it doesn’t make sense to pair them together all in one file since separating them improves readability and reduces coupling. Each row in the coursework store has an attribute called “Author” which is the username of the user that submitted the coursework which is also their unique identifier. Each Milestone also has a “courseworkId” attribute tying it to a specific coursework as well as its own ID. These relationships between Entities are very similar to that of a Foreign-Key relationship found in traditional RDMS such as MySQL and allow us to differentiate each row of data as required.

It is important to note that the database is configured not to be in memory meaning changes in data will not be undone if the website is restarted. However, during development there were several issues that kept reappearing even after being fixed several times which ultimately caused us to run the website in a docker container to provide stability which ended up working perfectly. With the website running in a docker container, the database essentially becomes in memory as the database then exists purely in the container which when stopped, is entirely destroyed. As seen in the GitHub, the website can be run using nodejs without docker but we would not feel comfortable deploying it like that.

Each entity has its own class that is responsible for handling the database functionality which can all be found in the model directory located within the routes directory. These classes contain functions which handle different functionality such as insert,find, update and delete for their respective entities. This allows there to be less code duplication within the application as the necessary functions from all of these classes can be called from anywhere in the application as required.

# Testing

## System Testing

For system testing, our team looked over the website first to check for styling issues and functionality errors and once we had determined that the website was ready for use we then tested the project hosted inside of a docker container on an azure VM to gather test data regarding the usability and functionality of the website. The following data was gathered by both the external users and the project team:

|  |  |  |  |
| --- | --- | --- | --- |
| Page | User Scenario | Acceptance Details | Yes/No |
| Login | User should try logging in using a valid login otherwise they will be redirected to the login again | No details entered and login button pressed, login fails | Yes |
|  |  | Incorrect Details entered, login fails | Yes |
| Evidence in Figure 2 - Login Page (Error) |  | Correct details, Username: SenorBeep, Password: beep , entered, user is logged in | Yes |

|  |  |  |  |
| --- | --- | --- | --- |
| Page | User Scenario | Acceptance Details | Yes/No |
| Register | User should try registering using valid details otherwise an error message appears | No details entered and register button pressed, register fails | Yes |
|  |  | Username and Password entered but not confirm password, register fails | Yes |
|  |  | Password and Confirm Password entered but not username, register fails | Yes |
| Evidence in Figure 3 - Register Page (Error) |  | Username, Password and Confirm password with matching password entered, user is registered | Yes |

|  |  |  |  |
| --- | --- | --- | --- |
| Page | User Scenario | Acceptance Details | Yes/No |
| My Coursework | User should be able to view coursework that they have added to their account and update, delete, add or view the coursework | If a coursework has been added, the user should be able to view it | Yes |
|  |  | User should press the delete button on a saved coursework, coursework should be deleted | Yes |
|  |  | User should press the update button on a saved coursework, the user should be redirected to the modify coursework page | Yes |
| Evidence in Figure 4 - My Coursework page |  | User should press the view button on a saved coursework, the user should be redirected to the view coursework page | Yes |
|  |  | User should press the create coursework button, user should be redirected to the add coursework page | Yes |

|  |  |  |  |
| --- | --- | --- | --- |
| Page | User Scenario | Acceptance Details | Yes/No |
| Add Coursework | User should be able to add new coursework to their account | User doesn’t enter any data and presses add, error messages appears | Yes |
|  |  | User enters title and module but not due date, error message appears | Yes |
| Evidence in Figure 5 - Create Coursework Page (Error) |  | User enters data in every field other than completion date, user is redirected to add milestones page | Yes |

|  |  |  |  |
| --- | --- | --- | --- |
| Page | User Scenario | Acceptance Details | Yes/No |
| Modify Coursework | User should be able to edit the details of the coursework | User removes due date of coursework, error message displays | Yes |
|  |  | User removes name of module, error message displays | Yes |
|  |  | User removes coursework title, error message displays | Yes |
| Evidence in Figure 6 - Update Coursework Page (Error) |  | User adds completion date, user redirected to view page | Yes |
|  |  | User clicks on add additonal milestones, user is redirected to add milestones | Yes |

|  |  |  |  |
| --- | --- | --- | --- |
| Page | User Scenario | Acceptance Details | Yes/No |
| View Coursework | User should be able to view saved coursework and create a share link | User presses share button, share link displayed and other users can now view the coursework | Yes |
| Evidence in Figure 7 - View Coursework Page (Share Link) & Figure 8 - View Coursework Page (Private) |  | User presses make private button, link will no longer be displayed | Yes |

|  |  |  |  |
| --- | --- | --- | --- |
| Page | User Scenario | Acceptance Details | Yes/No |
| Modify Milestone  Evidence in Figure 9 - Modifiy Milestone Page | User should be able to modify the milestone for the chosen coursework | User changes the status of the milestone using the dropdown list, successfully updated and redirected to modify coursework page | Yes |

