**Plant Oasis – Final Documentation**



Daniel Cullen – x00168408

Hubert Bukowski – x00161897

Jibril Abdulazeez – x00164441

Supervisor - Patricia Magee

Student Compliance with Academic Integrity

All students are expected to complete their courses in compliance with University regulations and standards. No student shall engage in any activity that involves attempting to receive a grade by means other than honest effort, for example:

1. No student shall complete, in part or in total, any examination or assignment for another person.
2. No student shall knowingly allow any examination or assignment to be completed, in part or in total, for himself or herself by another person.
3. No student shall plagiarize or copy the work of another and submit it as his or her own work.
4. No student shall employ aids excluded by the instructor in undertaking course work.
5. No student shall knowingly procure, provide, or accept any materials that contain questions or answers to any examination or assignment to be given at a subsequent time.
6. No student shall procure or accept assignments from any other student from current or prior classes of this course.
7. No student shall provide their assignments, in part or in total, to any other student in current or future classes of this course.
8. No student shall submit substantially the same material in more than one course without prior authorization.
9. No student shall alter graded assignments or examinations and then resubmit them for regrading.
10. All programming code and documentation submitted for evaluation or existing inside the students computer accounts must be the students original work or material specifically authorized by the instructor.
11. Collaborating with other students to develop, complete or correct course work is limited to activities explicitly authorized by the instructor.
12. For all group assignments, each member of the group is responsible for the academic integrity of the entire submission.

**N.B to be filled out by each member of the team.**

By including my name in the form below, I declare that I understand and will abide by the University Regulations and Policies covering Academic Integrity. I accept that each member of the group is responsible for the academic integrity of the entire submission. I will retain a copy of this agreement for future reference.

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name (1):** | **Daniel Cullen** | **Student No.:** | **x00168408** |
| **Student Name (2):** | **Hubert Bukowski** | **Student No.:** | **X00161897** |
| **Student Name (3):** | **Jibril Abdulazeez** | **Student No.:** | **X00164441** |
| **Student Name (4):** |  | **Student No.:** |  |
| **Module title:** | **Year 2 Project** | | |
| **Programme Title:** | **Computing with Software Development** | | |
| **Date: DD/MM/YYYY** | **24/04/2021** |

**Responsibilities:**

**Daniel Cullen:** Sign Up, Log In, Log Out, Profiles, Manager, Blogs, Comments, Likes/Dislikes, Discount Code, Blogs Crud, Profiles Crud

**Hubert Bukowski:** Products, Categories, Subcategories, Product CRUD, Reviews, Reviews CRUD, Wishlist, Home styling, Cart styling

**Jibril Abdulazeez:** Search app, Order app, Checkout, Stripe payment system, Cart, Pagination

**Table Of Contents:**

1….. Introduction ………………………………………………………………………………………………………….. 5

2….. Plant Store Design …………………………………………………………………………………………………. 6

3….. Use Cases ………………………………………………………………………………………………………………. 6

4….. Use Case & Class Diagrams ……………………………………………………………………………………. 16

5….. Iteration One Report …………………………………………………………………………………………….. 17

6….. Iteration One Test Case Matrix ……………………………………………………………………………… 22

7….. Iteration Two Report …………………………………………………………………………………………….. 26

8….. Iteration Two Test Case Matrix ……………………………………………………………………………… 34

9….. Iteration Three Report …………………………………………………………………………………………… 43

10… Iteration Three Text Case Matrix ……………………………………………………………………………. 45

11… Performance Tests …………………………………………………………………………………………………. 46

12… Conclusion ……………………………………………………………………………………………………………… 49

13… References ……………………………………………………………………………………………………………… 50

**Introduction:**

As a group we decided to create an ecommerce website by the name of Plant Oasis for our 2nd Year Project. We came up with the idea of doing a plant store as Hubert had a lot of plants in his house and Daniel had a small desk plant that he had been caring for during quarantine. The name Plant Oasis was not decided on until the workings of iteration 1 of the project after the design phase.

We believed that a plant store would allow us to implement other features into the website such as the blogs as this was a feature we had seen on many other plant focused websites that we researched for inspiration. The subject of plants also gave us the opportunity of using subcategories as many plants come from different species of plants.

It was apparent throughout each iteration that we were challenging ourselves to implement new features and make the website as user friendly as possible. Due to being stuck at home during quarantine we received a lot of input from our families on how to improve the user interface of our website. It was a struggle not being able to work together face to face, but we took advantage of an application called discord throughout the iterations to plan and help each other when we came across issues. This was very helpful as we could display our screens to the other members and get their input on the issues or get their ideas on the implementation of a new function.

The design of the website was inspired by a combination of many websites that we researched for our project and it came down to using a white background with green, yellow, and brown accents which was pleasing to the eye and easy to navigate. We used Selenium IDE and test cases to do the testing for each feature that we implemented into the website.

Plant Store Design – by Daniel Cullen, Hubert Bukowski and Jibril Abdulazeez.

Goals:

We plan on creating a user-friendly online plant store that gives the user the ability to purchase plants, pots and tools at an affordable and competitive price. In addition to this we plan on providing our uses with an intuitive blog area that they can use to exchange knowledge and queries that they may have with the rest of the community about plants. We also plan on receiving feedback for our users by allowing them to leave reviews and ratings on our many products. We would like to send an email to our users containing new information about plants, we will do this by creating an email subscription box that the user will access on the homepage. We acknowledge that not all our products are on the affordable side so we will attempt to make a wish list that users will be able to add a product to, so they do not forget about it until they have the funds for it. The website will be designed for professional and newcomer gardeners alike as we will have a range of products for all kinds of gardeners. The website will be easy to navigate for people with any amount of knowledge about plants. The main functionality of the website is to allow users to purchase gardening products and to also gain knowledge and ask questions about the subject.

Feasibility:

We believe that the project is feasible in the timeframe that we have been given. We have had a fair amount of exposure to creating an ecommerce website already, so we are quite confident in our abilities to create the core functionality of the project. We plan on adding some new functionalities that we have not explored in class yet so we do not know how smoothly we will be able to execute them. Due to this we believe we will come across some difficulties, but we have found many websites that we will refer to during the creation of the new functionalities. The risks that we may come across are that the new functions that we would like to add may not function correctly, we will either must scrap thew function or attempt to design it in another way for it to work on the website if this happens. Another risk we might come across is that we may not have the time to test all new functionality if we spend too much time researching on how to implement them into the website. We may have to decide on when we must stop attempting to implement a new function due to time constraints. We plan on deploying our website on two different host websites in the fear that one may crash, this will allow us to keep the website online in that circumstance.

Use Cases:

|  |  |
| --- | --- |
| **Use Case:** | Create Account (Priority 1) |
| **Actor (s):** | Unregistered User |
| **Goal:** | To create an account that can be used on the store website. |
| **Overview:** | The unregistered user should see a 'sign up' button on the home page. Once pressed the unregistered user will be prompted to provide their name, age, Email address and password. Once all details are entered the user will be able to click sign-up and will have an account for the store. |
| **Pre-Condition:** | The current user accessing the system must be of type unregistered user. |
| **Post Condition:** | The unregistered user will become a registered user and will receive an account. |
| **Successful Scenario:** | 1. User clicks the sign-up button. 2. User is redirected to the sign-up page. 3. User inputs their name. 4. User inputs their age 5. User inputs their email address. 6. User inputs their password. 7. User input their password again. 8. User clicks the sign-up button. 9. User is redirected to the login page. 10. User signs in using their account details 11. User is redirected to the homepage and is signed into their account. |

|  |  |
| --- | --- |
| **Alternative Scenario (s)** | Alt 3. User inputs name longer than 30 characters which will prompt the user to input a shorter name, must not exceed 30 characters.        Alt 4. User inputs a negative value for their age, the user will be prompted to input a value higher than 0.        Alt 5. User inputs an invalid email address, the user will be prompted to input a valid email address that contains '@email.com'.        Alt 6. User inputs an invalid password, user will be prompted to enter a password that exceeds 8 characters, contains a capital letter and a number.        Alt 7. User re-enters the password incorrectly, the user will be prompted to re-enter the password correctly.        Alt 10. User inputs the incorrect account details, the user will be prompted that the name or password are incorrect and must be input again. |

