# PROJECT INTRODUCTION:

## Team Information:

**Team Number:** 25

**Title of The Project:** E-Commerce Website (PC and Laptop Selling Website) **Team Members:**

* Aaron Wills (210155800)
* Faran Zafar (20036414)
* Kirill Ushakov (220036159)
* Zixu Wang (200105886)
* Hassan Ali (200132734)
* Gbanet Ibrahim (210187269)

## Project Brief:

**Title: E-Commerce Website (PC and Laptop Selling Website)**

The task is to make an e-commerce platform to sell products. Name, vision, and product line must be decided by the team. It has been decided that the website will sell computer related products and the given name is “CompUK”.

According to Project Brief 1, the solution should be built with PHP and a database (MySQL is recommended to be used), and the solution should be hosted, Heroku can be used for thar purpose.

For the backend of the website PHP is going to be used, and for the frontend HTML, CSS, SCSS and JS.

For a team management a Trello is going to implemented and we are going to use GitHub to keep the repository of website.

## Team Contribution:

**Aaron Wills:** Input my product knowledge and background. Selecting a range of products that would best suit our business. As well as grouping the products together and managing product details (ie. product description, price, etc.)

**Faran Zafar:** Take Lead o Front End programming. Delegate tasks regarding storyboarding, CSS box model and documenting the process of the design section. I would be going over intricate tasks related to building the design and dynamic functioning of the website from the ground up using JavaScript, HTML,CSS and SCSS. Meanwhile, I’ll be also looking over and making sure that correct documentation structure is followed.

**Kirill Ushakov**: Take lead on back-end programming. Delegate tasks regarding processes related to flow data and databases as well as the overall approach. To illustrate and plan the concepts flowcharts, schemas, use cases, process diagrams are used.

**Zixu Wang:** Maintain Team cohesion to help team function. Make sure the tasks that are split between everyone function together. Quality control on the overall program.

**Hassan Ali:** Help the team stay organised and on track. Organise meetings, write up meeting notes and keep team on track regarding the original plan.

**Gbanet Ibrahim:** Will work on the connection between HTML and PHP. Work alongside Kirill with backend and focus mainly on hosting the website.

# REQUIREMENT ANALYSIS:

## Requirements Investigation:

### Functional Requirements:

|  |  |  |
| --- | --- | --- |
| № | Definition | MoSCoW |
| 1 | The website must have pages: | M |
| 1.1 | A home page | M |
| 1.2 | A page to display the products that the company sells | M |
| 1.3 | A page to display the user’s current basket | M |
| 1.4 | An “About Us” page (or similarly named) to state the vision of the business | M |
| 1.5 | A page with contact details for the business | M |
| 1.6 | A page to sign up to use the website, or log in if they already have an account | M |
| 2.1 | The website must be deployed, and must always be in a functional state | M |
| 3.1 | The company must have a brand | M |
| 3.1a | Logo | M |
| 4.1 | There must be at least two types of user | M |
| 4.1.a | An admin | M |
| 4.1.b | A customer | M |
| 4.2.a | An admin must be able to see a list of customers | C |
| 4.2.b | An admin must be able to see a list of currently placed orders | M |
| 4.2.c | An admin must be able to see a list of products that are stocked | S |
| 4.2.d | An admin should be able to allow new administrator accounts to be created | S |
| 4.2.e | An admin should have access to the admin login page | S |
| 4.2.f | An admin should be able to add new products | S |
| 4.2.g | An admin should be able to edit the products | S |
| 4.2.h | An admin should have access to accounts’ details | S |
| 4.3.b | Customers must be able to place an order, which submits their basket and registers it as an order in the database along with the total price | M |
| 4.3.c | Customers must be able to view their past orders | M |
| 4.3.d | Customers must be able to view a status of their orders | M |
| 4.3.e | Customers should be able to search for items | S |
| 4.3.f | Potential user should be able to create a new account | S |
| 4.3.g | Customers should have access to user login page | S |
| 4.3.h | Customers should be able to edit their accounts | S |
| 4.3.i | Customers should be able to put products in the basket | S |
| 4.3.j | Customers should be able to remove products from the basket | S |
| 5.1 | The web application must be deployed in a working state and accessible at all times | M |

Table .: Functional requirements

### Non-Functional Requirements:

|  |  |  |
| --- | --- | --- |
| № | Definition | MoSCoW |
| 1.7.a | Pages load within 5 seconds. | C |
| 1.7.b | Intuitive navigation. | S |
| 1.7.c | Compatibility of website on all devices. | W |
| 2.1.a | Secure service (security of personal details). | S |
| 3.1b | Colour scheme | S |
| 3.1c | Consistent font | S |
| 4.3.a | Customers must be able to access their basket at any time | M |
| 5.1.a | Has to holds 200 users at any given time. By using a sandbox program that simulates users. | W |
| 6.1 | The code for the website must also be professionally managed with a version control system GitHub/GitLab | M |
| 6.2 | Its use must be documented in a README file for accessibility and maintainability | M |

Table .: Non-functional requirements

## Target Audience/ User Identification:

In this section the table with user identification is presented:

|  |  |
| --- | --- |
| Parameter | Identification |
| Age | 18+ |
| Gender | No gender focus |
| Region | The website is going to be made for the UK |
| Language | English only |
| Purpose | Sell computer related products |