## Website Screenshots (System Testing)

### Login Page (Error)

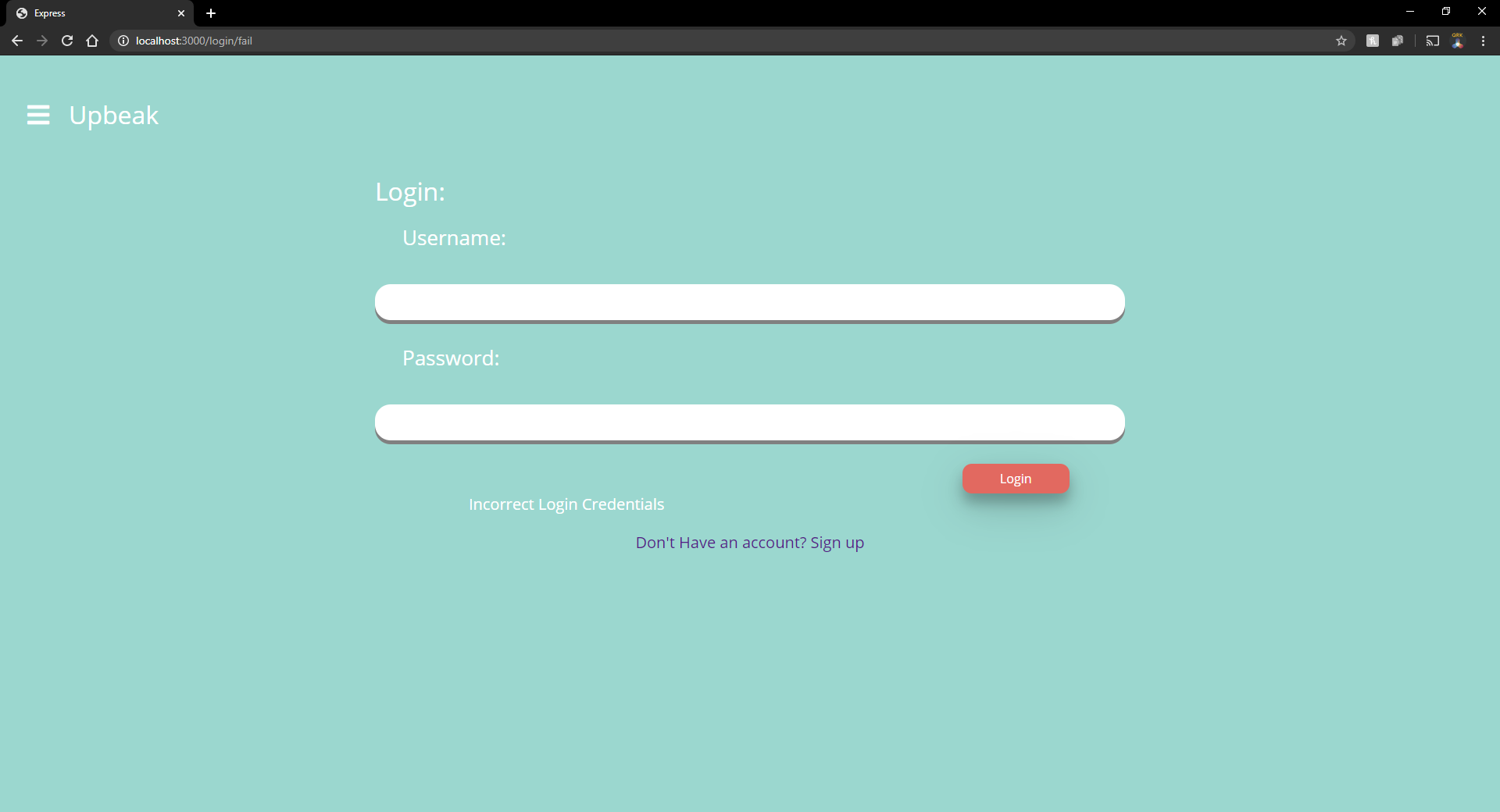


Figure 2 - Login Page (Error)