|  |  |
| --- | --- |
| **Use Case:** | Manage Account (Priority 2) |
| **Actor (s):** | Registered User, Manager |
| **Goal:** | To edit the details of a user’s account. |
| **Overview:** | The registered user should see an 'edit' button on their account page. Once pressed the registered user will have the ability to alter the details of their account. Once all details are entered the user will be able to click the save button which will save their new details. The user will also be able to click a delete button to delete their account. The manager also has access to this button if they wish to delete the account. |
| **Pre-Condition:** | The current user accessing the system must be of type registered user or a manager. |
| **Post Condition:** | The registered user will have altered the details of their account or the account will have been deleted. |
| **Successful Scenario:** | Registered User:   1. User clicks the edit profile button. 2. User is redirected to the edit profile page. 3. User inputs their new name 4. User inputs their new age. 5. User inputs their new email address. 6. User inputs their new password. 7. User input their new password again. 8. User inputs new profile image. 9. User clicks the save button. 10. User is redirected to their profile. 11. User clicks delete button. 12. User is redirected to the homepage and their account is deleted.   Manager:   1. Manager clicks manage profiles button. 2. Manager is redirected to manage profiles page. 3. Manager clicks on user profile. 4. Manager is redirected to user profile. 5. Manager clicks delete button. 6. Manager is redirected to manage profiles page. 7. User profile is deleted. |

|  |  |
| --- | --- |
| **Alternative Scenario (s)** | Alt 3. User inputs name longer than 30 characters which will prompt the user to input a shorter name, must not exceed 30 characters.        Alt 4. User inputs a negative value for their age, the user will be prompted to input a value higher than 0.        Alt 5. User inputs an invalid email address, the user will be prompted to input a valid email address that contains '@email.com'.        Alt 6. User inputs an invalid password, user will be prompted to enter a password that exceeds 8 characters, contains a capital letter and a number.        Alt 7. User re-enters the password incorrectly, the user will be prompted to re-enter the password correctly.        Alt 8. User attempts to use an unsupported file for their profile image, the user will be prompted to use the correct file. |

|  |  |
| --- | --- |
| **Use Case:** | Create Blog (Priority 8) |
| **Actor (s):** | Registered User |
| **Goal:** | To create a blog that can be seen by other registered users. |
| **Overview:** | The registered user should see a 'create' button on the blog page. Once pressed the registered user will be prompted to provide the title, topic and image relating to the blog. The user will also be prompted to write the contents of their blog. Once all details are entered the user will be able to click create which will create the blog. |
| **Pre-Condition:** | The current user accessing the system must be of type registered user. |
| **Post Condition:** | The registered user will have a blog created. |
| **Successful Scenario:** | 1. User clicks the create button. 2. User is redirected to blog creation page. 3. User inputs their title. 4. User inputs their topic 5. User inputs blog image. 6. User inputs the blog contents. 7. User clicks the create button. 8. User is redirected to the blog page. 9. User clicks their blog on the blog page. 10. User is redirected to their blog. |

|  |  |
| --- | --- |
| **Alternative Scenario (s)** | Alt 3. User inputs title longer than 100 characters which will prompt the user to input a shorter title, must not exceed 100 characters.        Alt 4. User inputs a topic longer than 20 characters which will prompt the user to input a shorter topic, must not exceed 20 characters.        Alt 5. User inputs an incorrect media file for the image which will prompt them to use the correct media file.        Alt 6. User inputs content that exceeds 300 characters, which will prompt the user that the contents can only be 300 characters long.       Alt 7. If the user does not fill out all the mandatory parts of the blog then they will be prompted to complete the blog before it can be created. |

|  |  |
| --- | --- |
| **Use Case:** | Manage Blog (Priority 9) |
| **Actor (s):** | Registered User, Manager |
| **Goal:** | Allow a registered user or a manager to edit a blog. |
| **Overview:** | The registered user should see an 'edit' button on their own blog. Once pressed the registered user will be prompted to edit the title, topic and image relating to the blog. The user will also be capable of editing the contents of their blog or just updating it. Once all details are entered the user will be able to click save which will save the edited blog. The registered user or a manager will also be able to delete the blog. |
| **Pre-Condition:** | The current user accessing the system must be of type registered user or a manager. |
| **Post Condition:** | The registered user will have an updated blog or a deleted blog. |
| **Successful Scenario:** | Registered User:   1. User clicks the edit button. 2. User is redirected to edit blog page. 3. User inputs their updated title. 4. User inputs their updated topic 5. User inputs an updated blog image. 6. User inputs the updated blog contents. 7. User clicks the save button. 8. User is redirected to the blog page. 9. User clicks their blog on the blog page. 10. User is redirected to their updated blog. 11. User clicks the delete button. 12. User is redirected to the blog page. 13. The user's blog is deleted.   Manager:   1. Manager clicks the blog. 2. Manager is redirected to the blog. 3. Manager clicks the edit button. 4. Manager is redirected to edit blog page. 5. Manager clicks the delete button. 6. Manager is redirected to the blogs page. 7. The blog has been deleted. |

|  |  |
| --- | --- |
| **Alternative Scenario (s)** | Alt 3. User inputs title longer than 100 characters which will prompt the user to input a shorter title, must not exceed 100 characters.        Alt 4. User inputs a topic longer than 20 characters which will prompt the user to input a shorter topic, must not exceed 20 characters.        Alt 5. User inputs an incorrect media file for the image which will prompt them to use the correct media file.        Alt 6. User inputs content that exceeds 300 characters, which will prompt the user that the contents can only be 300 characters long.       Alt 7.If the user does not fill out all the mandatory parts of the blog then they will be prompted to complete the blog before it can be saved. |

|  |  |
| --- | --- |
| **Use Case:** | View Products (Priority 3) |
| **Actor (s):** | Registered User/Unregistered User |
| **Goal:** | To search for a product within the store inventory and be able to filter their product search further by different assortments. |
| **Overview:** | The website will have an accessible search bar where users can text in a product to search for and filter their search further by category, price, relevance, popularity, alphabetically and number on page. |
| **Pre-Condition:** | The user must be on the webpage with a stable connection |
| **Post Condition:** | A new webpage for the searched item whether filtered or not is generated |
| **Successful Scenario:** | 1. User logs into their account and is redirected to home page 2. User inputs the product name into the search bar 3. User filters their product search further by an available assortment. 4. A suggested product drop-down list appears in accordance to what the user has texted in 5. User confirms search. 6. User is redirected to the newly loaded product webpage |
| **Alternative Scenario (s)** | Alt 1. User is unregistered/logged out and proceeds as such.        Alt 5. User incorrectly texts in the product name and no result alongside an error webpage is returned.   |  | | --- | | Alt 3. The searched item cannot be filtered by the user choice so a result of “not available" is returned. | |

|  |  |
| --- | --- |
| **Use Case:** | Create Review (Priority 10) |
| **Actor (s):** | Registered User, Manager/Administrator |
| **Goal:** | To review and rate an item purchased in the store |
| **Overview:** | A registered user should be able to leave a review and rating on the items they purchased through the product page or their purchase history. If a user has purchased an item and go into their purchase history, they should see a ‘leave review’ button next to items they purchased. Once pressed the user is redirected to a review page, where they are prompted to leave a message review of max 256. Characters, any media of the item and a rating between 1 and 5. Once they are happy with their review, they submit it, and the review is displayed on the product page. If the user changes their opinion, edit and delete buttons will be available to press next to the user’s reviews. When a user chooses to edit their review, they will be able to adjust the review message, rating and media. The system will scan for profanity in the message and will not allow a review to be posted, but in the case of system error, the website manager/administrator will be allowed to delete and null reviews and ratings. |
| **Pre-Condition:** | A user has registered on the website and they have purchased a product from the store. |
| **Post Condition:** | The review is successfully posted and is displayed on the product page, the rating is taken, and an average is calculated between other ratings left on that product. The rating is displayed appropriately on the product page. |
| **Successful Scenario:** | Registered User   1. The user is logged in and has products previously purchased. 2. The user accesses the product page and clicks ‘leave review’ button. 3. User gets redirected to the ‘leave review’ page. 4. User chooses a rating between 1 and 5. 5. User enters their review message. 6. User attaches a photo of their item. 7. User clicks the submit button. 8. User is redirected back to the product page and their review is displayed appropriately. 9. User clicks ‘edit review’ button. 10. User changes review message 11. User changes rating between 1 and 5 12. User deletes old media. 13. User adds new media. 14. User submits updated review. 15. User is redirected back to the product page and their review is updated appropriately. 16. User clicks ‘delete review’ button. 17. User is prompted to confirm their choice. 18. If user confirms, the page is adjusted appropriately, and their review is no longer displayed.   Manager/Administrator   1. User accesses the admin page using a button located in the navigation bar 2. User selects they would like to view all reviews 3. User finds an inappropriate review 4. User presses the ‘delete’ button located next  to each review 5. System asks user for confirmation 6. User confirms and the list is changed appropriately |
| **Alternative Scenario (s)** | Alt User 2. User did not purchase that item and leave review button won’t appear  Alt User 5. Message is longer than 256. Characters, User will be asked to change their message  Alt User 5. Message contains banned words, User will be asked to change their message  Alt User 6. User media is of unsupported format  Alt User 8. One or more values entered are incorrect, pressing the confirmation button will display a message to the user informing them of their errors  Alt User 9. User hasn’t reviewed that item yet and their review, along with the button won’t appear  Alt User 10. Message is longer than 256. Characters, user will be asked to change their message  Alt User 11. Message contains banned words, User will be asked to change their message  Alt User 13. User media is of unsupported format  Alt User 14. One or more values entered are incorrect, pressing the confirmation button will display a message to the user informing them of their errors  Alt User 18. User changes their mind and clicks ‘cancel’, the review stays as it was    Alt Admin 1. User is not of type ‘admin’ and the button doesn’t appear  Alt Admin 6. Admin changes their mind and clicks ‘cancel’, the review stays as it was |

