Helpdesk and IT Technician Request Form for WearView Academy:

Testing Table and Critical Evaluation for the System

Name: Samuel Neil Holmes

Testing Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Number | Test Title | Test Description | Predicted Result | Actual Result | Fix |
| 1 | Database Connection | The database connection is established throughout the project using a database.php file. Within this file a MySQLi object is created that represents the connection and the connection parameters. After writing the file it was important to check for successful connection. I included error handling within the file in the case of an unsuccessful connection and echoed any error messages before the site was out of production. | That the database would connect and provide the connection success message that was coded. | An error message that stated the arguments for the connection were in the wrong order. | To firstly find the issue during the connection it was important to establish a display if connection was unsuccessful. To understand the correct order of arguments, relevant documentation was consulted. The file was then altered accordingly. Upon retesting the connection, the successful connection message was displayed. |
| 2 | Database Creation | It was necessary to check that the database structure had been created correctly in line with the SQL query script. This was significant especially due to the use of hashed passwords within the database. | That the tables would be created correctly and that the password’s contained within the Staff\_User\_Auth and Tech\_User\_Auth tables had been hashed with the selected SHA 256 cryptographic hash function. | There were multiple errors with the SQL syntax when I first attempted to create the database. The file import within PHP MyAdmin would not execute the query for database creation due to this. | Consultation of documentation helped identify the errors made so these could be corrected. It was positioning of commas and brackets in table creation. Once these errors were adjusted the database was created with the structure specified. As this is a smaller project with less information contained within the database tables consulting the structure row by row allowed me to see the whether the database had been created correctly. For larger projects test queries would need to be executed |
| 3 | URL Access | This test was regarding access of the pages created. The pages had been developed in VS code and tested using webspace hosting. I wanted to ensure that the pages could be accessed via their URL and function as intended. To achieve this, I used a secondary device and only attempted connection with the pages using their provided URL. | That the page would be accessible and function as intended. | The page was accessible and functioned as intended. | No fix required for this test the predicted result matched the actual result. |
| 4 | Submission of a Ticket | At this stage, a separate PHP file to handle form submission using a POST method had been created. It was important to check that the data was being sent to and stored in the database. Using a prepared SQL statement could allow for errors if the parameters had not been bound correctly or the SQL statement was failing to prepare correctly. | The page would be operational and the data from the submission of the form would be posted to the form\_submit.php file. These values would then be correctly bound to a prepared SQL statement. The statement could then be executed to store the data in the database. | There was a syntax error within the prepared SQL statement. One of the column names that I had specified to INSERT INTO did not exist due to this grammatical error. | As a description of the error was provided, I consulted both the code and PHP MyAdmin to establish the discrepancy. I updated the SQL statement to contain the correct column identifier. I then retested the form submission and ensured that the relevant data had in fact been stored in the database |
| 5 | Viewing Submitted Tickets | Having created a PHP file for admins to view request tickets I needed to check the prepared statement was executing and the if conditional and while loop were working as intended. | The code would work correctly, or I would get an error due to the if and while loop as I found this quite complicated to construct. | There was an issue within the code which meant that only one row of the form data table was shown. Two test tickets had been submitted but only one was visible. | I researched this issue online and found a stack overflow discussion pertaining to a similar issue. Due to this I found out that I had made the same error in which I had included the table closing tag within my while loop rather than outside it. This meant the table was being closed after the first iteration. I made this adjustment and both rows were displayed correctly. |
| 6 | Submission of a Multi-Line Ticket | It was important to check how legible a ticket with multiple lines within the issue description field would be. This is regarding viewing within the tech requests page. | Ticket would be legible with the necessary data displayed. | Ticket was legible. | No fix required. |
| 7 | Job Status Update | The href contained in the table listed with the value complete in the update column was linked to an external PHP file the href would pass the ID value of the row by appending it to the URL. The ID of the row in which the href was clicked would need to be retrieved with a GET action in the PHP file. I wanted to check this was working as intended. | The code would correctly work, and the job status would update. | The job status did not update when the href was clicked. | I was following a video tutorial on YouTube for this part of the programming. I watched the section of video again and realised I had not put delimiters within the code correctly. I was using double quotes in areas where single quotes should have been used. I made this alteration and the job status successfully updated. |
| 8 | Testing the Sorting CASE Logic Within the Job Retrieval SQL Statement | I had attempted to have the tech requests sorted on the technician portal with jobs with an incomplete status shown first. I had done this using CASE logic within the SQL statement. To check that this was working correctly I updated the job that was the first test ticket with the ID 1. If the sorting was working correctly this ticket would now be displayed after ticket two due to its complete status. | On page refresh handled by the job\_status\_update.php file the tickets displayed on the page would change to the specified order. | The sorting worked correctly and job one was displayed after page two on refresh. | No fix required. |