### 

### Register Page (Error)

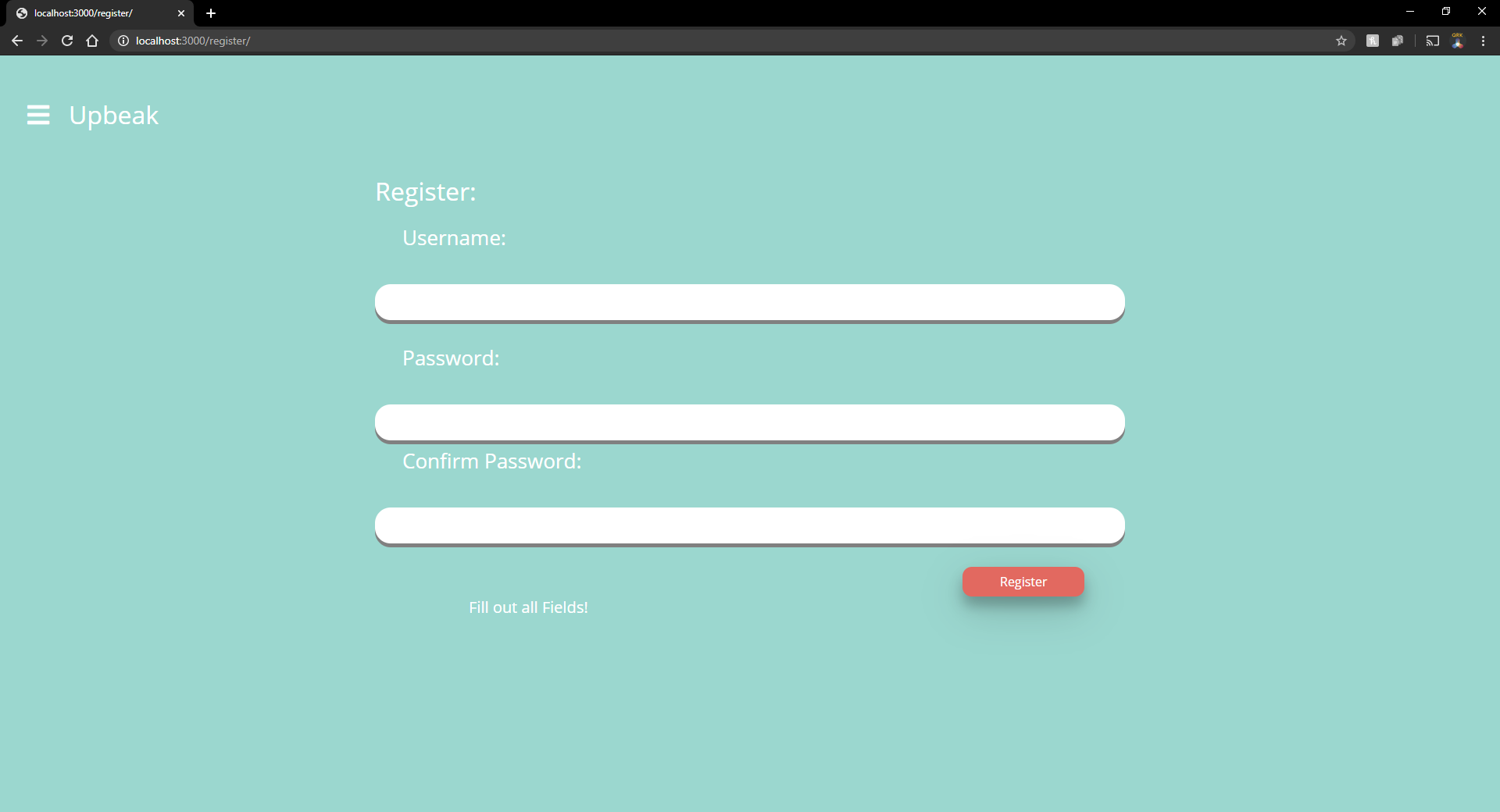


Figure 3 - Register Page (Error)