|  |  |
| --- | --- |
| **Use Case:** | Manage Cart (Priority 4) |
| **Actor (s):** | Registered User/Unregistered User |
| **Goal:** | The user can manage their cart to match their needs |
| **Overview:** | The user can prepare their ordering process by removing products from their cart, editing product quantity or moving products out of the cart to their Wishlist. |
| **Pre-Condition:** | The user must have at least one product in their cart |
| **Post Condition:** | None |
| **Successful Scenario:** | 1. User logs into their registered account before adding a product(s) to cart 2. The user accesses the cart by clicking the cart icon 3. The user is able to remove multiple product(s) from the cart 4. The user can edit the quantity of multiple products in the cart 5. The user can redirect the product(s) to their Wishlist for later purchase or add the product(s) to their Wishlist to buy again later 6. The order total updates instantly to reflect the users' changes |
| **Alternative Scenario (s)** | Alt 1. User does not log in and proceeds as such    Alt 5. As the user is not logged in, the user will be prompted to register if they are new or log into an existing registered account to permanently save their wish list. |

|  |  |
| --- | --- |
| **Use Case:** | **Make Payment** (Priority 5) |
| **Actor (s):** | **Registered User, Unregistered User, Payment System** |
| **Goal:** | **User successfully can purchase a product(s).** |
| **Overview:** | **After adding a product to cart and proceeding with checkout, the user is prompted to enter in their payment, shipping and billing address details which is then authorized, checked and verified by the payment system** |
| **Pre-Condition:** | **User has proceeded to checkout with their payment and address details readily available**  **The shopping cart contains at least one item** |
| **Post Condition:** | **User has purchased a product(s) to be delivered to their denoted address and has received a order confirmation email** |
| **Successful Scenario:** | **Registered User:**   1. **User creates the online order and is returned an order total and the list of products within their order alongside their respective quantities** 2. **User is prompted to enter in a discount code if applicable** 3. **User is forward to a shipping and billing address webpage**   **3.1 If the registered user has pre-emptively entered in a adequate shipping and billing address, an option will be made available to auto-fill in the details        3.2 User manually enters or auto-fills in their shipping and billing address details, fulfilling all the fields labelled with a red asterisk        4. User is forwarded to a shipping options webpage       4.1 The system automatically calculates the shipping costs based off the users' shipping address and provides the user with options of express delivery       5. User is forwarded to the billing options web-page with an order total        5.1 The registered user is prompted to auto-fill their payment and personal details or used a saved payment choice if applicable and saved to the registered user account       5.2 User manually enters or auto-fills in their payment and personal details, fulfilling all the fields labelled with a red asterisk       6. User confirms their order  Unregistered User:**   1. **User creates the online order and is returned an order total and the list of products within their order alongside their respective quantities** 2. **User is prompted to please sign in if they are a logged out registered user** 3. **User is given the choice to "Checkout as guest" granted they provide a email address for purchasing** 4. **User is given the option to register before checkout and use the newly registered account for checkout** 5. **User is forwarded to a billing and shipping address web-page** 6. **User proceeds with making payment normally, manually entering any details that must be provided** 7. **User confirms their order**   **Payment System**   1. **The payment system authorizes and captures the users' payment info.** 2. **The payment system check if the payment info provided is void** 3. **The payment system verifies the users' payment info** |
| **Alternative Scenario (s)** | Registered User:  Alt 2. Discount code is invalid and is unable to be applied to the order  Alt 3.2 User entered details are incorrect or invalid and the user is unable to proceed until these details are correct and all red asterisk labelled fields are filled  Alt 5.2 The user payment details are not recognized or are incorrect and the user is unable to proceed with payment until the issue(s) is corrected  Unregistered User:  Alt 2. User attempts sign in with an unregistered account which throws back an error informing the user the details entered are incorrect and not recognized  Alt 3. User attempts to checkout as guest without providing a valid email address, halting the user from proceeding until a valid email address is entered, highlighting the empty field in red  Alt 6. User manually enters incorrect details and is unable to proceed from their current step until the issues are rectified  Payment System:  Alt 1. Payment info is void and the user is redirected to the billing options page to re-check or re-enter their payment details and make sure all field are correct  Alt 2. The payment info is void and the user is redirected to the billing options web-page and prompted to check their balance/funds as the payment is unable to proceed due to insufficient funds  Alt 3. The users' payment info is unverified |

|  |  |
| --- | --- |
| **Use Case:** | Manage Wishlist (Priority 11) |
| **Actor (s):** | Registered User, Manager/Administrator, Email Service |
| **Goal:** | To create and manage a Wishlist of desired products. |
| **Overview:** | If a user has registered, they will see a ‘Wishlist’ button on the home page. When accessed, the user will see the products that they have added, or an empty page telling them their Wishlist is empty. To add items to the Wishlist, the user would browse the store, once they see something the like, they would Access the product page, and there would be a button to add to Wishlist. If there already are items in the Wishlist, buttons for adding to cart, deleting from the Wishlist and sharing the Wishlist are displayed. A button to clear the entire Wishlist should appear at the bottom of the product list. A website manager/administrator should be able to access a user’s Wishlist. When they do, they should be able to clear the Wishlist, in case a user account has been deleted or suspended. When a user chooses to share their Wishlist, they receive a copy of said Wishlist through Email. The Email is sent by the websites Email service. |
| **Pre-Condition:** | The user must be registered on the website. There must be pre-existing products on the website. |
| **Post Condition:** | The user is able to modify all aspects of their Wishlist. |
| **Successful Scenario:** | User:   1. Web user registers an Account on the website and is redirected to the home page 2. User accesses their Wishlist using a button located in the navigation bar 3. User Deletes an item from their Wishlist using a button located next to the unwanted product 4. User adds an item from their Wishlist into their cart, upon doing so the item from the Wishlist gets removed 5. The user chooses to share their Wishlist, when they click the button, a message telling them that an email with the Wishlist has been sent to them 6. The user chooses to clear their Wishlist. When they click the button, they get a confirmation screen, upon confirming, the Wishlist gets cleared   Manager/Administrator:   1. In case of error in the system, the administrator can delete Wishlist. User accesses the admin page using a button located in the navigation bar 2. User chooses to view all users Wishlist’s. They get a list of Wishlist’s and their associated users 3. If user sees an undesired wish list, they will click a ‘delete’ button located next to the Wishlist 4. When the user clicks the button, they will be asked for confirmation 5. After confirmation, the system prints a message to the screen, informing the user that the Wishlist was removed successfully 6. The website email service sends out an email to the account associated with the Wishlist informing them that an Administrator decided to remove their Wishlist |
| **Alternative Scenario (s)** | Alt User 2. User is not yet registered and Wishlist button does not appear.  Alt User 4. The desired item could be out of stock, the system will print an appropriate message to the user.    Alt Admin 1. The user is not of ‘admin’ type, the button will be missing |

|  |  |
| --- | --- |
| **Use Case:** | Manage Product (Priority 6) |
| **Actor (s):** | Manager/Administrator |
| **Goal:** | To modify data on a pre-existing product in the stores’ database. |
| **Overview:** | The website administrator should have an ‘admin’ button on the home page, once accessed, the administrator can choose to edit/modify an existing item in the store. He will have the options to change the products name, categories, description, price, stock and any media associated with that product. On success, a message will appear telling the user that his modifications we are successful, a link will be provided to the user so that they can inspect their changes. |
| **Pre-Condition:** | The current user accessing the system must be of type ‘admin’ in order to see and access the admin page. There must be a pre-existing item in the stores’ database. |
| **Post Condition:** | The selected product is modified successfully, and the webpage is adjusted appropriately. |
| **Successful Scenario:** | 1. User logs into their account and is redirected to the home page 2. Use accesses the admin page using a button located in the navigation bar 3. User selects they would like to modify a pre-existing product in the store 4. System redirects the user to the appropriate product modification page 5. User changes Product name 6. User changes Product category 7. User changes Product description 8. User changes Product price 9. User changes current stock on hand of the Product 10. User changes media associated with the Product 11. User clicks confirmation button 12. User is redirected to the newly edited product page |
| **Alternative Scenario (s)** | Alt 2. User is not of type ‘admin’ and the button will not appear        Alt 5. Name assigned by the user is more than 256 characters, when the user goes to confirm the product, they will be prompted with a message informing them that the product name must be less than 256 characters long.        Alt 8. User enters a negative value, when the user goes to confirm, they will be prompted with a message informing them that the price must be a positive value.        Alt 9. User enters a negative value, when the user goes to confirm, they will be prompted with a message informing them that the price must be a positive value.        Alt 10. Selected media is in an incompatible format to the website        Alt 11. One or more values entered are incorrect, pressing the confirmation button will display a message to the user informing them of their errors. |

