

Web Technologies – Coursework 2 Resit

Thomas Diggins – 40203272

INTRODUCTION

The aim of this assignment is to create a website that allows a user to register for an account, sign in to said account, then send an encoded message to another user. The user can also decode and read messages sent to them from other users. The website is created using Hypertext Markup Language (HTML), uses JavaScript to provide the encoding methods and database connection, and Cascading Style Sheets (CSS) to implement style attributes.

This report provides a concise explanation of the steps taken to achieve completion of the assignment.

*Bold text in brackets refers to a specific reference, listed on the last page, e.g. **(Simon Wells)***

SOFTWARE DESIGN

The basic plan for the website design was to implement controls that would allow the user to create an account using a form, then sign in to the account from another form and send an encoded message to other users.

Layout for register.html

ENCODER	ABOUT	OTHERS	REGISTER	SIGN IN
---------	-------	--------	----------	---------

REGISTER

Short statement about creating an account

Username

Password

Confirm Password

Short statement about Terms and Privacy

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

Layout for signin.html

ENCODER	ABOUT	OTHERS	REGISTER	SIGN IN
---------	-------	--------	----------	---------

SIGN IN

Sign in to your account.

Username

Password

Don't have an account? [Register](#)

Layout for courier.html

ENCODER	ABOUT	OTHERS	REGISTER	SIGN IN
---------	-------	--------	----------	---------

COURIER

Short explanation of how to use the messenger

Username

Message

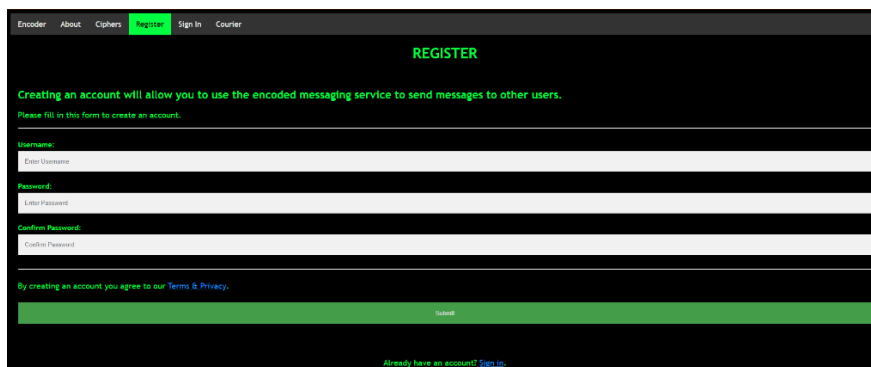
IMPLEMENTATION

The website has six implemented html files titled index.html, about.html, ciphers.html, register.html, signin.html and courier.html, each corresponding to a web page. The service chosen in testing to host the website was node.js, loaded from files on a local node directory. (index.html, about.html and ciphers.html have been excluded as they were discussed in the previous assignment)

Modules used in the program include:

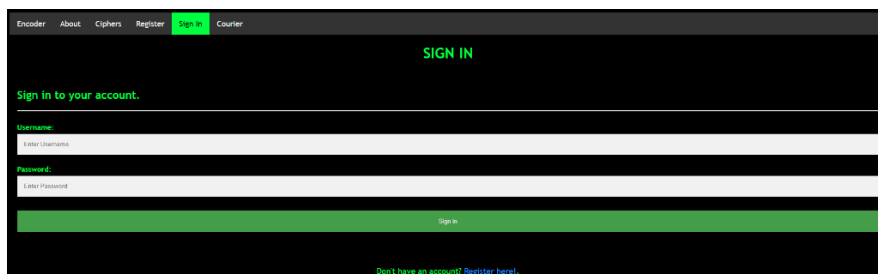
- express – Used to allow routes and requests to be made, helping to manage the program.
- sqlite3 – Used to construct and access the database containing user information.
- jquery – Used to provide methods that correspond to HTML events.
- body-parser – This allows the JavaScript to take information being POSTed from a html form.

register.html



The screenshot shows the 'REGISTER' page of a web application. At the top, there is a navigation bar with links: Encoder, About, Ciphers, Register (highlighted in green), Sign In, and Courier. Below the navigation bar, the title 'REGISTER' is displayed in green. A message states: 'Creating an account will allow you to use the encoded messaging service to send messages to other users. Please fill in this form to create an account.' The form consists of three input fields: 'Username' (with placeholder text 'Enter Username'), 'Password' (with placeholder text 'Enter Password'), and 'Confirm Password' (with placeholder text 'Confirm Password'). Below the form, there is a green button labeled 'Submit'. At the bottom, a link says 'Already have an account? Sign In.'