### 

### My Coursework Page

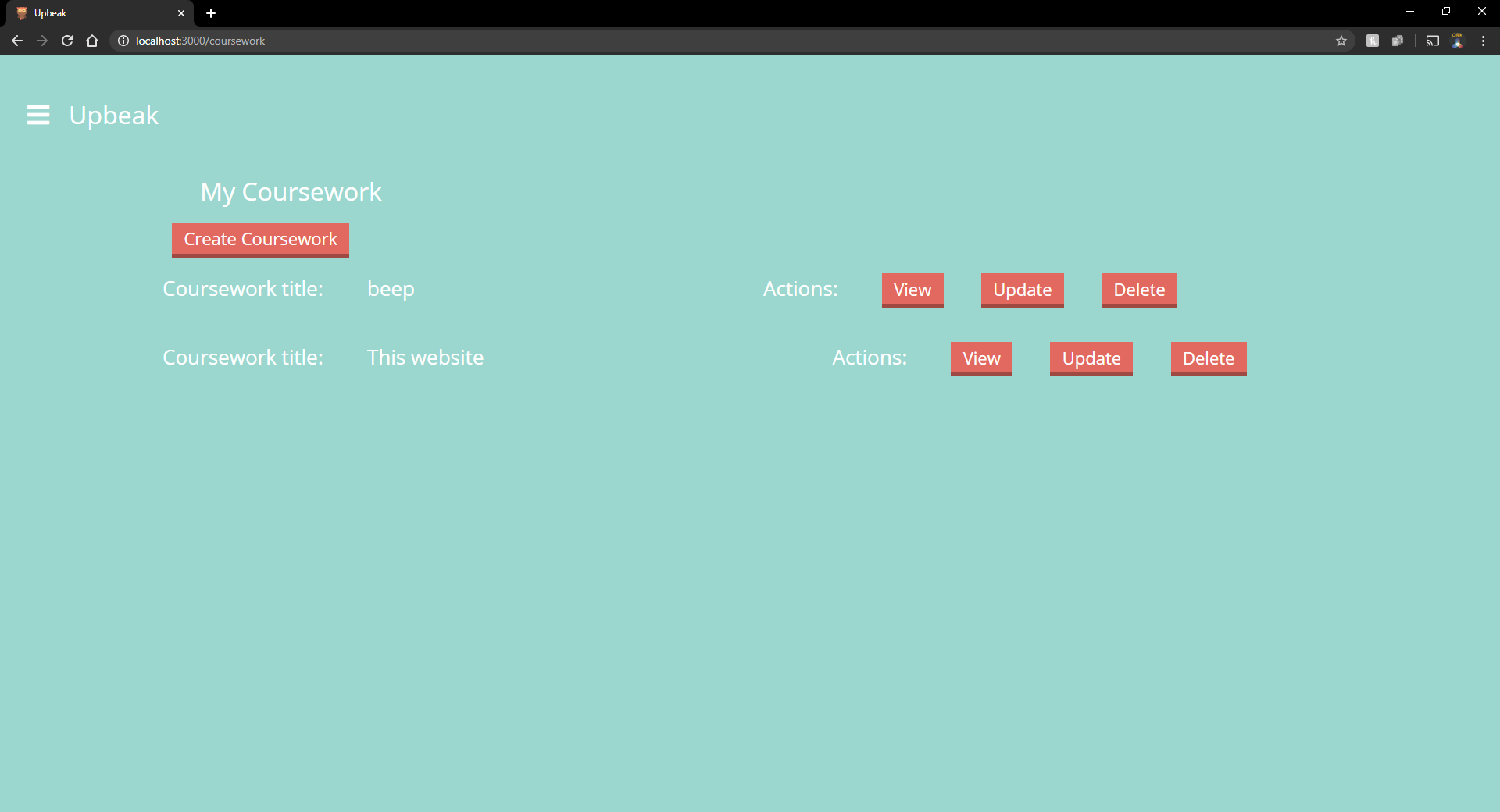


Figure 4 - My Coursework Page

### Create Coursework Page (Error)

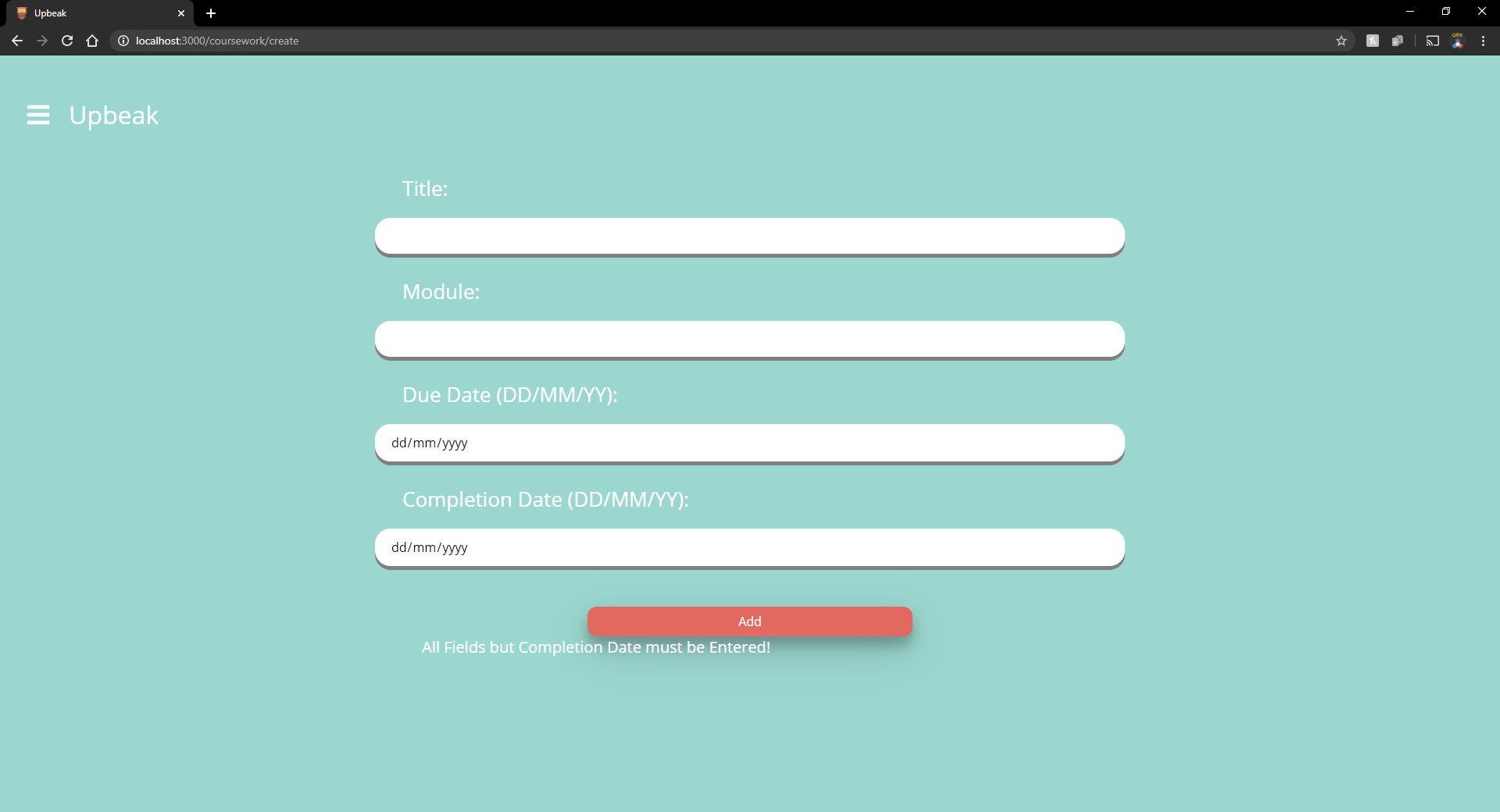


Figure 5 - Create Coursework Page (Error)

### Update Coursework Page (Error)

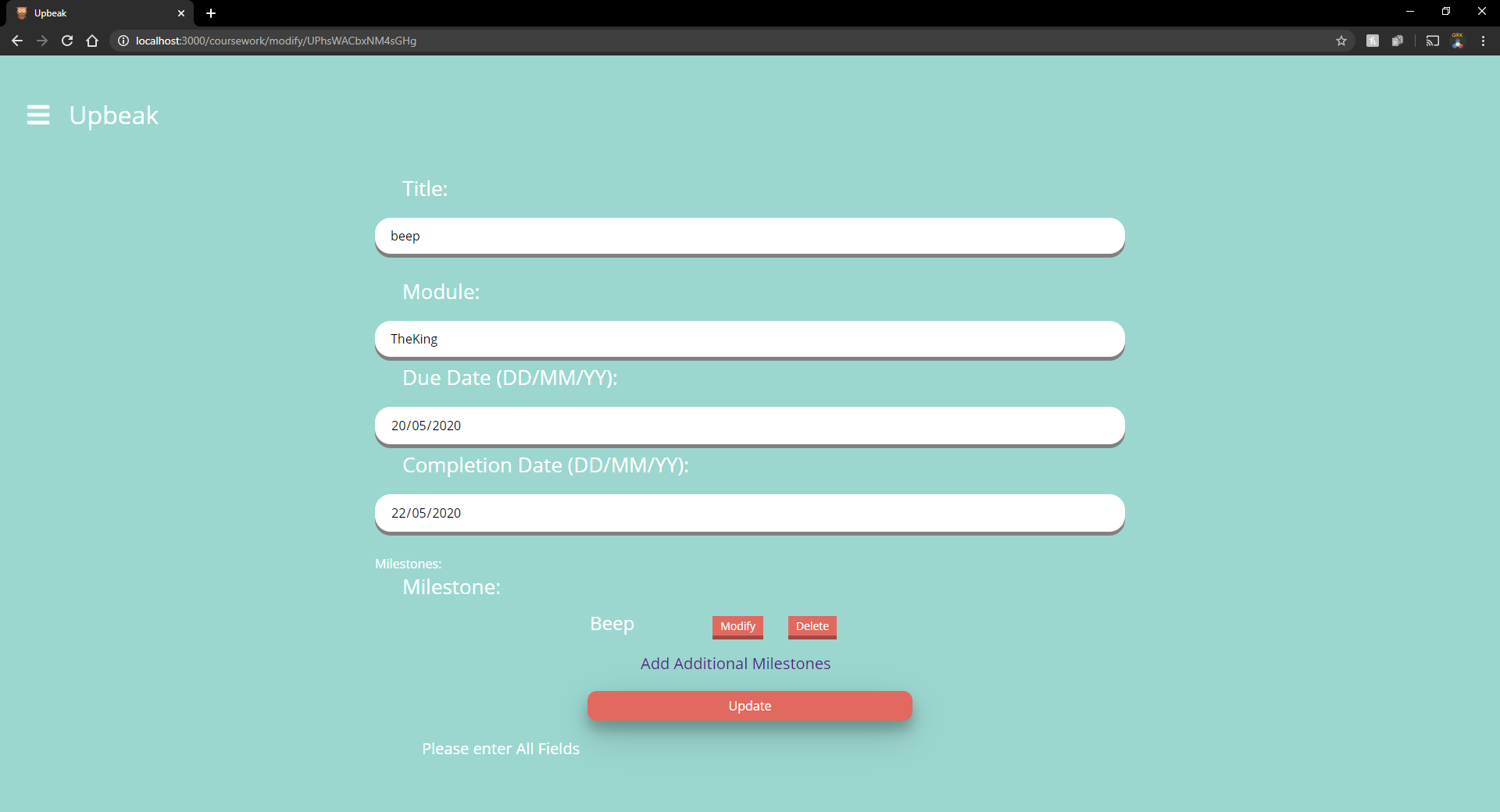


Figure 6 - Update Coursework Page (Error)