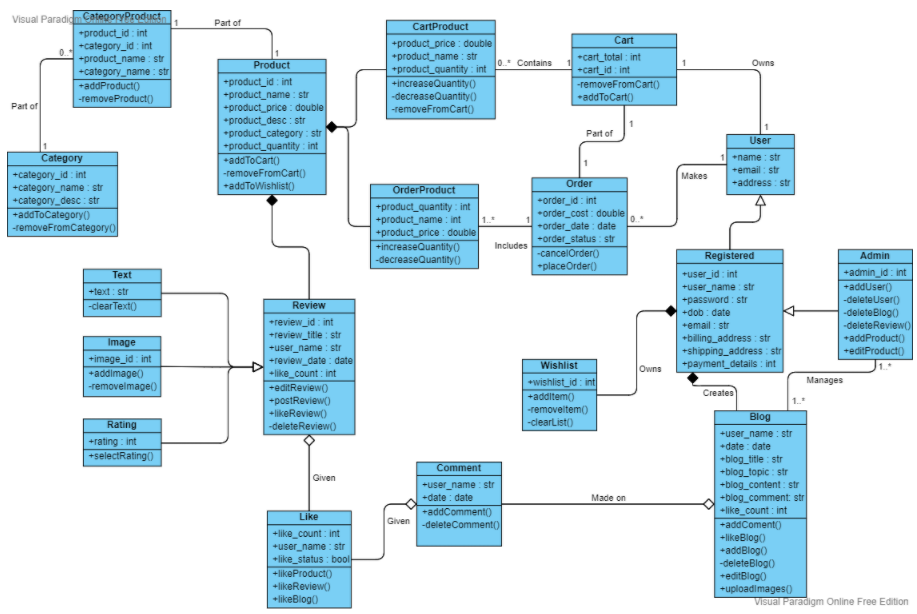
|  |  |
| --- | --- |
| **Use Case:** | Create Product (Priority 7) |
| **Actor (s):** | Manager/Administrator |
| **Goal:** | To add a new product to the store that is not yet stored in the database. |
| **Overview:** | The website administrator should have an ‘admin’ button on the home page, once accessed, the administrator can choose to add a new product to the store. He will be asked for the items name, category, description, price, stock and any media associated with that item. On success, the item will be added to the stores database and a new webpage for that item will be generated. The Administrator will be redirected to the newly created product page. |
| **Pre-Condition:** | The current user accessing the system must be of type ‘admin’ in order to see and access the admin page. |
| **Post Condition:** | A new webpage for the new item will be generated, and the store can commence selling that product. |
| **Successful Scenario:** | 1. User logs into their account and is redirected to home page 2. User accesses the admin page using a button located in the navigation bar. 3. User selects they would like to add a new product to the store 4. System redirects the user to the appropriate product creation page 5. User inputs the product name 6. User selects a category for the product 7. User input the product description 8. User inputs the product price 9. User inputs the current stock on hand of that product 10. User uploads any media associated with that product 11. User click confirmation button 12. User is redirected to the newly created product page |
| **Alternative Scenario (s)** | Alt 2. User is not of type ‘admin’ and the button will not appear        Alt 5. Name assigned by the user is more than 256 characters, when the user goes to confirm the product, they will be prompted with a message informing them that the product name must be less than 256 characters long.        Alt 8. User enters a negative value, when the user goes to confirm, they will be prompted with a message informing them that the price must be a positive value.        Alt 9. User enters a negative value, when the user goes to confirm, they will be prompted with a message informing them that the price must be a positive value.        Alt 10. Selected media is in an incompatible format to the website.        Alt 11. One or more of the values entered are incorrect, pressing the confirmation button will display a message to the user informing them of the errors. |

Use Case Diagram:

Diagram

Description automatically generated with medium confidence

Class Diagram:



Plant Oasis – by Daniel Cullen, Hubert Bukowski & Jibril Abdulazeez

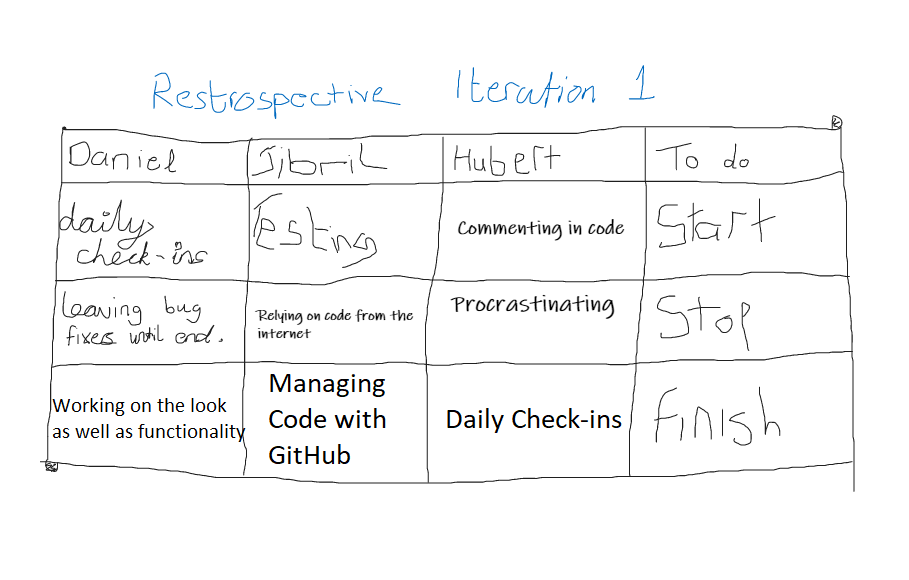
Table of Contents

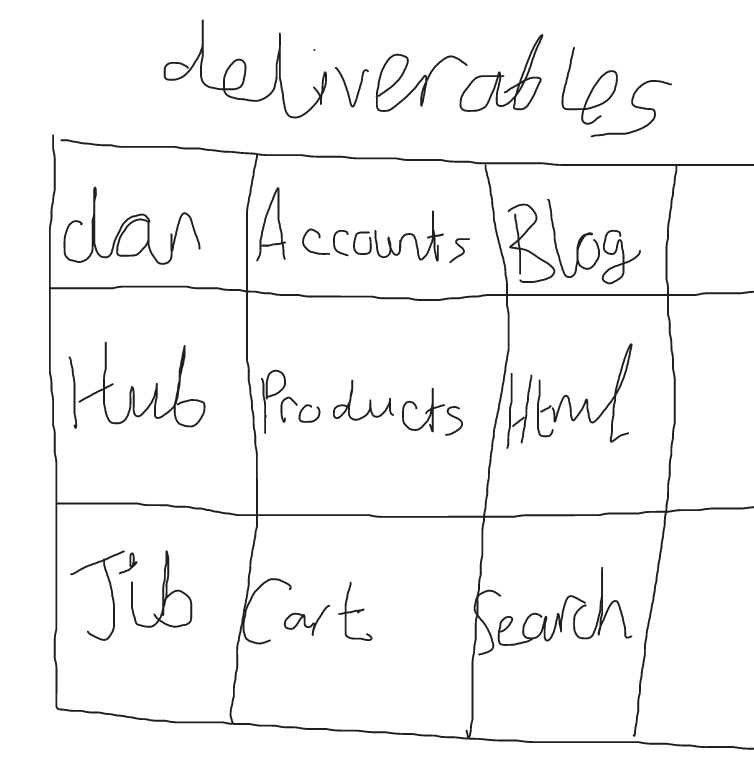
1. Planning
2. Planning
3. Planning
4. Design
5. Design & Tests
6. Tests
7. Tests
8. Tests
9. Tests, Issues & Solutions