signin.html



The screenshot shows the 'SIGN IN' page of a web application. At the top, there is a navigation bar with links: Encoder, About, Ciphers, Register, Sign In (highlighted in green), and Courier. Below the navigation bar, the title 'SIGN IN' is displayed in green. A message states: 'Sign in to your account.' The form consists of two input fields: 'Username' (with placeholder text 'Enter Username') and 'Password' (with placeholder text 'Enter Password'). Below the form, there is a green button labeled 'Sign In'. At the bottom, a link says 'Don't have an account? Register here.'

courier.html



The screenshot shows the 'COURIER' page of a web application. At the top, there is a navigation bar with links: Encoder, About, Ciphers, Register, Sign In, and Courier (highlighted in green). Below the navigation bar, the title 'COURIER' is displayed in green. A message states: 'To use the messenger, enter the username of the user you wish to send the message to, then press "Send via..." to choose your encoding method.' The form consists of two input fields: 'Username' (with placeholder text 'Enter Username') and 'Message' (with placeholder text 'Enter Message'). Below the form, there are three buttons: 'Send via ROT13', 'Send via Caesar', and 'Send via ROT135 Caesar'.

CRITICAL EVALUATION

This evaluation will compare the completed assignment with the requirements in the specification.

The requirements:

1. A user can sign up for an account.
2. A user can log in to their created account.
3. The user can send an encoded message to other users.
4. To view and decode messages sent by other users

Completion of assignment based on these requirements:

1. A form takes the username and password of the user and stores it in a database.
 2. The user CAN log in to their account.
 3. The user CANNOT send an encoded message to others.
 4. The user CANNOT receive and decode messages from other users.
- 4.5 The encoder and decoder part of the site is now functional.

Based on the requirements specified, the submitted assignment does NOT meet all base requirements.

Possible Improvements:

- **Completion of the base requirements..**
- **Checking to see if a username has already been used.**

PERSONAL EVALUATION

Reflecting on What I've Learned:

I've learned more about CSS, particularly about navigation bars (https://www.w3schools.com/howto/howto_js_topnav.asp) and form design (https://www.w3schools.com/howto/howto_css_register_form.asp).

As I did not find the lab notes very helpful in this assignment I ended up finding a video guide (<https://www.youtube.com/watch?v=RaNpujNOAHg>) that I found was useful in getting the basic structure and idea of my site functioning.

The Challenges I Faced:

I faced many challenges during this assignment. To start with I faced pathing errors which were resolved by adding directory paths to the 'path' system environment variable. Then I had issues in writing my code and getting it to do what I needed it to. I could not get a separate POST to check a user login against details in the database working.

I did not find the lab notes to be particularly helpful when working on this assignment. I found myself trying to get the examples in the lab notes working instead of doing work on the assignment.

I have had some prior experience in html but never had to do something quite as expansive as this assignment. I ended up finding my previous knowledge was not enough to help me on this assignment.

The Methods I Used to Overcome the Challenges:

The pathing errors were resolved by adding the necessary directory paths to the 'path' system environment variable, suggested by (Simon Wells). I was finally able to get the register form to store data in the database.

How I Feel I Performed:

I feel that I performed well enough in a restricted time period. Whereas the project is not perfect, I have much improved upon the functionality of the website. Previously, the additional JavaScript files that ran the code for the encoder and decoder on the index page of the site were being excluded by the server, and a user could not sign in to an account. Now the additional JavaScript files are included and functioning, and a user can sign into an account provided the respective username and password are present in the database.

REFERENCES

Creating a Navigation Bar:

https://www.w3schools.com/howto/howto_js_topnav.asp

Designing a Registration Form:

https://www.w3schools.com/howto/howto_css_register_form.asp

Build Your First Web Application Using HTML, CSS, Java Script, Node Js and SQLite

<https://www.youtube.com/watch?v=RaNpujNOAHg>

Adding Node to the 'path' System Environment Variable

(Simon Wells)

<https://love2dev.com/blog/node-is-not-recognized-as-an-internal-or-external-command/>