### View Coursework Page (Share Link)

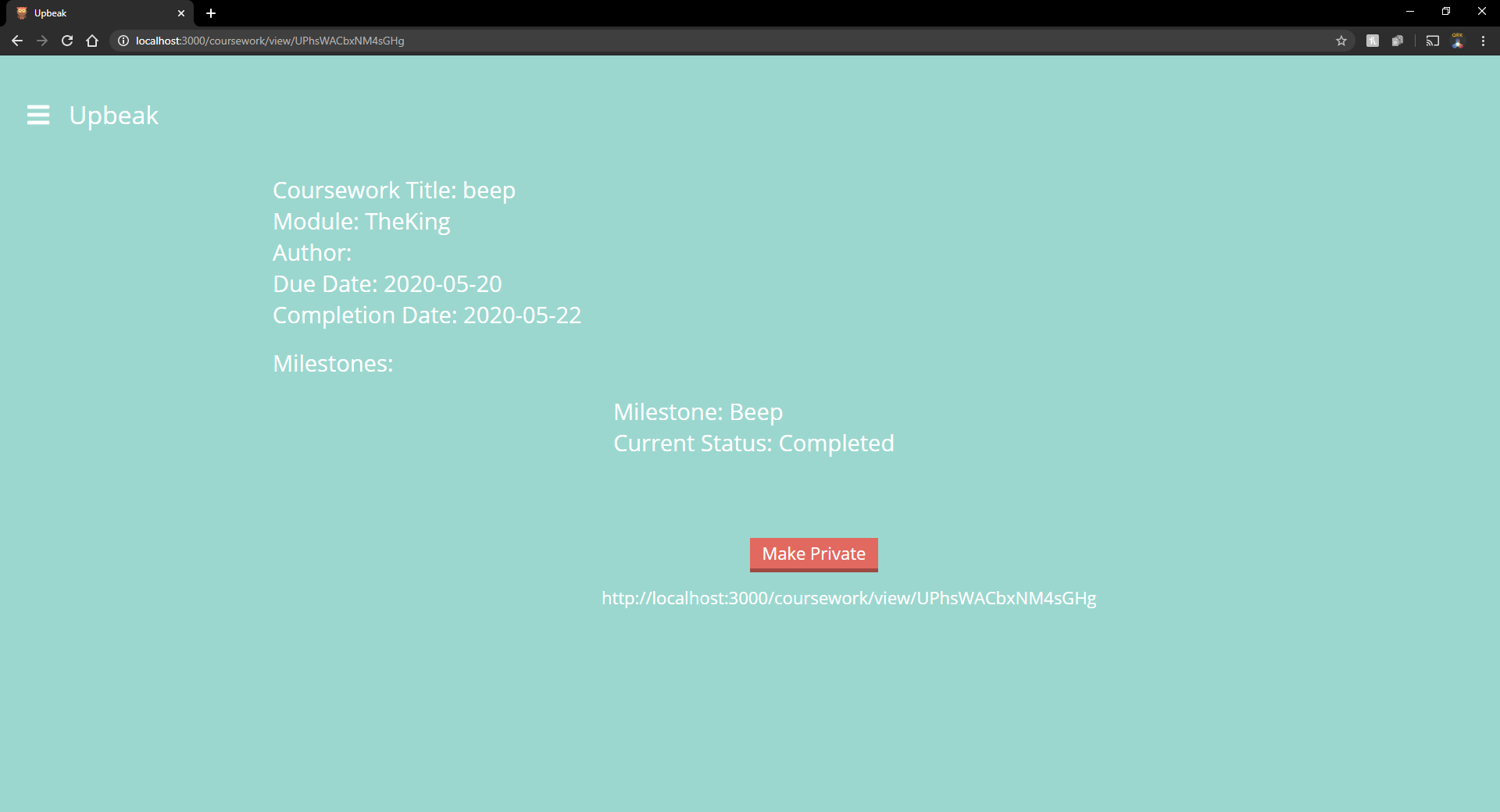


Figure 7 - View Coursework Page (Share Link)