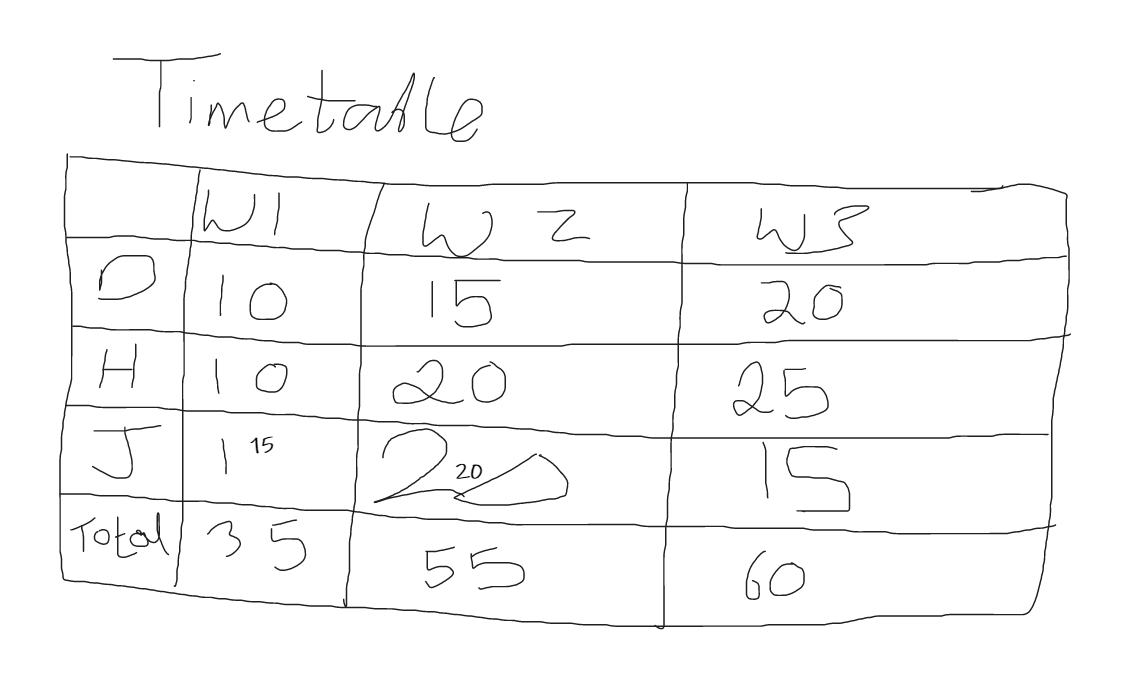
Iteration 1 – Prototype Report – Plant Oasis

Planning:

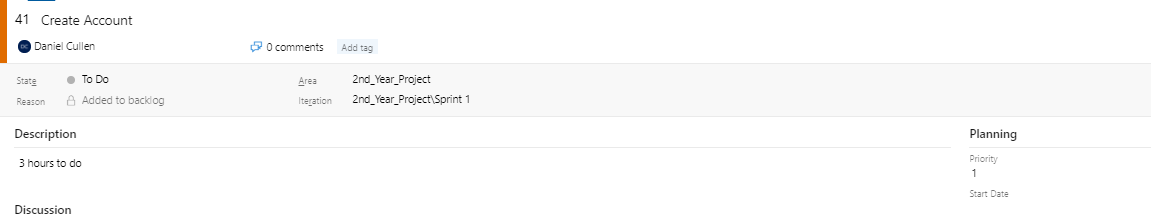
We all joined the teams meeting workshop that was on Tuesday 16th of February, we spoke about the retrospective for iteration 1 which can be seen below. We wrote what we thought we needed to start doing more, what we needed to stop doing and what we believed we should continue with. We had a daily check-in which we spent talking about what we intended to do for the day. During the meeting we also decided on how we would split up the work that needed to be done towards the first iteration. Daniel was assigned with doing the accounts and the blog, Jibril was assigned with doing the cart and search function and Hubert was assigned with doing the products and the html of the website. We also spoke about the maximum number of hours that we believed we could put towards the iteration weekly; we were unable to achieve these hours but we are satisfied with the amount of work and time that we put into the website.

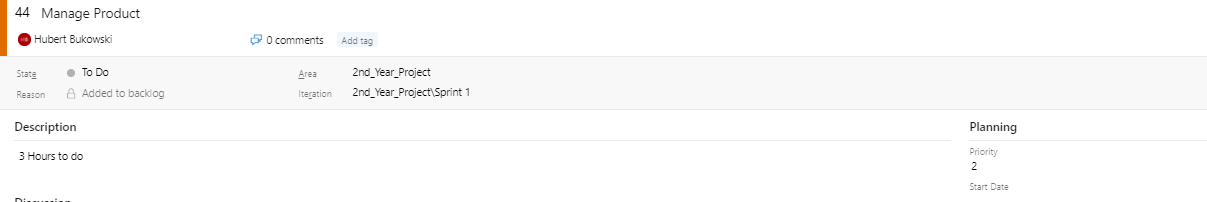






We used azure devops to manage our progress on the website as we pushed and pulled from the repo. We would notify the members of the team when we pushed a new feature to the repo so that the other members could pull it and update their features to work with it. At the planning phase of iteration 1 we prioritized the features that we would be implementing into the website so that we had a path to follow on what we believed were the essential components of the website. This allowed us to focus on the important features before attempting to implement extra features that we thought would add to iteration 1. We used the boards feature of azure devops to categorize these features by priority, an example of this can be seen below displaying the 2 priorities we set on the features.





Design:

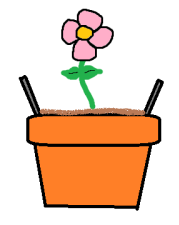
Before Hubert began on the html of the website, we worked on a design for the base html which we all agreed on. We also worked on some logos for the website before we decided on the product which we were happy with. These prototype designs can be seen below which are also compared to what we have finished with.

Diagram

Description automatically generated







We thought it would look more professional if we used a high-quality logo instead of drawing our own, so we used an online logo generator that helped us create a logo that we believed would suit the websites aesthetic and products.

Tests:

We decided to split our tests up between the team based on the features that the team member worked on so Daniel did tests on account and blog, Hubert did tests on the products and html and Jibril tested the cart and search functions. An image of the test cases can be seen below showing the results of each of our tests. The tests were conducted through the use of selenium ide and selenium web driver which we believed was an efficient and robust way of testing the features through different programs.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case Matrix** | | | | | |
| **Test Case No** | **Test Case Name** | **Input** | **Expected Outcome** | **Actual Outcome** | **Result (Pass/Fail)** |
| 1 | signup | Customer01, 20, [customer@gmail.com](mailto:customer@gmail.com), jinglebells, jinglebells | Taken to the sign in page after account is successfully created | Taken to the sign in page after account is successfully created | Pass |
| 2 | signin | Customer01, jinglebells | Taken to the homepage logged into the account | Taken to the homepage logged into the account | Pass |
| 3 | signout | Click profile button then click signout | Taken to the homepage logged out of the account | Taken to the homepage logged out of the account | Pass |
| 4 | profileView | Click profile name then click profile button | Taken to profile information page | Taken to profile information page | Pass |
| 5 | updateProfile | Click profile name then click profile button, increase age, name or email then click update | Taken back to profile information page displaying updated information | Taken back to profile information page displaying updated information | Pass |
| 6 | deleteProfile | Click profile name then click profile button then click delete button and click yes you want to delete account | Taken back to signup page since account is deleted | Taken back to signup page since account is deleted | Pass |
| 7 | addBlog | Click add new blog, new blog as title, choose blog slug, choose username, input blog content, choose to publish the blog, click add. | Taken back to blog list page, blog will be added to the viewable blogs | Taken back to blog list page, blog will be added to the viewable blogs | Pass |
| 8 | updateBlog | Click edit blog, change the title, change the content and change the slug, click update | Taken back to blog list page, blog will be updated in the viewable blogs | Taken back to blog list page, blog will be updated in the viewable blogs | Pass |
| 9 | deleteBlog | Click delete blog, click confirm | Taken back to the blog list page and the blog will not be present | Taken back to the blog list page and the blog will not be present | Pass |
| 10 | addComment | Click to read a blog, enter name, email and comment content, click comment | A message will appear saying that the comment is awaiting moderation, once moderated it will appear on the blog | A message will appear saying that the comment is awaiting moderation, once moderated it will appear on the blog | Pass |
| 11 | navbarNavigation | Click blog, click cart, click admin, click add item, click username, click edit profile, click home, click logo | All links in the navigation bar work as they should and redirect as appropriate | All links in the navigation bar work as they should and redirect as appropriate | Pass |
| 12 | categoryNavigation | Click Indoor Plants, Click outdoor plant, Click plant pots, click tools | All category links work as they should and redirect to their respective product lists | All category links work as they should and redirect to their respective product lists | Pass |
| 13 | productNavigation | Click product 1, click home, click product 2, click outdoor plants, click product 3, click plant pots, click product 4 | All the product links redirect to their respective product detail views | All the product links redirect to their respective product detail views | Pass |
| 14 | addProduct | Click admin, click add product, enter name, enter alt name, enter description, enter price, enter stock, click submit, select category, click submit | A new product with the specified fields is created and the user is redirected to the home page | A new product with the specified fields is created and the user is redirected to the home page | Pass |
| 15 | editProduct | Click product, click edit product, set stock to 0, click back to product button | The product stock has been reduced to 0 and the product page is displayed appropriately | The product stock has been reduced to 0 and the product page is displayed appropriately | Pass |
| 16 | RemoveProduct | Click Product, click delete product, click cancel, click product, click delete product, click confirm | The product is removed from the database and is no longer displayed on the home page | The product is removed from the database and is no longer displayed on the home page | Pass |
| 17 | searchProduct | Click searchbar, input “Bonsai Arts”, press enter to search | The product “Bonsai Arts” is displayed on page with the title “Bonsai Arts: Search Results” | Search results page is returned | Pass |
| 18 | searchProducts | Click searchbar, input “a”, press enter to search | The page should return all products containing “a” in their product name | Search results page returns every product with an “a” listed in the product name. | Pass |
| 19 | cartProduct | Click indoor plants, click product of choice, click add to cart | Product should add to cart, redirect the user to the cart page and display a shopping cart icon with the label (1) | User is redirected to shopping cart page with the product added, listed and the shopping cart icon appears to reflect the number of items in cart | Pass |
| 20 | increaseQuantity | Click shopping cart, click + icon below product item name | Cart should update price and quantity to reflect the increased number of products added to cart | Cart pricing and quantity updates to reflect the new number of products in cart | Pass |
| 21 | decreaseQuantity | Click shopping cart, click – icon below product item name | Cart should update price and quantity to reflect the decreased number of products added to cart | Cart pricing and quantity updates to reflect the new number of products in cart | Pass |
| 22 | emptyCart | Click shopping cart, click x icon next to the subtotal | Cart should empty and a new title labeled “Your shopping cart is empty” should appear, alongside the removal of the numbered label beside the shopping cart icon in the navbar | User page now displays “Your shopping cart is empty”, cart is emptied and a number label is no longer displayed alongside the shopping cart icon in the navbar | Pass |
| 23 | checkStock | Click product that is out of stock | User should be unable to add product to cart as it is out of stock | User is unable to cart the out of stock product | Pass |

Issues:

* A major issue that we came across was that the slugs for categories and products would not work which caused the dispatcher to be unable to distinguish between urls in the shop, blog and cart apps.
* Another issue that we came across was that the url for blog and cart would crash the website when accessed from any part of the site other than the homepage. This was caused by the url name being appended to the end of the url instead of the url changing.

Solutions:

* The first issue was fixed by Hubert as he added the app name before the slug in each of the urls to help distinguish them. An example of the difference of the urls can be seen here path('<slug: slug>/', views.allProdCat, name='products\_by\_category') compared to path('shop/<slug:slug>/', views.allProdCat, name='products\_by\_category').
* We fixed the second issue by changing the url from \cart to {% ‘cart\_detail’ %} and the same was done for the blog app.

An extra feature we plan on adding in iteration 2 is creating a new customuser called manager that will be used instead of admin for the front end editing and deleting.

Plant Oasis Iteration 2 – by Daniel Cullen, Hubert Bukowski & Jibril Abdulazeez



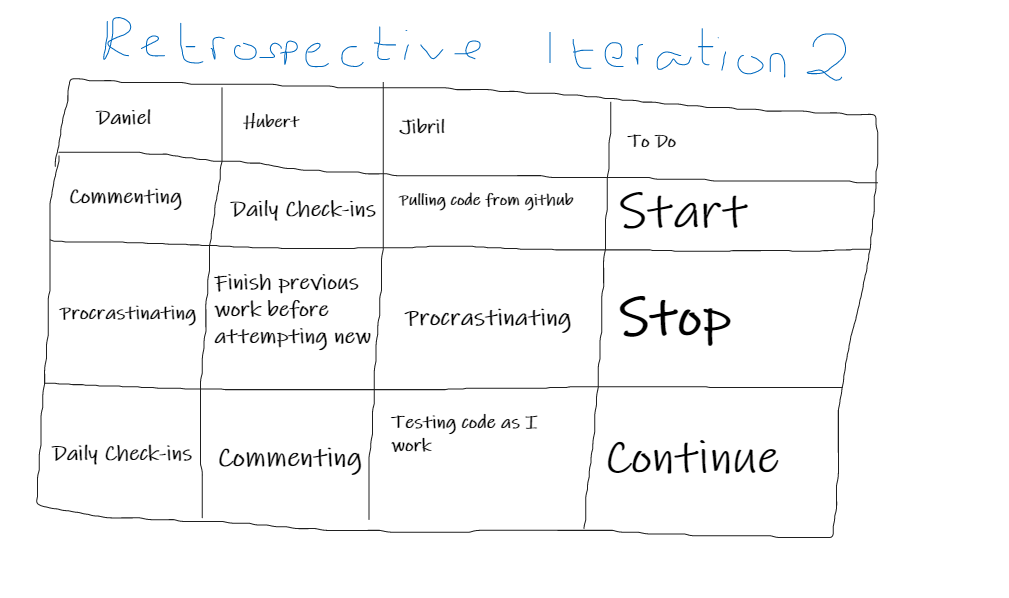
Table of Contents

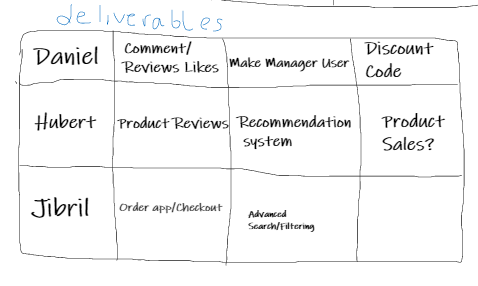
1. Table of Contents
2. Planning
3. Planning
4. Progress
5. Progress
6. Progress
7. Iteration 3 ideas
8. Tests
9. Tests
10. Tests
11. Tests
12. Tests
13. Tests
14. Tests
15. Tests
16. Tests
17. Tests

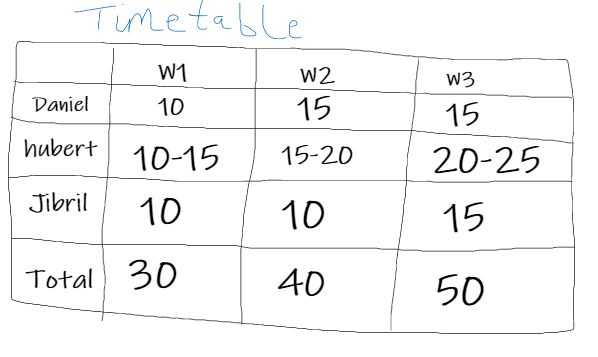
Iteration 2 – Prototype Report – Plant Oasis

Planning:

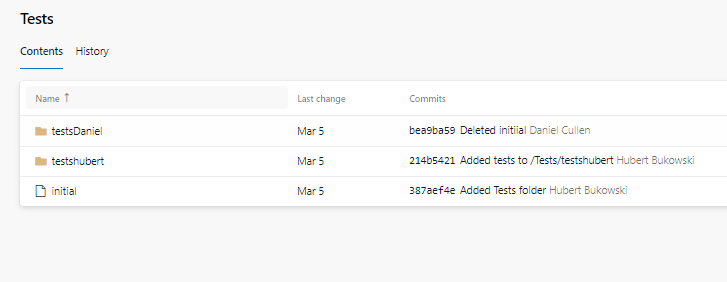
On the 9th of March we joined the workshop teams meeting for preparation of iteration 2 of the project. During the workshop we discussed the retrospective for iteration 2 which is attached below. Based on our experience of the previous iteration we wrote what we thought we should continue to do, what we could start doing and what we should stop doing. In iteration 1 we discussed doing daily check-ins which we continued to do throughout iteration 2 which we believe helped us a lot with communication. In iteration 2 we improved on the retrospective of iteration 1, including things that worked for us and adding new things that we believed would help us improve further. We also discussed the deliverables of each member, sharing the workload out equally and to the satisfactory of each member. Daniel was assigned Comments, Blog likes, Manager User and Discount Codes. Hubert was assigned Product Reviews, Recommendation System, Product and Product Sales. Jibril was assigned Order App, Checkout and Advanced Search/Filtering. The last section we spoke about was the time management of the group, we made a timetable that can be seen below containing the hours we believed we would be able to put towards the iteration. We did not hit the times that we set due to heavy workload from other subjects, but we were satisfied with the amount of time we put towards the iteration.