Table .: User identification

## Use Cases:

|  |  |
| --- | --- |
| Name | Sign Up System (SUS) |
| Breath Description of the use case | This use case describes the sign-up system as a part of the online website for a user to create a new account as a customer or admin. |
| Actor | Customer, Administration, Database |
| Main success scenario | 1)User puts all the required details accordingly to set format.  2)The application checks if the database contains the account with the same email.  2.1) If the entered email has not been used accept and create an account with “Customer” status.  2.2) If has the entered email has not been used and admin registration is requested, create an account with “Potential Admin” status.  3)Show a message about:  3.1) Successful registration (As a customer)  3.2) Successful application (As an admin) |
| Alternative flows | 2.1) If entered email has been already used and found in the database, show a message about that  2.2) Suggest to:  2.2.a) Log in  2.2.b) Use a different email  2.2.c) Contact Administration |
| Pre-conditions | 1)Sign-up page is opened  2)User has an email address |
| Post-conditions | 1)Home page opens |

Table .: Sign Up System (SUS)

|  |  |
| --- | --- |
| Name | Check out basket |
| Breath Description of the use case | This use case describes the check-out system, for user to process the order. |
| Actor | Customer, Administration, Database |
| Main success scenario | 1)The application checks if basket is empty  2)The applications check if the product in the basket is available or not  3)If all conditions listed above are true, the application will be checked by administration to procedure it. |
| Alternative flows | 2)If the basket is empty the “Check out” button will not be available  3.1)If something from the list is out of stock the “Check out” button will not be available.  3.2)If something goes wrong show an error message |
| Pre-conditions | 1)User is signed in  2)The basket page is opened |
| Post-conditions | 1)The applications shows that the heck account has been successful |

Table .: Check out basket

|  |  |
| --- | --- |
| Name | Printing out products/categories |
| Breath Description of the use case | This use case describes how the application will show products/categories to users. |
| Actor | Database, application |
| Main success scenario | 1)Applications connects to the database  2)Applications reads the contents of the table(Categories/Products)  3.1)If categories page is opened, print all of the categories  3.2a)If a category page is opened, print all of the products related to the category  3.2b)If product is out of stock print the picture “Out of stock”. |
| Alternative flows | 2)If connection has failed, show an error message |
| Pre-conditions | 1)Database needs to contain some products/categories  2)The products/categories page needs to be opened |
| Post-conditions | 1)The products/categories are printed |

Table .: Printing out products/categories

|  |  |
| --- | --- |
| Name | Edit a product |
| Breath Description of the use case | This use case is about editing the product details that is stored in the database through the online website. Provided the admin know the PID of the product that needs to be assessed |
| Actor | Admin, Database |
| Success Scenario | 1)The admin searches the product using either the product name or product ID  2)The admin then clicks on the edit button.  3)This action makes all the fields of the product details editable allowing the admin to edit the details of the product |
| Alternative flows | If the product admin is looking for or if the admin is not able to edit the details of the product:  2)The admin gets shown with a pop up that the product you are looking for doesn’t exist or the changes you have made to the product aren’t saved.  3)The admin gets redirected to the product finding page. |
| Pre-conditions | 1)Product searching page  2)A signed in admin  3)A product with unique PID |
| Post-conditions | 1)A confirmation message of product deletion is sent  2)The Product searching page opens up |

Table .: Edit a product

|  |  |
| --- | --- |
| Name | Add new product |
| Description of Use Case | The use case of Adding new product describes the functioning of the online website to add new products to the database as well as put them on display for the users. |
| Actor | Admin, Database, Customers |
| Success Scenario | 1)The admin puts in all the details of the product.  2)The application checks the product ID of the product to check if it’s already in the database.  3)If the application finds that there is no product with the specific id it gets added into the database  4)The added product gets displayed on the website which can be assessed by the customer as well |
| Alternative flows | If the product to be added is already present in the database or there is an error adding the product:  3)The admin gets redirected to the product adding page with a pop up “Product wasn’t added into the database”. The product details are not added into the database and customers won’t be able to see the added product which doesn’t effect the functioning of website in anyway |
| Pre-conditions | 1)Product searching page  2)A signed in admin  3)A product with unique PID |
| Post-conditions | 1)A confirmation message is sent  2)The Product page opens up |

Table .: Add new product

|  |  |
| --- | --- |
| Name | Sign in as admin |
| Description of Use Case | Admin signs in to access and control certain aspects of the website, such as, accessing and editing products, accepting new admin registrations, etc. Admins login on admin login page. |
| Actor | Admin, database |
| Success Scenario | 1)Admin inputs required details in correct format.  2)The application checks the database if correct username and password has been entered, and if the account has the rights.  3)If correct username and password then admin successfully logs in.  4)Successful login message shown |
| Alternative flows | 2)If entered details are incorrect  3)Show the wrong password or username message  2)If the account does not have rights.  3)Open a Sign def Page. |
| Pre-conditions | 1)Created admin account is required  2)Sign in page is open |
| Post-conditions | 1)Admin panel opens |

Table .: Sign in as admin

|  |  |
| --- | --- |
| Name | Sign in def |
| Description of Use Case | The use case is for a user signing into our website, so that they can gain access to exclusive parts only account holders can take advantage of. |
| Actor | User, database |
| Success Scenario | 1) User inputs required details in correct format.  2)The application checks the database if correct username and password has been entered.  3)If correct username and password then user successfully logs in.  4)Successful login message shown |
| Alternative flows | 2)If entered details are incorrect  3)Show the wrong password or username message |
| Pre-conditions | 1)Created account is required  2)Sign in page is open |
| Post-conditions | 1)User page opens |

Table .: Sign in def

# PROJECT MANAGEMENT/PLANNING: