**Evaluation Report – Tiani Perera**

**Working Functional Requirements**

|  |  |
| --- | --- |
| **Working Functional Requirements** | **Status** |
| **SYSTEM** | |
| Should have links and menus to navigate through the system | Working |
| Must have a staff log In page | Working with error detection |
| Must allow staff to login with their username and password | Working with error detection |
| Forgot Password feature works for log in | Working with error detection |
| The system must allow different staff roles : Manager and Sales Assistant | Working |
| Each role has their unique navigation links based on the requirements | Working |
| The system must have a stock system, a members control system along with sales system | Working |
| The stock system must have products of three categories: Computer Parts, Computer Systems and special offers | Working |
| When the stock levels for a product is below 10, the manager is alerted via email | Working |
| Stock level automatically updates itself when an order is processed | Working |
| Must have a stimulated payment feature that produces receipts | Working |
| 20% VAT must be added to all orders | Working |
| Must have a view basket page | Working |
| Must have a view products page | Working |
| Products can be added to the basket | Working with error detection |
| Products can be removed from the basket | Working with error detection |
| **ALL STAFF** | |
| View customer orders | Working |
| Create customer orders | Working with error detection |
| Register customer details | Working with error detection |
| Add customer information to the order | Working with error detection |
| View customer details | Working |
| View voucher codes | Working |
| Add unused voucher codes to a customer order | Working with error detection |
| View Stock Level | Working |
| Update Stock Level | Working with error detection |
| Edit their own details | Working with error detection |
| Change their own password | Working with error detection |
| Process a payment for a customer with the customers bank card, expiry date, account name and cvv | Working with error detection via stripe |
| Print receipts / View Receipt | Working |
| Customer Receipts must include all the necessary details | Working |
| **ONLY MANAGERS** | |
| Create New Staff and edit their details | Working with error detection |
| Delete Staff Accounts | Working with error detection |
| Change Staff Role | Working with error detection |
| Add new stock | Working without error detection |
| Amend stock | Working with error detection |
| Delete Stock | Working with error detection |
| Create new voucher codes – 15% off, 10% off and 5% off | Working with error detection |
| Send unique voucher codes to customers via email | Working |

**Non-Working Functional Requirements**

**SYSTEM:**

1. The system must have a feature that checks how many purchases a customer has made and sets their account with a permanent discount status (5% off for 30+ shops, 10% off for 60+ shops and VAT Free for 90+ shops) **– NOT ATTEMPTED**

I decided to leave this part of the system out as I was running out of time. I also had trouble with my voucher codes and how they were being added onto my order, I didn’t want to complicate it by adding a new discount system as I felt as If I would potentially ruin the voucher code system. If I had more time, this would be the main system that I would integrate next in my project.

1. Must allow staff to login with multi-factor authentication **– DESIGN CHANGE MADE IRRELEVANT**

Multifactor authentication was not implemented, I decided to take this feature out as it didn’t make sense in a shop system, as it would be time consuming to login. This wouldn’t be a feature I would necessarily add unless was required by the company

1. Customer email should be un-editable **– PARTIALLY WORKING**

This was an error validation mistake on my part. Due to inadequate testing before submission, I didn’t change this. When I try to change a customer email, it takes me to an error page. To fix this I would make this field a read only field, this would prevent from allowing a customer email to be changed and any error pages from being displayed.

1. An already used voucher code cannot be used again **– PARTIALLY WORKING**

When a voucher code is used, it sets it to “Used”, however if the code is removed and added on again without processing the order, it will still be used. I didn’t have time to fix this, on the day of submission I had noticed major errors with my basket page and voucher code, and while fixing these errors, I missed this. Fixing this would ensure that my voucher codes work in any situation.

1. Edit customer details with 2 factor authentication **– NOT ATTEMPTED**

This didn’t seem like a necessary feature to add. If I were to have more time and an account that would let me send OTPs, I would add 2fa for a phone number change only. This will allow phone numbers to be securely added/ changed for customers.

1. When a product in the basket is reduced in quantity on the basket page, it reduces below 0, and this decreases the total basket value **– PARTIALLY WORKING**

This error occurred due to insufficient testing on my part, this occurred because I ran out of time to thoroughly test the basket page before submission. This error is something that negatively impacts my shop, and would be something extremely important to fix. The alternate option to remove a product would be to hit the “X” button that is located at the start of the orderline

**Impact of the Non-working Requirements on Final Project vs Intended Application**

The discount system is the biggest feature that has not been added, this was a feature that was required by the company. Not having this system in place makes me feel like the application is incredibly incomplete, as I have not met the main conditions stated in the project brief. I never planned to get rid of this system, as it was quite an important feature. The other features such as making the customer email un-editable, fixing voucher codes, using 2FA for editing customer details, and the reducing quantity button on the basket page would have made system more robust and user-friendly and reduces the risk of any unwanted errors. In my opinion not having multifactor authentication for staff login has made the system better, as it is a time consuming process and is not ideal in a shop environment.

**Strengths & Weaknesses of the Outputs of the Practical Assignment**

Strengths:

* Having a Feature Rich System: I was determined to add as many features as possible to make my application more modern and user-friendly. effort was made to include numerous features, enhancing the modernity and user-friendliness of the application. This approach ensures that users have access to multiple features, and it makes my application more attractive to them. This also challenged me to learn new things and made me want to include even more useful features.
* User Feedback Integration: I actively sought out feedback from the people around me to ensure that my system is based on actual user needs. This approach allowed me to make changes to my user interactions and my application design, and in turn allowed me to continuously improve my applications while boosting user satisfaction.
* Having Detailed Documentation during my Practical Assignment: During the process I made it a habit to note down all the steps taken to complete the system. This helped me keep track of my work and made it easier for me to figure out which steps to take next. The documentation that I kept may be useful to me in the future, when building new applications/systems.