| 9 | Responsiveness of the Table Displayed in the tech\_requests.php file | As the previous pages had been developed with responsiveness in mind it was important to maintain this practice. Technicians may need to access job information whilst moving about the school on smaller screen size devices. To do this the developer tools within the browser would be used to emulate a smaller screen device. | The table would not be responsive as up until this point no code had been written to ensure this. | The table was elongated and overflowed the viewport width. | I did some online research on how to make a table responsive within html. I found a w3c schools’ tutorial that suggested placing the table inside a div with a style attribute applied that would enforce horizontal scrolling on the table if it exceeded viewport width. My table was already within its own container div, so all I needed to do was add this styling within the linked css file. With this code employed the table was now responsive. However, I noticed that particularly on the iPhone SE the issue description column was massively elongated making the table hard to read. I fixed this by adding an id within the echo for this row. Then calling its id in the css sheet and setting a minimum width for this td within a media query. |
| 10 | Testing the Custom Error Handler | For security reasons it is important with a website that is in live that runtime error messages are not displayed to end users. At times, they can contain sensitive information that should not be readily accessible. However, as developers it is important that we can still view these errors. This allows issues to be noticed and fixes provided. In the database.php I had programmed a protocol for errors using a function called customErrorHandler. The function changed error message syntax to a legible format along with displaying them in a txt file. To test this function, I inserted trigger\_error(“this is a test error”) in my PHP scripts. | I expected that an error would code along with the string inputted “this is a test error” would be displayed in the txt file in the syntax specified by the customErrorHandler function. | Errors were not being displayed within the txt file. | I had initially set\_error\_handler(customErrorHandler) within my database.php file. I had believed that because it was required by the PHP files this would work adequately. However, it clearly was not working. As all PHP files would have access to the function due to requiring the database.php file, I tried setting the error handler in each individual PHP file. This fixed the issue and the trigger\_error function was now displaying the correct string in the txt file. |
| 11 | Code Review | My familiarity with fundamentals and concepts within web development has improved since the assessment one submission. The main aim of this test would be refactoring areas of code to allow flex box layouts within the css. This was an aim that I had specified I wanted to achieve in the future during the first assessment task. Feeling more confident in my abilities it seemed logical to attempt this now. | I expected the html and corresponding css for the IT form submission page and its corresponding stylesheet would require alteration. | The code did need altering. | I rewrote the html and css code to allow flex containers and their corresponding flexboxes to be created. I then set up the new media query with established breakpoints to reflect these changes. |
| 12 | Testing Login Incorrect Details | I had specified within the conditionals for the log in.php file that if neither condition were satisfied the else code block would be run. This is supposed to display an alert using JavaScript ‘Invalid Username or Password’. I wanted to check that this was working correctly. | Upon entering incorrect details, I expected the message to be displayed. | The alert was not displayed. | Instead, I used header refresh and set a time value of five seconds. Along with this I provided an echo stating that the username or password was incorrect and instructing the user to wait to be redirected. This provided similar functionality and worked as intended. |
| 13 | Testing Login Staff User | Staff members that are not admin users should be redirected to the IT form submission page upon successful login. | I expected to enter the login details “staffmember” and “letmein!123” and then be redirected to the IT request form submission page upon a successful login. | Redirection to the appropriate page did not occur. | I had initially been using the password verify function to try and check the user inputted password against the password stored in the database. I could not get this to work and after reading PHP documentation and attempting to fix the issue for multiple hours I could not understand why. The only thing I thought it could potentially be is a salt being applied by the password\_verify function. I did not apply a salt to the password stored in the database. I decided to adopt a different approach. Instead, I hashed the users inputted password using SHA 256 the same as the database password. I then saved this to a variable named $hash. Then within my conditionals I simply checked if $hash was equal to the passwords stored in the database. With appropriate redirects employed dependent upon whether the staff member was an admin or not. |
| 14 | Testing Admin User Login | Staff members that are admin users who enter the correct credentials in line with project specifications ‘admin’ and ‘heretohelp!456’ should be redirected to a page containing tech requests . | The redirection would occur, and the tech requests page would be viewable. | Appropriate redirection did occur. | Redirection did occur largely due to the test prior to this one that highlighted errors in the code. No fix required. |
| 15 | W3C HTML Validation | The created pages did pass validation checks during assessment one. However, lots of changes have been made within the pages post assessment one submission. To maintain best practice, it is necessary to conduct these checks. | Pages should pass the validator with no issues. However, if issues are present these can be located and fixed. | There were a few errors within both html files. | They were a few trailing slashes that I corrected. A stray div closing tag within the login html file and a mismatched for attribute in the form submission file. I went through the w3c validator list and corrected these errors one by one. I then resubmitted both files and no errors were found. |
| 16 | W3C CSS Validation | I wanted to also ensure that the CSS for the pages passed the necessary validation checks using the W3C validation checker. | Pages should pass the validator with no issues. However, if issues are present these can be located and fixed. | All files apart from the tech\_requests\_style.css file passed the validator. | There were two errors due to syntax. The first was an invalid selector so I adjusted this. The second was a problem with repetition in which I had written the same line twice. I also corrected this and then resubmitted the file which validated on second attempt. |

