

Coursework Assessment 2- Web Technologies **SET80101**

Ismael Souf

Edinburgh Napier University

I. Introduction

The aim of this coursework is to extend our cypher site from the first coursework to design and implement a coded messaging platform. The content of this coursework will include the previous pages from the first coursework, a server and client elements. My website is called "AloKet" which is displayed in the index. This project includes many pages such as cypher from the first coursework but we can now navigate into a messaging platform. However this is possible only if the user register.

The user can send encoded messages in the "AloKet" page to any user registered. In the other hand, the users can check the inbox page to view the messages received in order to decode them. And finally, a profile page available in order to change the password making the account more secured.

- The user profile gives a little control over their information.
- Password hashed and stored.

I decided to keep ciphers to be used by anyone in order to give an experience before trying to use AloKet.
In order to build my project I had to do background reading and watch videos.

II. Software Design

The approach of implementation

In order to create our client and server we are going to use:

- PUG for the creation of content such as dropdown, headings, button...
- CSS will be used to make the website prettier with colours, choose fonts and effects.
- JavaScript/JSON will be used to allow most of the functionality such as register, login, send messages and receive messages.
- Node JS (JavaScript code) will be used in order to approach the server side of the website.

List of Requirements

- The design will be consistent through pages with a navigation bar
- Profile experience (change password)
- A registration form
- A login form
- Messages sent and receive in different pages (Allowed only if registered)
- Test different cyphers from the first coursework before register
- Grey and black colours to match the visual
- Text will be visible in each page with the same colour and font
- Contact me

III. Implementation

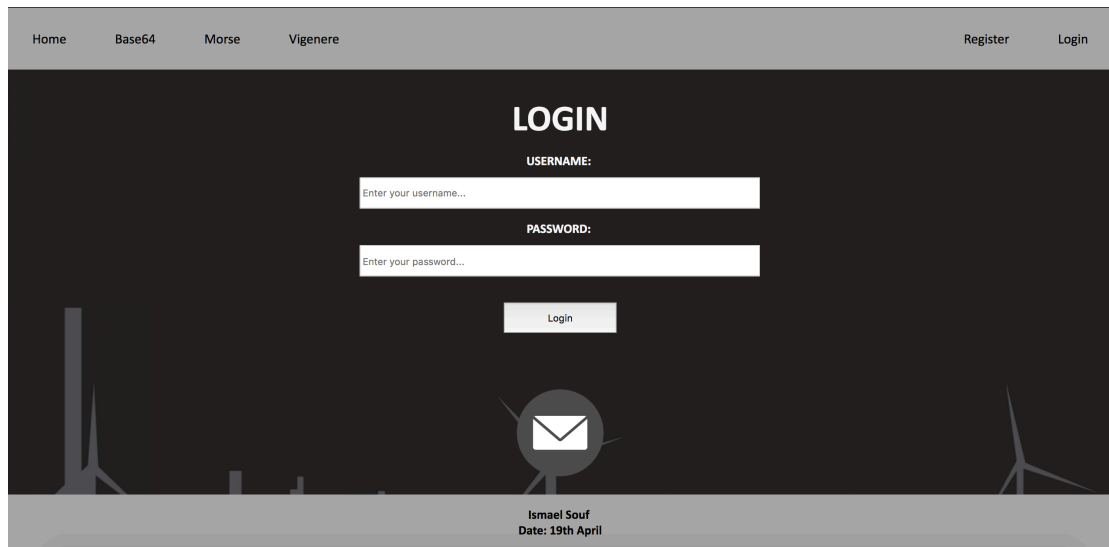
In order to implement this coursework, I used Node.js to write JavaScript code for the server side. For the client interface, all relevant .pug files were used instead of HTML because Pug is much more error-proof, given that you don't need to use closing tags and it looks cleaner. However, I kept my .html files in a folder because it is from my first coursework. We can find the pug files in the views folder of the project. These allow me to design my site.

Express: This module is recommended by the specification coursework and is a minimal and flexible node.js web application framework that provides a robust set of features for web and mobile applications.

Fs: I used Fs module to be able to read file and write file to JSON files. Handling JSON Files in Node.js is a good idea because I opted for the "stringify" method for converting the JSON object into a string data but also parse data. Every messages (sent/received) and user registered are store to a JSON files.

Bcrypt: I used Bcrypt to hash password using 10 salt rounds in order to increase strength and the security of the password entered by the user.