Weaknesses:

* Messy & Inconsistent Code: Halfway through the project I realised how I have redundant code. For example, with my site designs, there was one common site.css file, but I replicated the designs in individual files. I only realised this later on. Not realising this and not fixing it (due to a lack of time), is one of my project weaknesses.
* Inadequate Testing: Before submitting my project, I did not do proper error checking, which resulted in vulnerabilities. It also caused certain areas of my program to crash. The reason this is a weakness is because, this could potentially have led to further errors and vulnerabilities. Overall, this resulted in a less robust application.

**Modifications to the Project Plan, Solution Design, and Implementation**

* I didn’t stick to the exact dates of the project plan, this was due to several reasons, such as certain tasks being more time consuming due to errors (for example, I could not download my stripe.net package due to my project being saved on one drive. This caused a delay in my project) and certain features that I started earlier and finished faster than anticipated, such as manager functionality. Changing my personal deadlines allowed me to finish certain things quicker and give me more time to focus on other features.
* My testing stage started very late due to a lack of judgment, and this negatively influenced my assignment, as I could have resolved the errors and fixed any vulnerabilities.
* After designing my company logo, I re-designed it closer to the date of submission and redesigned most of my interface, including the colour scheme. I did not intend to do this, but redesigning my application made it more modern and aesthetically pleasing to look at.
* In my activity diagram and project plan, I had mentioned an account lock feature. This was not implemented because I seemed to have missed it. This feature would have increased the security and protection of my application.
* I did not plan to have two factor authentication when a staff member changes their phone number, I added this when trying to add my email system. This feature enhanced my application as it improved security and allows users to trust my application as there are extra steps taken for data protection.
* In my ERD my Order and Order line relationship was wrong, I realised this later on, My ERD states that my order and order line class have a many to one relationship, when it should be the other way around. This error that happened caused future confusion in the coding stage, which resulted in more time being taken.
* My design diagrams stated that my product class has an attribute called “category”, I took this out and created a “Category” class. This improved the data structure of my system and enhanced data integrity and consistency. Having a separate class also makes it easier to scale the category class in the future.
* I removed my PayPal class as I didn’t end up using this, this was due to me using stripe. Stripe allowed me to perform secure payments across multiple vendors. Adding stripe enhanced my application, and removing the class got rid of unnecessary code.

**Recommendations for future development of the project**

* Remove all the unnecessary classes that have not been used due to the addition of stripe. This will allow reduce the amount of storage space being taken up.
* Increase the security of my project, by implementing the “lock account” feature after 3 failed attempts, adding 2FA for editing customer phone numbers. This will also in turn give a more professional feel to the application.
* My application uses a drop create database, I would change this so that any orders that are placed will be saved upon closing the project. This would be useful when managers need to view reports.
* In my application I created a class for sending mail, when there was already an email service. I didn’t realise this, removing one of them would get rid of redundant code and reduce storage space.
* In the future, I would add better features to find customers or staff. This would include typing a part of their name/email and the system filtering out the rest. My current find username search bar only works if you type the full email address. Changing this would make the application more robust.
* I would have better structured emails. The emails I currently have consist of a line or two. Creating a template for it and adding the company logo would make it more user friendly.
* A main section I would add would be my discount system. This is important as it was a functional requirement in the project brief.
* Another feature I would add In the future would be more error handling code, this will ensure that my application can handle any situation without an error.
* A new system that I would like to add would be a system to accept returns and perform refunds via the method of payment. This would be an interesting challenge that will help me improve my project as well as my skill set.

**Skills Gained During Practical Assignment**

Planning Stage: This stage was one of the easier stages. Drafting interview questions, creating a project plan, and planning report helped me think about I were to proceed with my project. And I truly understood the importance of this stage when I moved on to coding my program. The main thing I learned from this was how important it is, and how it affects the rest of the project. Additionally, another thing I learned was how I could have asked better interview questions by broadening my view (e.g.: questions such as would you like a returning system). Thinking out of the box and coming up with new ideas will help me in the future when I need to plan certain projects.

Design Stage: This stage felt like the toughest, as I wasn’t confident in building designs for an application I have not created. Doing this and moving on to my coding stage did help me. However, the main thing I learned at the end of my journey, was how I could have done this better and how I could have been more accurate.

Coding Stage: I learned a lot during this stage. I am not good at designing from scratch, and creating a logo and a colour scheme was a challenge for me. After having to edit the html of my program to change the look of my application, I learned more on how it works and how to design things to make it more user friendly. At first, I thought it was quite hard, but when I started working on it more, I realised how I had certain inconsistencies and I became more confident, now I can say that I can easily design something with html. This stage also helped me understand all the changes I had to make to the design stage to make my application work. When it comes to the coding, I realised that there were easier ways to do things. For example, using dictionaries. The more I coded, the more I figured out different ways to make features work. Keeping a Diary on what I coded also helped me and is something I will consistently be doing in the future. During this stage I added extra features into my application, this was really interesting and fun to do and made me want to find more features and add it as well.

Testing Stage: One of the main things I learned during this stage was that I should have done this before submitting my coding stage. This would have helped me avoid the errors of my application. The testing stage helped me identify different ways I could do certain processes in my application.

Overall, following a methodical lifecycle, helped me in achieving the best that I could. It has made me more confident that I could start over and do things even better with the amount I have learnt from this. In the future I would probably hope to put in more effort into the designing stage so I can fully understand the requirements of my project. Additionally, I would start testing earlier and repeatedly alternate between the coding and testing stage more than I did.