A Critical Evaluation of the Helpdesk System Designed for WearView Academy

Considering some of the key strengths of the website, firstly, the site does function as intended and aligns directly with the specifications laid out within the project requirements. The employment of minimalistic design principles provides a level of professionalism expected by end users. Throughout the project, management of client expectations was considered, this was the reasoning behind attempting to provide accurate wireframes at the earliest opportunity. Ensuring that wireframes and design documentation is accurate is an important aspect of setting client expectations as supported by (Lakatos, 2023).

Secondly, another strength of the system, is the way in which it considers and deals with challenges presented by ensuring a secure user experience from a cybersecurity standpoint. When initially constructing the MySQL database, it was apparent that storing plain texts passwords within the database was not the correct procedure. This was established as part of the logical thought process within the design stage of the software development lifecycle (SDLC) (Fishpool, 2020). Storing plaintext passwords presents huge risks to end users a notion that is supported by Naiakshina et al. (2017). Due to many users often using the same username and password for multiple sites or applications allowing an attacker with access to the database and these details multiple opportunities of exploitation. Further reading on the subject and the ideas presented by Martin Paul (2016) solidified the importance of this aspect of system design.

Another important security consideration made, is the attempted prevention of SQL injection attacks. With any SQL database this is can be a considerable threat and means of malicious actions, such as unauthorised access or deletion of data. To counteract this the system uses an object-oriented approach. Prepared statements are used at every opportunity where interaction with the database is occurring, a methodology supported by (Bierer, 2013). Any user inputs are then attached as bound parameters that replace placeholder values. This greatly limits the likelihood of an SQL injection attack as the statements overall structure is not modified regardless of what the user enters.

In addition, due to various tutorials followed in construction of the site, the PHP filles within the http docs were initially coded in a procedural format. Further research was conducted, and this highlighted the readability and comprehension benefits of object-oriented programming comparative with procedural programming (Wiedenbeck et al., 1999). The decision was made to refactor the code for it to be written in an object-oriented style during the development stage of the SLDC. This aligned with the bottom-up approach that was required from this project and allowed code to be reused if necessary. This was particularly important for this project due to it only being a prototype, this would imply that further development would need to take place in the future.

The final strength of the system to be discussed is its adaptation as it developed, in the evaluation contained within assessment task one there was mention of researching and employing flexboxes within the css styling of the pages. This was implemented in this project to enforce an attitude of reflection and continued development. This aligns directly with the integration and testing stage of the SLDC. An opportunity to highlight potential flaws that can be improved upon.