### View Coursework Page (Private)

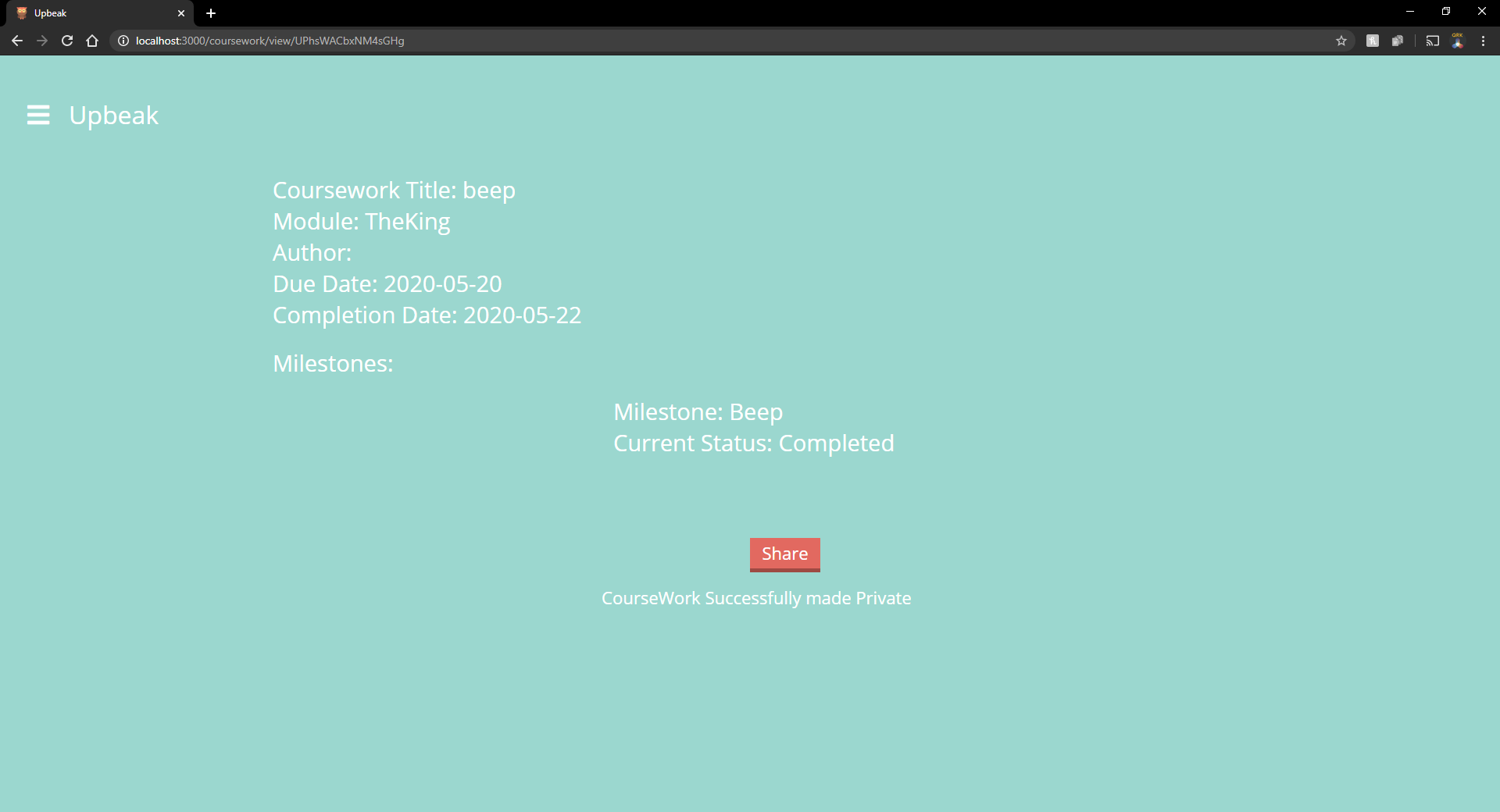


Figure 8 - View Coursework Page (Private)

