# Section 4 – Evaluation

## Part (A) – Post Development Testing

As outlined in the design section of my project, I originally planned on using four method of testing:

* White box testing
* Black box testing
* Destructive testing
* Unit testing

However as my project progressed I realised how long it was going to take to finish it. I then chose to ignore unit testing as it was very time intensive and didn’t provide many benefits for a project this small scale (compared to a project made and maintained by a team of people over several years). This is bad in terms of maintainability, so if I was continuing to develop this project, writing unit tests would be one of my first priorities to ease future development. I did use all three of the other testing approaches.

I also performed tests that didn’t require any user data such as simulated load tests on the server and database. This is useful to see how the website would perform if it had lots of concurrent users.

### White box tests

White box testing is when a programmer with access to the source code tests the program for functionality and to locate bugs. They look at the source code and test legal and illegal inputs, as well as checking every section actually does what its supposed to do. I used white box testing in my project as my primary testing method because it’s quick and it doesn’t require asking anyone else to test it, which is good for pre-release testing. I tested all the scripts that required input using white box testing by checking to code to see what could trip the program up, and then testing it against expected results. The results of my white box testing are below:

Browser tests:

|  |  |  |
| --- | --- | --- |
| **Browser** | **Works/Doesn’t work** | **Notes** |
| Mozilla Firefox | Works | None |
| Google Chrome | Works | None |
| Internet Explorer 11 | Works | None |
| Microsoft Edge | Unknown | Won’t connect to website at all so impossible to test |
| Mozilla Firefox (Android) | Mostly works | Site hasn’t been specifically designed for mobile but most pages work. |
| Google Chrome (Android) | Mostly works | Site hasn’t been specifically designed for mobile but most pages work |

Basic functionality:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Page** | **Test** | **Test data** | **Works/Doesn’t work** | **Notes** |
| signup.php | Can create an account | All valid data | Works | Script creates an account using the data provided |
| signup.php | Can create an account | No data | Works | Script prompts user to fill in missing fields |
| signup.php | Can create an account | ☺ ☺ ☺ ☺ ☺ | Works | Script prompts the user to try again |
| login.php | Can log in to existing account | Username and password for an account existing in the database | Works | Script logs user into account |
| login.php | Can’t log into account with wrong password | Username for an account that exists in the database, wrong password for said account | Works | Script prompts user that username/password combination is invalid |
| login.php | Can’t log into account that doesn’t exist | Username and password that doesn’t exist in database | Works | Script prompts user that username/password combination is invalid |
| ask.php | Can ask a question with valid inputs | Valid question title and question body | Works | Script adds a new question to the database |
| ask.php | Can’t ask a question with nothing filled in | No data | Works | Script prompts the user to fill in every field |
| ask.php | Can ask a question containing unexpected characters | ☺ ☺ ☺ ☺ ☺ | Works | Script adds a new question to the database |
| searchq.php | Can search for a question with valid inputs | Valid search String | Works | Script shows all questions that match the search query |
| searchq.php | Can’t search for a question without supplying any query | No data | Works | Script prompts the user to search for questions |
| searchq.php | Can search for a question containing unexpected characters | ☺ ☺ ☺ ☺ ☺ | Works | Script shows all questions that match the search query |
| comment.php | Can comment on a question using valid input | Valid input comprised on standard characters | Works | Script adds a new comment to the database |
| comment.php | Can’t comment on a question without supplying a comment | No data | Works | Script prompts the user to input a comment |
| comment.php | Can comment on a question using unexpected input | ☺ ☺ ☺ ☺ ☺ | Works | Script adds a new comment to the database |
| findpeople.php | Can post a job listing using valid data | Valid job title, description, location, company, salary and tags | Works | Script adds a new job to the database |
| findpeople.php | Can’t post a job listing with no data | No data | Works | Script prompts the user to fill in all the fields |
| findpeople.php | Can post a job listing with unexpected characters | ☺ ☺ ☺ ☺ ☺ | Works | Script adds a new job to the database |
| findpeople.php | Can add a valid tag | A valid tag | Works | Script adds a new tag to the filter |
| findpeople.php | Can’t add a tag with no data | No data | Works | Button to add tag does not appear if the input box is empty |
| findpeople.php | Can add a tag with unexpected characters | ☺ ☺ ☺ ☺ ☺ | Works | Script adds a new tag to the filtered |
| findPeople.php | Can search for a job with valid input | Valid search String | Works | Script shows all jobs that match the search query |
| findPeople.php | Can’t search for a jobs without supplying any query | No data | Works | Script prompts the user to search for jobs |
| findPeople.php | Can search for a job containing unexpected characters | ☺ ☺ ☺ ☺ ☺ | Works | Script shows all jobs that match the search query |
| findjobs.php | Can add a valid tag | A valid tag | Works | Script adds a new tag to the filter |
| findjobs.php | Can’t add a tag with no data | No data | Works | Button to add tag does not appear if the input box is empty |
| findjobs.php | Can add a tag with unexpected characters | ☺ ☺ ☺ ☺ ☺ | Works | Script adds a new tag to the filtered |