However, it should be noted, there are various weaknesses presented by the system. From a security standpoint, the website does not use session cookies to properly login and logout users. From a cybersecurity perspective this opens the site to potential problems surrounding unauthorised access (Powers, 2021). Enforcing session cookies could be considered an opportunity for the development of the site also. Moreover, the site itself although functional could be more aesthetically pleasing. The use of background colours of a darker hue and white fonts may detract from readability for end users, the same could be said for the bold titles contained within the pages. In the future there is opportunity to make sites more aesthetically pleasing by being more ambitious with design documentation. This is not to say a minimalistic approach cannot be used, contrarily that minimalism should be maintained but with higher level elements besides just basic divs, paragraphs and forms. Furthermore, during assessment task one a requirement had been missed, the issue location field in the form was not included. I should pay closer attention to client requirements in future and implement strategies such as a checkbox list to ensure all necessary features are incorporated.

Aside from the design of the site itself there is also important considerations regarding the database connection methodology. This can be viewed as a potentially limiting factor for WearView Academy moving forward, since MySQLi connection only supports MySQL databases (W3schools, 2012). If the institutions data needs change in the future leading to an alternative database format the code will have to be rewritten in its entirety. This aspect massively detracts from the benefit of code reusability within object-oriented programming. Again, this can be viewed as a further opportunity to learn about PDO (PHP Data Objects) in database connection and consideration of implementing this methodology in future projects.

The Clientside validation is also another weakness within this project. In the future there is scope to include further JavaScript validation with specific functions for recognising email domains. There could be limits placed on the issue description characters within the IT form submission to prevent excessively long issue descriptions. Although with many of the inputs have required attributes attached the system could be improved by using jQuery (MDN Contributors, 2019). This would allow for selection of elements within the html document, custom functions could then be attached to these elements when triggered by an event listener. This is a methodology of Clientside validation that should be employed within future projects. The approach adopted within the IT form submission page could have been simplified and more legible with jQuery, this presents an opportunity to review and refactor this code soon.

The process of developing this system has instilled various research methods, provided learning prospects, and fostered a greater respect for, and larger understanding of web development. This project and module have fostered ambitions of establishing a career within full stack web development, acting as a starting point for development of my personal portfolio. However, it will be important as I continue to learn about web development to establish proper time management routines. I found myself committing long and extensive hours to this project and in the future, I will need to limit screen and work time to avoid the risk of over exertion. In maintaining an attitude of continued professional development this project will not end here. The code will continue to be refactored, reviewed, and developed until the website is entirely complete.

Reference List

Bierer, D. (2013). *PHP Security [Video]*. [online] www.oreilly.com. Available at: https://learning.oreilly.com/videos/php-security/9781771371636/9781771371636-video157006/.

Fishpool, B. and Fishpool, M. (2020) *Software Development in Practice*. 1st edn. Swindon: BCS Learning & Development Limited.

Lakatos, D. (2023). *Crafting Docs for Success*. Apress.

Eve, M.P. (2020) *Password*. London, England: Bloomsbury Academic. Available at: <https://doi.org/10.5040/9781501314902>.

MDN Contributors (2019). *What is JavaScript?* [online] MDN Web Docs. Available at: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript.

Naiakshina, A. *et al.* (2017) “Why Do Developers Get Password Storage Wrong? A Qualitative Usability Study,” *arXiv.org* [Preprint]. Available at: <https://doi.org/10.48550/arxiv.1708.08759>.

Powers, D. (2021). Authenticating Users with a Database. *Apress eBooks*, pp.537–550. doi:https://doi.org/10.1007/978-1-4842-7141-4\_19.

W3schools (2012). *PHP Connect to MySQL*. [online] W3schools.com. Available at: https://www.w3schools.com/php/php\_mysql\_connect.asp.

Wiedenbeck, S., Ramalingam, V., Sarasamma, S. and Corritore, C. (1999). A comparison of the comprehension of object-oriented and procedural programs by novice programmers. *Interacting with Computers*, 11(3), pp.255–282. doi:https://doi.org/10.1016/s0953-5438(98)00029-0.