Coursework 2 Report

Steven Pyper

40319882@napier.ac.uk

Edinburgh Napier University – Web Tech (SET08101)

Contents

[1. Introduction 2](#_Toc6490296)

[1.2 Server Side 2](#_Toc6490297)

[1.3 Database 2](#_Toc6490298)

[2. Software Design 2](#_Toc6490299)

[2.1 Database Interaction 2](#_Toc6490300)

[2.1.1 Register Account 2](#_Toc6490301)

[2.1.2 Login to Account 2](#_Toc6490302)

[2.1.3 Update Password 2](#_Toc6490303)

[2.1.4 View Messages 2](#_Toc6490304)

[2.1.5 Remove Messages 3](#_Toc6490305)

[2.1.6 Send Message 3](#_Toc6490306)

# Introduction

The main scope of this coursework will be to build a website that allows users to create accounts and send messages to other users (or themselves) which will be stored in a database ciphered. This data will then be deciphered when it is being sent to the recipient allowing them to view it in plaintext. There will be two main sections, the server routing and setup and then there will be the database section. As I know my design skills are lacking just due to the way I think and see things I will be focusing on adding as much functionality as I can. I will implement a system that allows users to create an account, login, allow them to update their password if they are logged in and to logout manually. Additionally, I will allow them to send messages to themselves and to other people based on usernames and then view these messages on a different page. These messages will also allow for lots of different methods of ciphering(which since the other user will see it in plaintext anyway is more to allow the user to feel like they have a choice) the methods the user will be allowed to use are a Caesar cipher, a direct substitution cipher and Morse code. As an additional goal if there is extra time there will be an additional button which allows viewers to see their messages in ciphered text.

## 1.2 Server Side

The server side must allow the user to route around the website in a way that makes sense, it must allow them to login or register a account, then it should show them pages that they could only see if they were logged in, if they are not logged in they must be restricted in what they can(in reality there would be static pages such as the first coursework but these will be excluded as they will no longer fit with the website although if there is enough time they will be added in).once a user has logged in they will see a slightly different website, in theory it should be possible to change the navigation bar to then give them the links to see there messages, to send messages and to log out if they wish. The server will also have to deal with logged out users trying to access pages that they are not allowed, the simplest way to do this would be to redirect to the login page as it is likely they are a user if they have those links saved. The server will have to have a system to track which users are logged in and identify who each request comes from. This can be done with cookies which will be made unique to the user and to when the server was started.

## 1.3 Database

The database will have to contain a table that will keep track of users such as their username and password. Another table will keep track of messages showing who sent them, who they are for and what the message itself contains. This will allow the server to interact with the tables and join them in a way to get the correct information for the specific user. The user’s password will eventually have to be stored in a encrypted way, so they are not stored in plaintext as this would be a massive security issue.

# Software Design

The initial design for the server will be to simple allow users to register and login and send plaintext messages to each other. To do this I will need a database system in place, I have chosen to use sqlite3 as we used it In the lab and there is a lot of information online showing how to use it, also as I already know SQL this is another advantage as I don’t need to learn a new language as well.

## 2.1 Database Interaction

There will be a lot of interaction with the database as most things will need to be verified before allowing the user to access anything. The database section will not contain information about encrypting the data as I have decided that is more of a server-side programming issue than a database side (even though the database will store the encrypted password).

There will be two tables. A User table which will contain Usernames, Passwords and Cookies. This will allow a user to be validated. The second table will include a Message ID as there is no nice way to have a unique ID excluding a automatically incrementing number. it will also store the Sender of the message, the Recipient, the message itself, as well as a way to show which cipher was used to cipher the messages.

### 2.1.1 Register Account

When a user creates an account, they will enter a username and password that they would like to use for there account. Due to this there will have to be a check to make sure that the username is already free and not in use by another user. This will consist of a select statement which will look for that username, if a result is returned that means that the username is already in use. Therefor the user should be informed of this and asked to choose a different name. once they have a username that is not already in use a database command will run that inserts the username, the password and the cookie that will be used to keep them logged in.

### 2.1.2 Login to Account

Logging into an account will require the user to enter their username and password. This then will start a select statement looking for that username, if the username is found that means the user exists and then you can compare the password in the database and password that they entered. If the passwords match, they will be logged in. If the user isn’t found or the passwords do not match the user will not be logged in. when a user successfully logs in the cookie in the database will be updated to the newest cookie.

### 2.1.3 Update Password

This should be fairly simple as the user will already be logged in, so the same statement that gets used to validate the user is logged in and the cookie they use will be used, then if they are validated a simple statement that will update the users row I the database with the new password can be used.

### 2.1.4 View Messages

This will be a slightly more complicated as the user is only allowed to see messages that is directed to them. This means there will have to be a statement that can figure out what user is logged in and use them as the recipient and get every single message from them from the database. The best way to do this would be to either do a foreach or an all database statement, these will both work in very similar ways but will slightly change the code. using the join statement would allow for there to only have to be one search to the database to get this information back which is important as this is the most likely to be used functionality of the website and as such should be more efficient than the rest of the website. Once this information is read form the database it should be sorted so that the most recent messages are shown first, and I might limit them just as a way to keep it tidier looking for now.

### 2.1.5 Remove Messages

Allowing users to delete their messages is an additional feature that would be useful but not necessary. For this you would have to do a statement o validate the user that is logged in, then another statement to make sure that the message they are trying to remove is their own. Finally, you would then run the delete statement removing that message from the database.

### 2.1.6 Send Message

Again you will have to validate the user who is logged in as they will be the sender, then a statement will be required to make sure that the user they are trying to send the message too actually exists because sending messages to no one will take up space and give no benefit to the user. Finally, an insert statement will be used that will insert the sender, recipient, the message content that the user has entered, and anything required for the ciphered messages to be deciphered.

## 2.2 Server-Side Coding

### 2.2.1 Routing Logged Out Users

As logged out users will be very limited in what they could do on the website they will have very limited abilities on the website, all they should be allowed to access is the login and register page, this means that the navigation bar should be changed to only let them see these.

### 2.2.2 Routing Logged Out Users

Logged in users have a lot more choices and as such will have to be validated often whenever doing tasks that would require the user to be logged in. they should also be shown a different nav bar that will show them links to get to there messages, sending messages, updating there passwords and logging out.

### 2.2.3 Passwords And Cookies

As passwords should not be stored in plaintext they will instead be stored in the database using bcrypt, this will allow them to be encrypted before the database and never store a plaintext version, when a user tries to login the plaintext password they send will then be encrypted and that compared ot the one in the database. Once a user has successfully logged in they will be given a cookie which will contain the time the server started and a signature of there username which should help to keep them more unique and less guessable by malicious attacks. These cookies will be stored in the database when the user is logged in allowing them to be quickly compared to the database helping to speed up the validation of that user. Theses cookies will either last only for a day or until the server is restarted.

### 2.2.4 Ciphering/deciphering the Text

Ciphering the text will be more complicated due to the fact that I have multiple ciphers with different keys(as well as substitution technically having two keys) what this will mean is that when the user is selecting how they wish to cipher the text they will have to choose these ciphers and keys. Then server will have to deal with all them individually and make sure to only look for the correct information, which is dependent on ciphers and methods, then the text will be ciphered using a modified version of the ciphers that were implemented during the last coursework. This will return the text which can then be sent to the database. Deciphering the text should be relatively simple once the ciphers have been implemented as the keys will just be reversed.

### 2.2.5 Logging in

The login page should only be visible to users who have not logged in and as such a redirect will be used to take them to the messages page. However if a use is actually trying to log in

## 2.3 Additional Features

An additional feature I would like to implement is an admin account which will be allowed to remove messages and users from the database. This should be simple as you would just do a check that the cookie lines up with the admin username in the table then delete statements depending on what is being removed. If a user is removed all messages that are sent to them should also be removed.