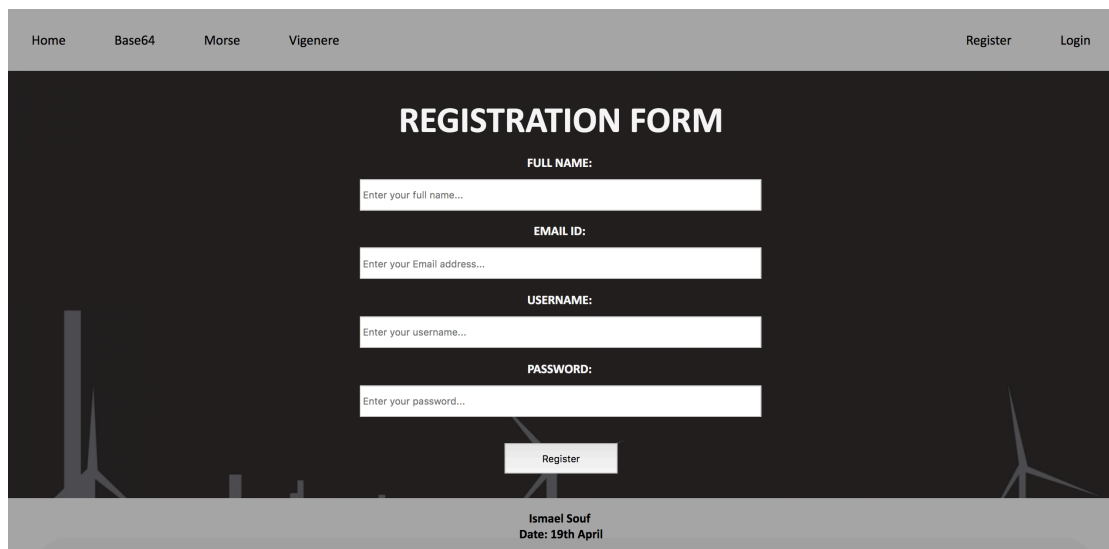
Cookie-parser: Cookies are still a great way of tracking visitors to a website including node.js projects made with express.js. In express the usual choice for parsing cookies is the cookie-parser module.



The screenshot shows a web application interface with a dark background and a light gray header. The header contains navigation links: Home, Base64, Morse, and Vigenere on the left, and Register and Login on the right. The main content area is titled "LOGIN" in large white letters. Below the title, there are two input fields: "USERNAME:" with a placeholder "Enter your username..." and "PASSWORD:" with a placeholder "Enter your password...". A "Login" button is centered below the password field. At the bottom of the page, there is a circular icon with an envelope inside, and the text "Ismael Souf" and "Date: 19th April" below it.

Figure 1. Login

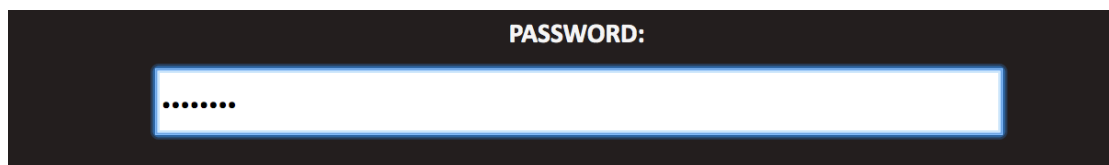
(The login page disappears once the user is logged in.)



The screenshot shows a web application interface with a dark background and a light gray header. The header contains navigation links: Home, Base64, Morse, and Vigenere on the left, and Register and Login on the right. The main content area is titled "REGISTRATION FORM" in large white letters. Below the title, there are four input fields: "FULL NAME:" with a placeholder "Enter your full name...", "EMAIL ID:" with a placeholder "Enter your Email address...", "USERNAME:" with a placeholder "Enter your username...", and "PASSWORD:" with a placeholder "Enter your password...". A "Register" button is centered below the password field. At the bottom of the page, there is a circular icon with an envelope inside, and the text "Ismael Souf" and "Date: 19th April" below it.

Figure 2. Register

In the Registration form the password is hidden and has a hash. A strong password storage strategy is critical to mitigating data breaches. So, I used Bcrypt and set the type of input to be password instead of text.



The screenshot shows a close-up of the password input field. The label "PASSWORD:" is in white text above the input field. The input field is a white rectangle with a blue border, containing several black dots to represent hidden characters.

Figure 3. Type password

Once the user is logged in there is a profile page, AloKet and inbox available to use. The profile will let the user change his password if he desire, AloKet send encode messages to the users selected. And the inbox will allow the user to see the username and the messages sent by anyone who is registered.

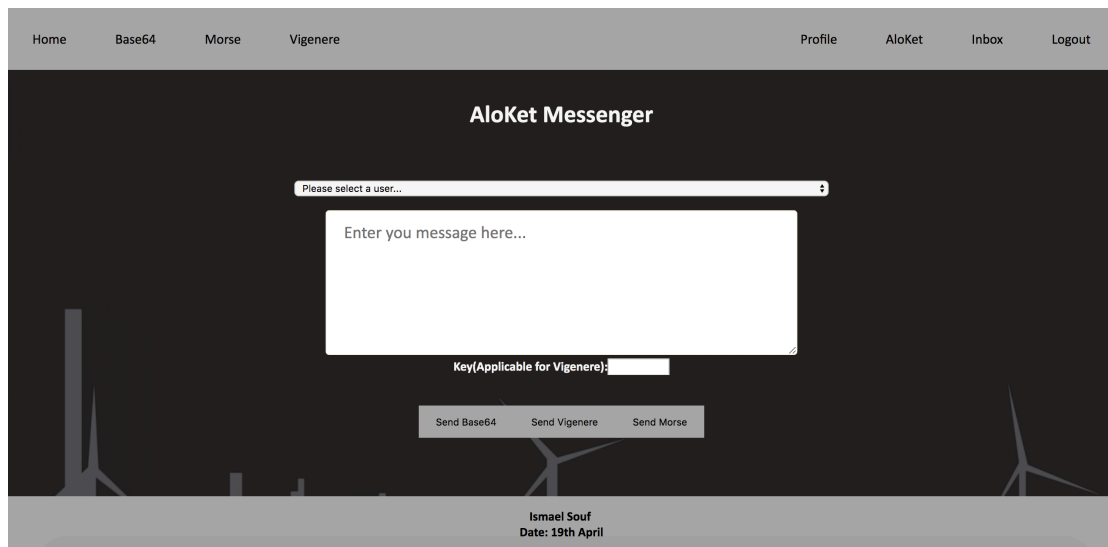


Figure 3. AloKet

IV. Critical Evaluation

The list of requirements set out in the coursework specification has been completed and features added:

- Implementation of a working Node.JS server
- User able to navigate between pages
- Sign-up for an account
- Send encoded messages to other users
- View decoded messages
- Messages are persistent
- Footer
- Colour's effect
- Brief introduction of the project in Home page
- Change password

Possible improvements to your application

- More security features
- Images
- Admin page delete account
- Mobile version of the website
- Better design
- Send notification
- More features such as animations

Comparing to other website, this coursework lack a lot of things due to the experience. But if I keep working on my skills I could improve my website into something which allow a good user experience.

V. Personal evaluation

Reflecting on what you learned

In this coursework I learned how to pursue my own path in studies like the first coursework. Now I have a good understand of Node Js and more experience with JavaScript. The client side is what I preferred the most because we have various ideas in mind that could bring a good user experience. Even though, don't like open coursework I found this one interesting and efficient in order to improve our skills in web design and server.

Challenges you faced

The main challenge I faced was "How do I start my coursework". I had no idea of what to choose in order to build my website. I started working with html then swapped to pug. Secondly, I tried to work with SQLite but changed to JSON. One week before the deadline I was behind and had couldn't make a choice. Thankfully the deadline changed and I managed to finish the coursework on time. Finally, I read books, looked at different website, videos. I spent a lot of time watching videos but it helped me understand basic features.

Methods you used to overcome challenges

In order to overcome my challenges I decided to follow the plan to look at different website by inspecting elements, look at the source, watching videos about Node JS. They were very helpful but I realised that I needed more time to do the coursework because I didn't know where to start. I have spent too much time in the implementation of the messages. I had different ideas of how to implement the messages such as the use of nodemailer, node imap... The mistake I made is that didn't want to start the coursework when I finished the first one because I had no intention at looking at the practical before the lecture. I was stuck for a week looking for different ways to build my platform instead of trying to do the practical. And I realised that JSON is what I needed and manage to finish the coursework after the new deadline announced for the 19th April.

Performed

So far in this module everything was new, especially this coursework, which show me that there is a lot of ways to build a web interface. However, I think that I did my best at trying to match the requirements and involved myself in the world of Node JS and JavaScript. It was also easy to work with pug files that I discovered in the practical.

I have lost a lot of time trying to choose how I would build my interface and was confused.

VI. Reference

Books recommended:

- Introducing Web Development
- Beginning Responsive Web Design with HTML 5 and CSS3
- JavaScript Quick Syntax Reference
- Beginning Node.js
- Pro Express.js (Template Engines and Consolidate.js/Socket.IO and Express.js)

YouTube:

- 8.5: Saving Data to JSON File with Node.js -
<https://www.youtube.com/watch?v=6iZiqQZBQJY>
- Node.js + Express – Tutorial – GET and POST Requests -
<https://www.youtube.com/watch?v=Sb8xyCa2p7A>

Stack Overflow:

- <https://stackoverflow.com/questions/10011011/using-node-js-how-do-i-read-a-json-file-into-server-memory>
- <https://stackoverflow.com/questions/44465006/nodejs-use-a-pug-template-to-display-results>