### Black box tests

Black box testing is when the tester has no knowledge of the internals of a program. They do not have access to the code and they don’t know how it works. Black box tests are often used to test functionality and interfaces. I’m using black box tests because the final version of project is so large and complex that it’s very difficult for me to test every single feature. I also use it because I’m biased in favour of the design and interface, and black box testing allows me to see lots of user’s opinions. This will allow me to see if it meets my success criteria.

In order to black box test my project I created a survey using Microsoft Forms. The survey has seven questions designed to test various aspects of my site. When I was designing the survey I tried to make questions that test the parts of the site that I would struggle to do myself, such as opinions on the user interface and how easy it is to use the site etc.

The testers who will be black box testing my project are a class of 14-15 year old GCSE Computer Science students. This is because they are available for me to use for testing the project. This is a very effective way of me getting 30 people’s opinions on my website.

The list of questions in my survey and a link to it are below:

1. Did you like the design of the website?

(yes or no, compulsory)

2. Did you struggle to do anything on the website?

(yes or no, compulsory)

3. Did you find the tutorials easy to follow?

(yes or no, compulsory)

4. Is the interface memorable? (Would you remember how to use the website after a break of a few months?)

(yes or no, compulsory)

5. Was it easy to use each of the features for the first time?

(yes or no, compulsory)

6. What (if any) improvements would you like to see?

(text input, optional)

7. Please list any bugs you saw along with steps to reproduce them

(text input, optional)

Link to survey:

<https://forms.office.com/Pages/ResponsePage.aspx?id=PYCwG3TptkKUJQEK3ByuatBasbOGJYRIgymUpAILxi1UOFYyUVZDV05SVkJSTVdLU0JDQ0RZMFVCWS4u>

The responses I got from my survey were very positive. In total 21 people responded, which I consider a big enough sample size to draw meaningful conclusions. On average people took exactly three minutes to complete the survey which shows people thought about their answers. This makes me more confident in the data. Analysis of the survey responses are below:

1. Did you like the design of the website?

86% (18/21) of people said they liked the design of the website. This means that the user interface and design of my website was successful. As a programmer I’ve always struggled with design and making things look nice, and I was please at the feedback in this category. The design of the website is the first thing a user sees when they first use it, and having something that people find visually pleasing is essential for keeping people using the site.

2. Did you struggle to do anything on the website?

71% (15/21) of people said they didn’t struggle to do anything on the website. While this is slightly lower than the design feedback, it is still very positive considering the majority of the testers were not the target audience for my project. During my analysis I predicted that most of my users would be amateur and more experienced programmers, mostly employed/looking for work, however the testers are all 14-15 and just starting to study computer science. This means that figure of 71% is even better, which shows that some of the design work I did paid off. If people struggle to use a site they’ll give up using something straight away.

3. Did you find the tutorials easy to follow?

86% (18/21) of people agreed that the tutorials I wrote were easy to follow. Unlike the some of the other features in my project, the tutorial pages are aimed directly at the kind of people who were testing my project. I was happy with the high percentage of people that thought the tutorials were easy to follow because it shows that they worked as tutorials. There’s no point having hard to follow tutorials as they're frustrating and don’t end up helping people.

4. Is the interface memorable? (Would you remember how to use the website after a break of a few months?)

71% (15/21) of the testers considered the interface memorable enough to be able to use the website after a break of a few months). Memorability was one of my success criteria, and I consider 71% to be enough to pass this criteria. Memorability is a hard aspect to test for, because it really needs the testers to take a break of a few months, however there wasn’t enough time for this. Because of this I’m not surprised that the same percentage of people that didn’t struggle to do anything on the website also considered the interface memorable. I included this in the survey because I wanted to get people’s initial impressions, and I was pleased with the feedback from this question.

5. Was it easy to use each of the features for the first time?

This is a harder question to analyse because I added a third answer to this question. 67% (14/21) of people thought every feature was easy to use for the first time and a further 29% (6/21) thought most of the features were easy to use for the first time. Only 1 person said that none of the features were easy to use for the first time. This is quite a positive response, but also gives me some insight one what to work on in the future.