### Modify Milestone Page

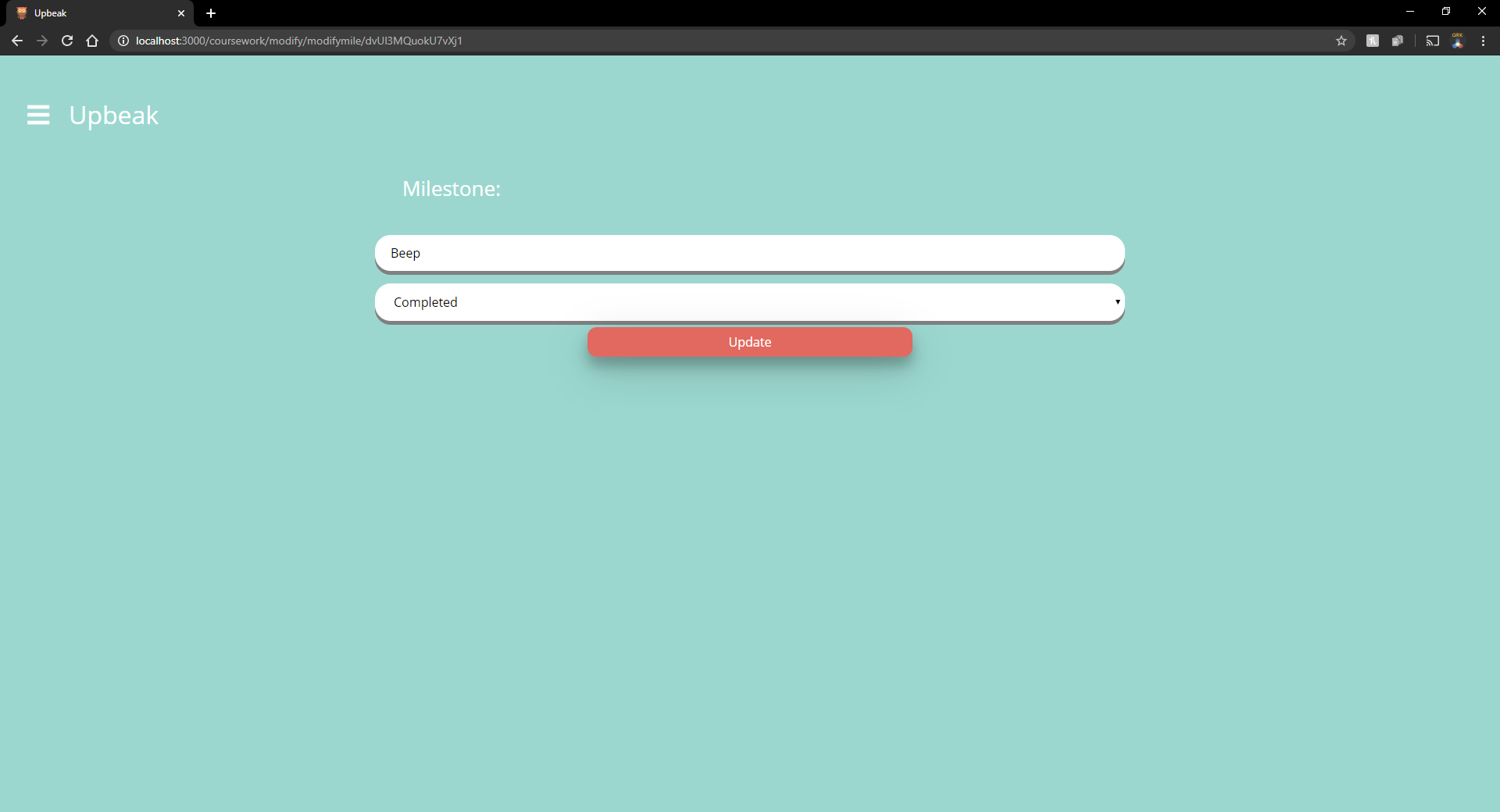


Figure 9 - Modify Milestone Page

## Unit Testing

As well as System Testing, the key database functionality is tested using tests from inserting users and finding them to modifying and deleting coursework. The advantage of Unit Tests is that they are much quicker to run than a person and if they have good code coverage and are run regularly it helps to ensure a higher quality application. The results from Unit Testing were as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Test Number | Functionality Tested | Acceptance Details | Pass/Fail |
| 1 | User Insert | Insert a new User called “UnitTest1” into the database and then retrieve them successfully | Pass |
| 2 | User Insert | Insert a new User called “UnitTest2” into the database and then retrieve them successfully | Pass |
| 3 | Coursework and Milestone | Create a Coursework for “UnitTest1” and add a milestone to this coursework. Insert them into the database. Test passes upon successful retrieval from the database. | Pass |
| 4 | Coursework and Milestone | Create a Coursework for “UnitTest2” and add a milestone to this coursework. Insert them into the database. Test passes upon successful retrieval from the database. | Pass |
| 5 | Coursework and Milestone | Modify the coursework created previously belonging to “UnitTest1” as well as the milestone added and then attempt to update this change in the database. Upon successful retrieval of a modified coursework and it’s milestone from the database is the test passed. | Pass |
| 6 | Coursework and Milestone | Modify the coursework created previously belonging to “UnitTest2” as well as the milestone added and then attempt to update this change in the database. Upon successful retrieval of a modified coursework and it’s milestone from the database is the test passed. | Pass |
| 7 | Coursework and Milestone | Delete the milestone and coursework created in the tests for “UnitTest1” from the database. If the coursework can’t be found after this using it’s ID the test passes. | Pass |
| 8 | Coursework and Milestone | Delete the milestone and coursework created in the tests for “UnitTest2” from the database. If the coursework can’t be found after this using it’s ID the test passes. | Pass |

## Running the Application

There are two ways to run the program, firstly the program can be ran using normal node commands. To run it this way simply run:

* **Npm install**
* **Node bin/www.js**

Secondly the program can be ran using Docker which is also the recommended way to do so. To achieve this run either of the following set of commands:

* **docker pull bean12345/wpd2**
* **docker run -p 3000:3000 bean12345/wpd2**

Or from the root of the project:

* **docker build -t <enterTagName> .**
* **docker run -p 3000:3000 <enterTagName>**

Upon request the application can be ran at the following URL <http://40.76.207.22:3000/> which is run in a docker container on an azure VM. Due to cost limitations we can’t leave it running but are happy to put it up if requested. If you wish to run the Unit Tests this can be done using – **npm run Tests**

# 

# Application Security

Overall the security is reasonable but could definitely be improved in the application and it was something that was considered during development but due to time constraints not everything could be implemented.

## User Sessions

The website makes use of User sessions to authorise Users access to resources once they have been authenticated via the login process. While any user can see their session in their browser via developer tools it does not reveal any form of crucial information that could be used as an exploit in accessing the website improperly. Furthermore if a User clears their session data while logged in, if they attempt to do something that requires Authentication which is a requirement for the vast majority of the website they will be redirected to a 401 Unauthorized view. Also the cookies used are httpOnly reducing the chance of a cross site scripting attack so overall the User sessions are quite secure and could maybe only benefit from switching to something like a Secure cookie and enabling HTTPS.

## User Input

As a whole user input is not cleansed. While there are checks to ensure users are entering characters into fields their input is trusted. While this is something the group totally disagrees with due to time constraints this could not be rectified. This does mean that user’s could maliciously enter javascript into an input field for a coursework which when shared and opened by other users could execute without them knowing as the script would not be visible on the page. To mitigate this, a node module such as sanitize could be used to make sure the user input is safe to enter in the database and then display on pages.

## Password

Passwords in the application are hashed before storing them in the database, which in the event of a data breach would make it harder for people to access user accounts illegitimately as the hash would need to be cracked first. Storing passwords in the database in plain text was never a consideration for the group as not only is it bad practice in the event of a data breach that occurred maliciously, the attacker gets more information that they can do a lot more with if the passwords are in plain text. To make the passwords even more secure, a salt could be applied after hashing which introduces randomness on top of the hash making the protection of hiding the password even harder to crack, which would be the next step going forward.