

Coursework 2 - Report

Ahad Malik

40122791@napier.ac.uk

Edinburgh Napier University - Web Technologies (SET008101)

1 Introduction

This report provides an insight into the planning and implementation of a messaging system which will be added to the cipher website that was developed as part of Coursework 1. The main objective of this coursework is to add functionality that will allow users to register themselves an account which then they can access using a login interface to access the messaging functionality. The messaging platform will allow registered users to send and receive encrypted messages to one another. The web app will include a client-side and a Server side. The client-side will provide the front end interface, what the user will interact with. This will feature the Login page, Registration, Messaging platform and the Cipher pages. The Server side will be used to persist user data and support client-side functionality.

2 Design

' Initial design of the software began with going through the requirements and determining the different components of the application. The express application generator was used to quickly create a basic application template.

```

.
├── app.js
├── bin
│   └── www
├── package.json
├── public
│   ├── images
│   ├── javascripts
│   └── stylesheets
│       └── style.css
├── routes
│   ├── index.js
│   └── users.js
└── views
    ├── error.pug
    ├── index.pug
    └── layout.pug

7 directories, 9 files

```

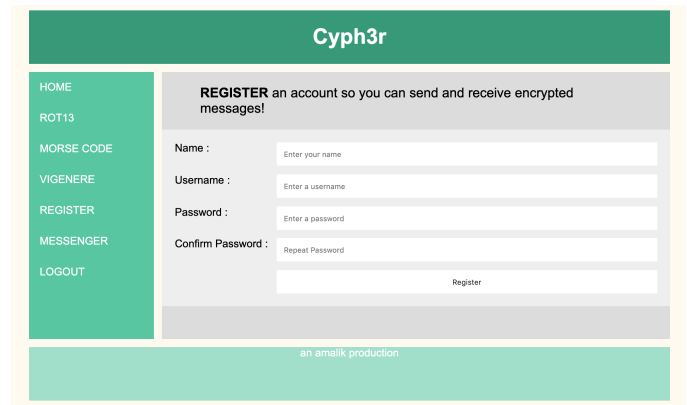
The folders structure is as follows:

- App.js : Initial start up file for whole app
- Bin : Contains the executable file that runs the app
- Public : Includes the Javascripts and stylesheets used for the apps webpages
- Routes : Router files that are used to route information
- Views : Stores .pug files for different views of the app. These are loaded dynamically using the server.

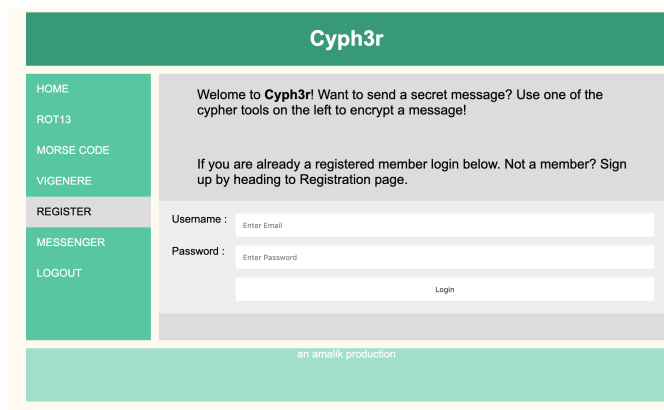
After the applications was structured. Login and Registration pages were planned and designed, initially sketched and then using wire frame diagrams.

3 Implementation

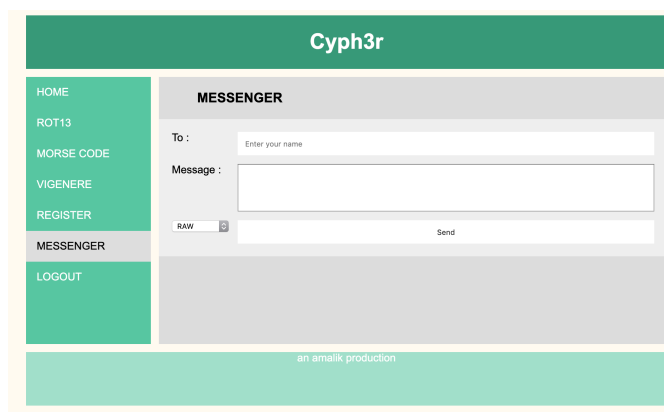
For users to be able to register an account and login it was vital to implement a way to persist their data. The local file system was used to store user data in a db folder within the app folder. Each time a user registers a new file is created within db folder with the users username as its title. Register screen below:



During the login process the users entered values are checked against the data in the database. If a file with the same name as the username exists in the db folder, the password is checked against that of the one in the file. If a match is made access is granted and the user is redirected to their profile page. Login screen below :



The messaging platform is accessed by the user clicking the "Messenger" button on the nav-bar. To send a message the user can type the username of the user they would like to send a message to in the "TO" text box and type the message in the text area below. After the message has been typed in, the user can select what encryption they would like to use on the message by using the cipher selection menu.



4 Critical Evaluation

The app manages to meet the very basic criteria that it was designed for. The user can register an account and login using a username and password, input validation helps make sure required input is entered.

MongoDB Atlas was initially used to persist the data but after a big chunk of time trying to insert data into the db it was abandoned and a simple file system functionality was implemented to store user data. The messaging platform is not fully complete, users can send other users messages but encryption facility is not fully implemented. Inbox fails to display received messages of user on screen.

5 Personal Evaluation

The assignment broadened my understanding on the implementation of a variety of different web technologies. This assignment not only required you to apply prior knowledge gained from labs but also helped me break down a problem into smaller manageable chunks. Breaking down the code into smaller pieces, using the the template provided by the express app generators helped with keeping track of whats going on.

Time allocation in an assignment like this should be thought out and followed rigidly otherwise one can lose traction and get sidetracked. A big chunk of my time was wasted trying to implement the MongoDB knowing that I could have met the basic criteria using the file system sooner, I could have used the time on implementing the messaging facility correctly. A more detailed project plan for the future would help tremendously, one that allocates time to each activity properly. A more rigid project plan can allow one to focus on the more important tasks before others.

6 References

<https://scotch.io/tutorials/build-and-understand-a-simple-nodejs-website-with-user-authentication>

https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm