

# 1 Group members

Andrew Sturdy - 100318044

Thomas

Tan

## 2 Privacy and ethics

## 3 Mitigations

### 3.1 SQL Injection

SQL Injection is when SQL code is ran on a website via a textbox or entry field. The way we have mitigated this is by using parametrised queries wherever the user inputs data for a query. A parametrised query is a query that "drops in" the variables instead of using string concatenation. Because the values are not added on the end and are instead dropped in, the user has no way of terminating the query to run their own code. We used the node library "pg" to communicate with the database, this library had built in support for parametrised queries so no other library was needed. These queries function the same as normal queries and run at similar speeds to normal ones meaning the user wont notice the difference. As this is all happening server side it will have no direct impact on usability, however if these queries were to run slow then the user would notice.

### 3.2 Account Enumeration

Account Enumeration is when a user iterates through a dictionary of possible username and uses the response from the server to determine if that username is in use or not. The way we have prevented this is by making the response from the server the same whether the username or the password is incorrect making it hard to tell which one if any were correct. We have also made the timing of the responses the same so that they can not be told apart that way, this was done by running through the entire login process no matter whether the username or password was correct. This can have an impact on usability as it can make the process of forgetting login information slower and less informative, if you forget your credentials the website wont tell you which of them you got right which can be frustrating for users.

### 3.3 Cross-site Scripting

Cross-site Scripting is when code is injected into a website to then be executed later, this typically happens in the form of html and JavaScript code that gets issued to a text form to run when its displayed later. The way we have mitigated this is through html encoding, all of the html characters (<, >, ", &) needed to write code have special characters that represent them so websites can display them safely, we use this when displaying posts as its the only time code could be injected to run later. Before the posts are displayed on the screen they all go through a function that converts html characters to their display counterparts so < becomes &lt while being ran but will display as <. This should have little to no affect on usability as the users wont see the converted characters and the process is very fast so it should not be much slower than if we didn't do it.

### 3.4 Cross-site Request Forgery

### 3.5 Session Hijacking

### 3.6 Hashing, Salting and Encryption

The passwords of each user is salted with their user id allowing their passwords to be distinct when hashed even if they were originally the same password. The reason the user id was chosen is because the hashes needed to be repeatable as the user id is something unique to the user that is accessible to the server.

Hashing was used on user's passwords before they were entered into the database, we used SHA-128 as our hashing algorithm, which is the 128 bit version of SHA-1. This hashing algorithm was implemented using a pre-built library called jssha which adds functions to hash using most versions of SHA. We chose SHA-128 because it produces strong but small hashes. As this web blog should not contain a lot of personal information there is no need for very strong security.

The encryption we used was AES-128-cbc which uses a 128 bit key size and performs 10 rounds of encryption. We use this algorithm in cipher block chain mode (cbc) which allows the output of the encryption to be of fixed sizes as cbc causes the data to be padded to the block size. All user data that enters the database is encrypted but the post data is not. Posts are not encrypted as they should contain no data about the user and encrypting data that does not need to be would slow down the flow of the website as all posts would need to be decrypted to be displayed.

## 4 Authentication methods

## 5 Testing

### 5.1 Think Aloud Testing

Think aloud testing is when users are handed the product, in this case the website, and asked to perform a series of tasks, they are monitored during this and their progress is recorded along with any issues they encounter. This type of testing is for usability and is typically performed in 3 or more rounds, you would record the results of people in different demographics and using the notes make improvements, after the improvements a new group of people from the same demographics would use the site.

The tasks the users were asked to perform were:

- Register an account
- Make a post
- Log out
- Log in
- View the post you made
- View all available posts

These tasks were performed in the order listed and allowed us to test all aspects of our site individually. We tested on 3 different demographics, young male university students, young female sixth form students and female adults.

The young male managed to create an account with ease and make a post, however he noted that the website gives a lack of information on whether the post was successfully made or not

causing this task to take longer than expected. He had no problems logging out and then back in, he tried to search for the category his post was under however our search bar only works based off of title so it didn't work. To view all posts he simply pressed buttons on the screen around the search bar and happened to press the search button while the bar was empty which shows every post. Overall he did not find it easy to use due to the lack of feedback the site gives on whether actions are successful or not. The table of results for this test can be found in Figure 3.

The young female did not like the 2 factor authentication used as it required a new app on their phone. She had no issues until tasked with viewing all posts, after a couple of minutes of looking they gave up. The table of results for this can be found in Figure 2.

The adult female had the same issues as the young male, they didn't know if the test was successfully made and they did not easily find the way to search for all posts. The table of results can be found in Figure 1.

## 5.2 Unit Testing

This website was thoroughly tested using unit tests, these tests were carried out throughout development and also at the end. The final test results can be found in figure 4 and it shows that all but 2 tests were passed. The two tests that did not pass are failing two factor authentication (test 30) and Searching for something that does not exist (test 32). In both cases the behaviour demonstrated is valid however the user is not informed. If you get the two factor authentication code wrong at log in or registration it does not tell the user this but it does not log the user in either. If the user searches for a title that has no results, the user is not told that nothing could be found but nothing is displayed either.

Female Adult

StepNum	Step	Comments
1	Register an account	No issues
2	Make a post	Doesn't know if the post was made
3	Log out	No issues
4	Log in	No issues
5	View the post you made	No issues
6	View all posts	Didn't know how to Guessed correctly – Not intuitive

Figure 1: Female Adult

Young Female Sixform student

StepNum	Step	Comments
1	Register an account	No issues - Didn't like having to download a new app
2	Make a post	No issues – assumed the post was made
3	Log out	No issues
4	Log in	No issues
5	View the post you made	No issues
6	View all posts	Didn't know how to

Figure 2: Young Female Student

Young Male University Student

StepNum	Step	Comments
1	Register an account	No issues
2	Make a post	Doesn't know if the post was made
3	Log out	No issues
4	Log in	No issues
5	View the post you made	Tried to search by category not title
6	View all posts	Didn't know how to Guessed correctly – Not intuitive

Figure 3: Young Male Student

Figure 4: Unit Testing plan/results

Sr.No.	Module	Sub-module	Pre-Requisite	Steps to be followed	Expected Result	Actual Result	Comments	Status (Pass / Fail)
1	Create Database	Connect	Client object defined	Call the createDatabase function	"Client connected" appears in console with no errors	"Client connected" appeared in console with no errors	Didn't pass using callbacks but did using promises	Pass
2	Create Database	Creating the database	Client object defined, client connected successfully, table doesn't already exist	Call the createDatabase function	"Database was successfully created" appears in console with no errors and the database should be visible in pgAdmin	"Database was successfully created" appeared in console with no errors and can be seen in pgAdmin		Pass
3	Create Table	Connect	Client object with password defined	Call the createTable function	"Client connected" appears in console with no errors	"Client connected" appeared in console with no errors	Didn't pass using callbacks but did using promises	Pass
4	Create Table	Creating user table	Client object defined, client connected successfully, table doesn't already exist	Call the createTable function	"User table was successfully created" appears in console with no errors and the table should be visible in pgAdmin	"User table was successfully created" appeared in console with no errors and can be seen in pgAdmin		Pass
5	Create Table	Creating posts table	Client object defined, client connected successfully, table doesn't already exist	Call the createTable function	"Post table was successfully created" appears in console with no errors and the table should be visible in pgAdmin	"Post table was successfully created" appeared in console with no errors and can be seen in pgAdmin		Pass
6	Salt		Some text to be salted and what to salt them with	Call the salt function with an input of hello and a salt of 1	"hello" will appear in the console with no errors	"2hello" appeared in console with no errors		Pass
7	Hash		Some text to be hashed	Call the hash function with an input of hello	"88f6d585121a4ccb3d1540527aee53a77c77abb8" will appear in the console with no errors	"88f6d585121a4ccb3d1540527aee53a77c77abb8" appeared in console with no errors	This test should be run multiple times to make sure the output is consistent	Pass

8	Encrypt		Some text to be encrypted	Call the encrypt function with an input of 88f6d585121a4ccb3d1540527aee53a77c77abb8	"3b3747e6e7eb406b4d51f386d7b0b0c986f4eda40554eb5cd8f95b78b04706ae39ea21b6fc8f02ca897d9fbb301d" will appear in the console with no errors	"3b3747e6e7eb406b4d51f386d7b0b0c986f4eda40554eb5cd8f95b78b04706ae39ea21b6fc8f02ca897d9fbb301d" appeared in the console with no errors	This test should be run multiple times to make sure the output is consistent	Pass
9	Register	Connect	Client object with password defined	Perform an app.post request for /register (attempt to register to the site)	"Client connected" appears in console with no errors	"Client connected" appeared in console with no errors		Pass
10	Register	Insert into database	Client object defined, client connected successfully, details entered into register form	Perform an app.post request for /register (attempt to register to the site)	The data should be visible in the database (using pgAdmin)	The data is visible in the database		Pass
11	Login	Connect	Client object with password defined	Perform an app.post request for /login (attempt to log into the site)	"Client connected" appears in console with no errors	"Client connected" appeared in console with no errors		Pass
12	Login	Setting session variable	Correct login details	Perform an app.post request for /login (attempt to log into the site)	You should stay logged in through pages refreshes	You stay logged in through page refreshes		Pass
13	Register	2 Factor authentication QR code	Enter valid credentials into the register form	Perform an app.post request for /register (attempt to register to the site)	You should be able to scan the QR code using google authenticator and have a 6 digit code that is randomly generated	You get a 6 digit code that is randomly generated on google authenticator		Pass

14	Register	2 Factor authentication	Enter valid credentials into the register form	Perform an app.post request for /register (attempt to register to the site)	Enter the 6 digit random code from google authenticator and then be logged in	You successfully log in after entering the 6 digit random code generated by google authenticator		Pass
15	Login	2 Factor authentication	Enter valid credentials into the login form	Perform an app.post request for /login (attempt to log into the site)	Enter the 6 digit random code from google authenticator and then be logged in	You successfully log in after entering the 6 digit random code generated by google authenticator		Pass
16	Escape		Some text containing html tags	Call the escape function with the input of "<script>this is a script</script>"	A returned string of "&lt;script&gt;this is a script&lt;/script&gt;"	A returned string of "&lt;script&gt;this is a script&lt;/script&gt;"		Pass
17	Search	Connect	Client object with password defined	Perform an app.post request for /search (attempt to search for a post)	"Client connected" appears in console with no errors	"Client connected" appeared in console with no errors		Pass
18	Search	Displaying posts	Client object defined, client successfully connected, query results, a post to search for	Perform an app.post request for /search (attempt to search for a post)	All posts of a similar title should appear under the search bar and should enlarge when hovered over	All posts of a similar title appear under the search bar and enlarge when hovered over		Pass
19	Search	Escape	Client object defined, client successfully connected, query results, a post to search for that contains html tags	Perform an app.post request for /search (attempt to search for a post)	All posts of a similar title should appear and the html tags should look correct but not function as valid html	All posts of a similar title appear and the html tags look correct but do not function		Pass
20	Create post	Connect	Client object with password defined	Perform an app.post request for /uploadpost (attempt to create a post)	"Client connected" appears in console with no errors	"Client connected" appeared in console with no errors		Pass

21	Create post	Insert into database	Client object	Perform an app.post request for /uploadpost (attempt to create a post)	The user should be redirected to the home page and the posts should be in the database and searchable for	The user was redirected to the home page and the post is in the database and can be searched for		Pass
22	Logout		A valid active session (logged in)	Perform an app.post request for /logout (attempt to logout)	The user should be redirected to the login page and there session should be destroyed (refreshing the page wont log them in)	The user was redirected to the login page and refreshing the page did not log them back in		Pass
23	isAuthenticated			Perform an app.post request for /login (attempt to log into the site)	The user should be redirected to the home page after entering valid credentials	The user was redirected to the home page after entering valid credential		Pass
24	app.get(/)			Go to https://localhost:5000	You should be directed to the login page	The user was directed to the login page		Pass
25	app.get(2FA)			Successfully go through the first stage of registration or logging in	User should be redirected to a screen that asks for a 6 digit code and contains a QR code if you are registering and doesn't if you are logging in	The user was redirected to a screen with a qr code during registration and when login in did not see the qr code		Pass
26	app.get(register)			Press the register button on the login screen	The user should be redirected to a registration form	The user was redirected to a registrations form		Pass
27	app.get(createpost)			Press the create post button after successfully login in	The user should be redirected to the create post screen	The user was redirected to the create post screen		Pass
28	Login	Fail		Enter invalid credentials into the login form	The user should receive a warning that either the username or password are incorrect	The user receives a warning that either there username or password is incorrect	The warning should be timed so that no matter which is incorrect the timing is the same	Pass

29	Login	Logout		Enter invalid credentials into the login form 10 times	The user should receive a message saying they have attempted too many times and they shouldn't be able to attempt again for a set period of time	The user receives a message saying they have attempted too many times and can no longer log in till session has expired		Pass
30	2 Factor Authentication	Failed		Enter an invalid code into 2FA	The user should receive a message saying they have entered an incorrect code and they should not be logged in	The user receives no message but is not logged in		Fail
31	Register	Fail		Enter credentials that are already in use into the register page	The user should receive a message informing the user that either the email or username is already in use	The user receives a message saying that the entered email or username is already in use	The warning should be timed so that no matter which is incorrect the timing is the same	Pass
32	Search	No result		Search for a title that is not like any posts that exist	The user should be informed that no results where found	The user is redirected to a page to display posts but no posts are displayed (none to display)		Fail
33	Login	No Username		Attempt to log in while leaving the username field blank	The user should be informed that either their username or password are incorrect, or they should not be allowed to submit the form as it is a required field	The user was told that the username field is required and they could not submit the form	This only works when accessing the site through a web browser as it is done html side not server side	Pass
34	Login	No Password		Attempt to log in while leaving the password field blank	The user should be informed that either their username or password are incorrect, or they should not be allowed to submit the form as it is a required field	The user was told that the password field is required and they could not submit the form	This only works when accessing the site through a web browser as it is done html side not server side	Pass

35	Register	No Username		Attempt to register but leave just the username field blank	The user should be informed that a username is required	The user was told that a username was required	This only works when accessing the site through a web browser as it is done html side not server side	Pass
36	Register	No Password		Attempt to register but leave just the password field blank	The user should be informed that a password is required	The user was told that a password is required	This is done HTML side so will only work in web browser	Pass
37	Register	No Email		Attempt to register but leave just the email field blank	The user should be informed that an email is required	The user was told that an email is required	This is done HTML side so will only work in web browser	Pass