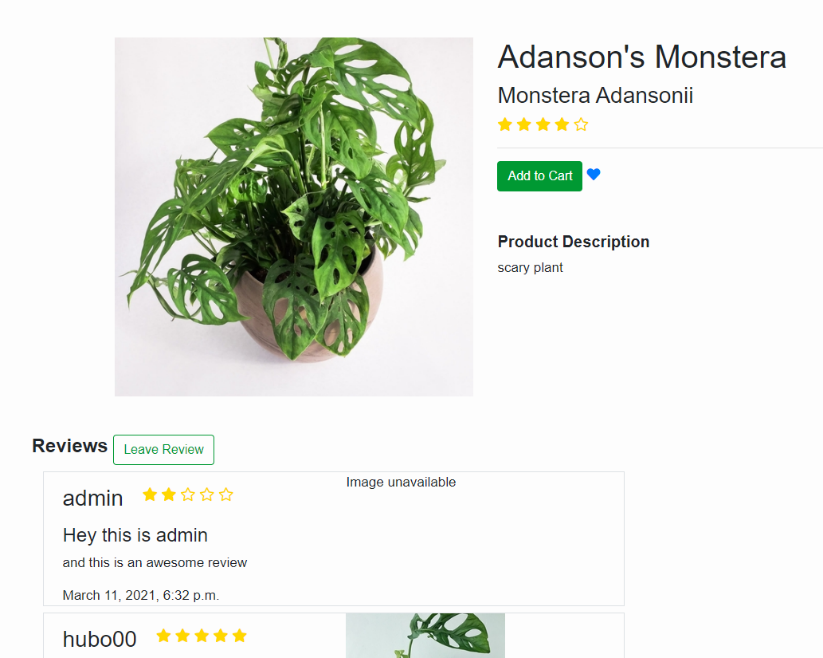


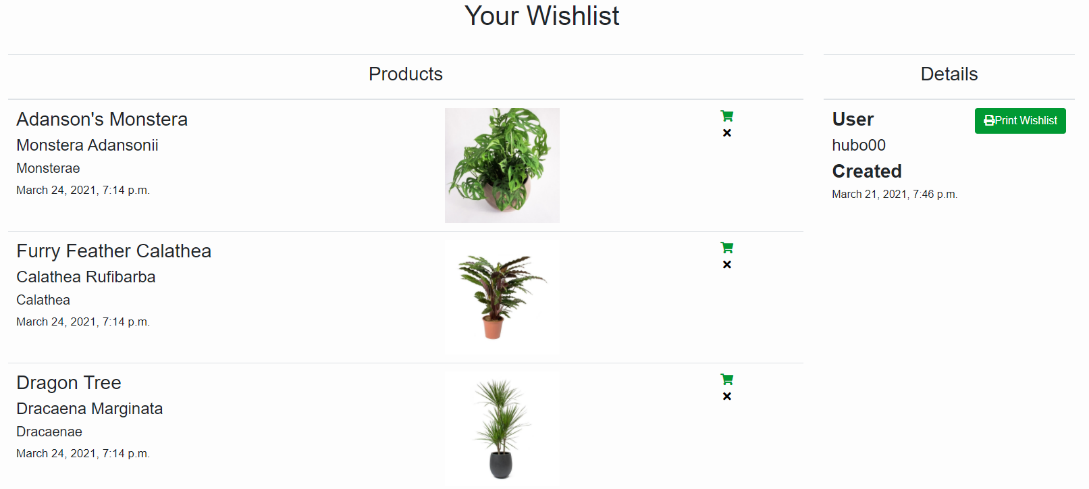
We used azure devops to create a folder four our tests and reports for each iteration. We used the test folder to keep our tests for each use case sorted and it also allowed us to see that each member was doing their tests for each part of the iteration that they worked on.

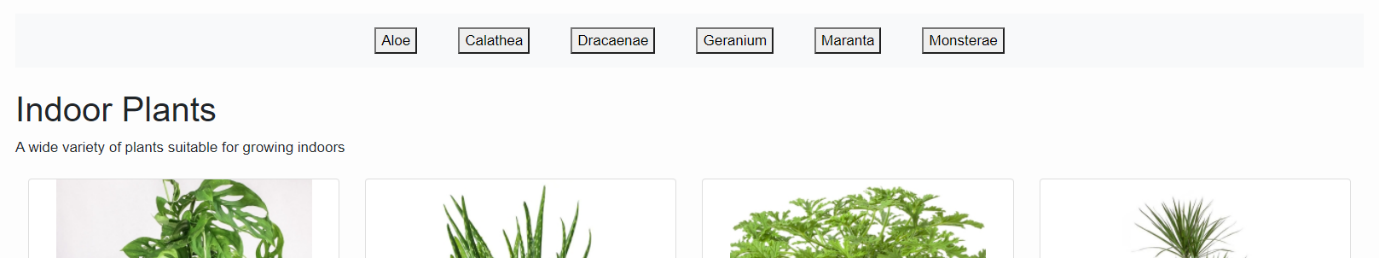


Progress:

During iteration, a lot of progress was made on the ecommerce portion of the website. Hubert focused on the creation of reviews, wish list and recommendation system. A focus that Hubert had for the reviews was a star rating that allowed the user to rate the product but also display the rating to other users in the product list page. Hubert also created a wish list that allowed the user to add products they liked to a list of their own without it being added to their cart. A function was also added that allowed the user to add these products to their cart directly from the wish list making it a much easier experience for the user to purchase the products that they loved. Hubert attempted creating the recommendation system but ran into some issues that he believes he will be able to rectify in iteration 3.

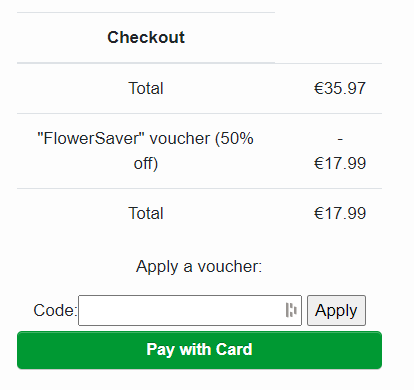


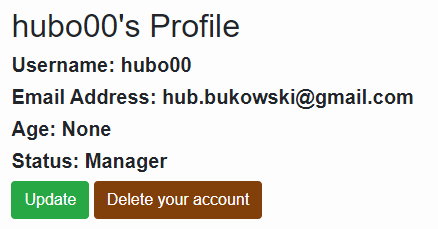




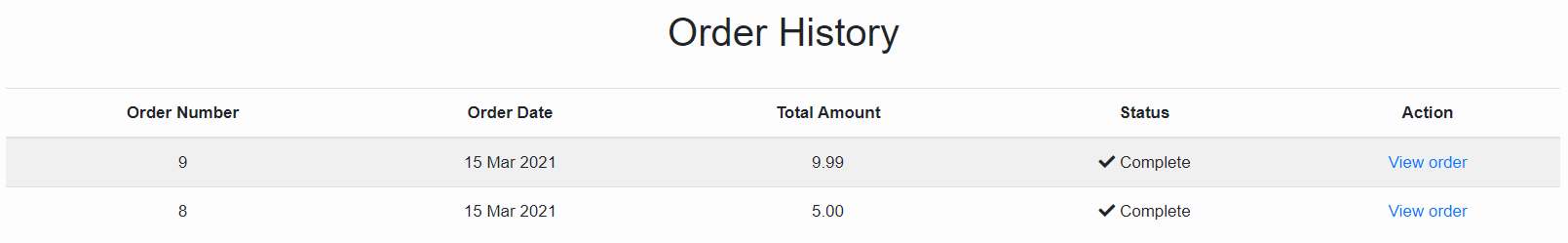
During the iteration Daniel worked on the likes for blogs, the discount codes for products and the manager users. The like/dislike button that appears on the blogs allows the user to add a like to the blog or dislike it which increases or decreases the amount of likes on specified blog. The discount codes allow the user to enter a discount code which will decrease the price of their cart depending on the percentage set for the discount code. Daniel also worked on the manager user of the website, allowing the manager to access, edit and delete certain features on the website such as products and blogs.

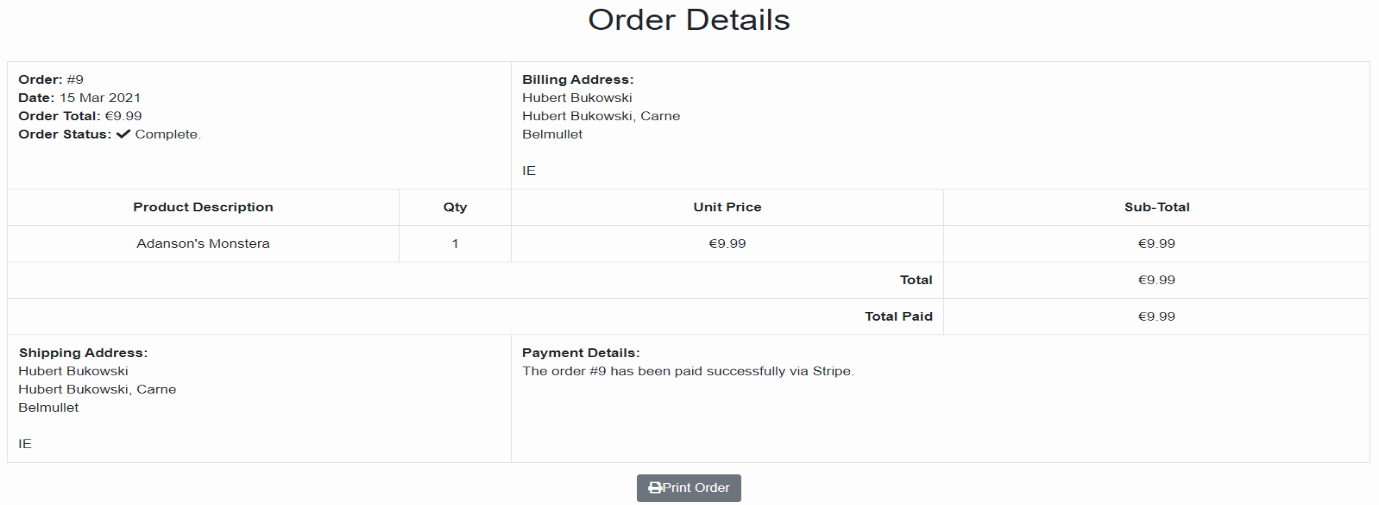


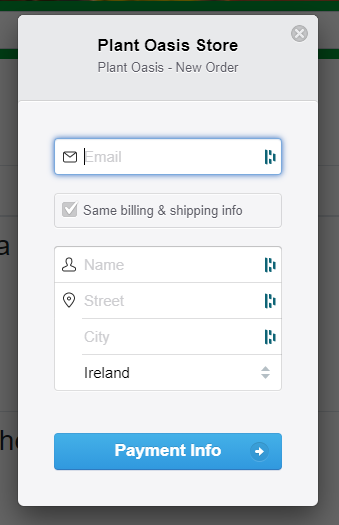




Jibril worked on the Order App and Checkout of the website during iteration 2. The order app allows the user to see their previous orders made on the website, this allows the user to see all the details of their order and even print it out. Jibril also created the checkout for the website allowing the user to view, add, delete or purchase the products that are in their cart, purchasing the products prompts the user to input their card details through stripe which would create the order.







Iteration 3 ideas:

* Separate like and dislike button.
* Advanced Search/Filtering.
* Recommendation System.
* Cancel orders.
* Adding further manager capabilities.
* Adding further wish list capabilities.
* Adding JavaScript features into the website.
* Email subscription.
* Adding detailed order statuses.
* Add images to blogs.
* Add likes to reviews.
* Add buy again feature.
* Add videos.
* Add contact page/form.

Tests:

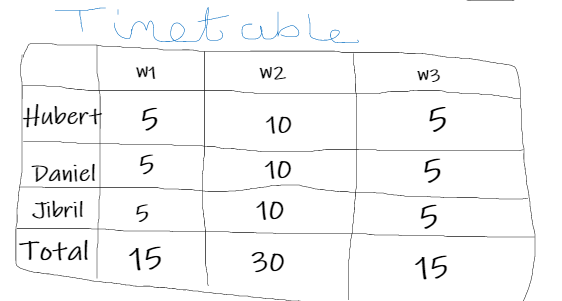
We used selenium ide to test all our new features that were implemented during iteration 2. These can be seen down below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case Matrix | | | | | |
| Test Case No | Test Case Name | Input | Expected Outcome | Actual Outcome | Result (Pass/Fail) |
| 1 | subCategoryNavigation | Click ‘indoor plants’, click ‘Calathea’, click ‘outdoor plants’, click ‘Geranium’, click ‘plant pots’, click ‘terracotta’, click ‘tools’, click ‘shears’ | The user can successfully navigate between categories and their subsequent subcategories with no issues | The user can successfully navigate between categories and their subsequent subcategories with no issues | Pass |
| 2 | browseReviews | Click product 1, scroll to bottom of page, click home, click product 2, scroll to bottom of page | The user can browse all reviews on a specific product and the reviews display under the correct products | The user can browse all reviews on a specific product and the reviews display under the correct products | Pass |
| 3 | addNewReview | Click product, scroll to bottom of page, click ‘leave review’ button, click ‘rating’ dropdown, select 4, click ‘title’ input box, enter title, click ‘content’ input box, enter content, click ‘submit’ button | The user is able to leave a review on their selected product consisting of a rating, a title and content for the review, once the user hits submit they are redirected back to the product page where their review is now displayed. | The user is able to leave a review on their selected product consisting of a rating, a title and content for the review, once the user hits submit they are redirected back to the product page where their review is now displayed. | Pass |
| 4 | editReview | Click product, scroll to bottom of page, click ‘edit’ under the users review, click ‘rating’ dropdown, select 1, click ‘title’ input box, enter new title, click ‘content’ input box, enter new content, click ‘submit’ button | The user is able to edit their review and alter the rating, title and content. Once they are finished with their review, they can hit the submit button, upon doing so they are redirected back to the product page where their updated review resides. | The user is able to edit their review and alter the rating, title and content. Once they are finished with their review, they can hit the submit button, upon doing so they are redirected back to the product page where their updated review resides. | Pass |
| 5 | deleteReview | Click product, scroll to bottom of page, click ‘delete’ under the users review, click ‘cancel’ button, click product, click ‘delete’, click ‘confirm’ button | The user is able to choose if they would like to delete the selected review, they have the option of selecting cancel, if they do they are redirected to the home-page. If they choose to delete the review, they are redirected to the home page and the review is no longer displayed on the product page | The user is able to choose if they would like to delete the selected review, they have the option of selecting cancel, if they do they are redirected to the home-page. If they choose to delete the review, they are redirected to the home page and the review is no longer displayed on the product page | Pass |
| 6 | navigateWishlist | Click ‘wishlist’ button, click home, click product 1, click heart icon ( wishlist ), click home, click product 2, click heart icon ( wishlist ), click home, click product 3, click heart icon ( wishlist ) | The user is able to navigate the website and add products that they like into their wishlist using an icon situated next to the buy button, when a user adds an item to their wishlist, they are redirected to their wishlist and their newly added product is displayed | The user is able to navigate the website and add products that they like into their wishlist using an icon situated next to the buy button, when a user adds an item to their wishlist, they are redirected to their wishlist and their newly added product is displayed | Pass |
| 7 | removeFromWishlist | Click ‘wishlist’ button, click product 1 ‘delete’ icon, click product 2 ‘delete’ icon, click product 3 ‘delete’ icon | The user is able to remove individual items from their wishlist | The user is able to remove individual items from their wishlist | Pass |
| 8 | WishlistAddToCart | Click ‘wishlist’ button, click add to cart icon, click ‘wishlist’ button, click add to cart icon, click ‘wishlist’ button, click add to cart icon | The user is able to add items to their cart directly from their wishlist | The user is able to add items to their cart directly from their wishlist | Pass |
| 9 | discountCode | Click ‘cart’ button, click ‘discount code’ box, input ‘Flower Saver’ and click ‘submit’ | A savings of 50% of overall cart value is deducted from the total price. | A savings of 50% of overall cart value is deducted from the total price. | Pass |
| 10 | likeBlog | Click ‘Blogs’ link, click ‘read more’ on a blog, click ‘like/dislike’ button | If the user has not already liked the blog the like counter will increase in 1 | If the user has not already liked the blog the like counter will increase in 1 | Pass |
| 11 | dislikeBlog | Click ‘Blogs’ link, click ‘read more’ on a blog, click ‘like/dislike’ button | If the user has already liked the blog the like counter will decrease in 1 | If the user has already liked the blog the like counter will decrease in 1 | Pass |
| 12 | managerProfile | Click ‘username’ button, click ‘profile’ button, look at status | If the user is a manager their profile will display them as a manager, admin and manager have been separated. | If the user is a manager their profile will display them as a manager, admin and manager have been separated. | Pass |
| 13 | managerDeleteBlog | Click ‘Blogs’ link, click ‘delete’ button on any blog, click ‘yes’ when confirming deletion of said blog | The blog that was chosen to be deleted will be deleted. | The blog that was chosen to be deleted will be deleted. | Pass |
| 14 | managerDeleteProduct | Click on any of the products displayed, click ‘delete’ button on said product page, click ‘yes’ to confirm deletion of the product | The product that was chosen for deletion will be removed from product list page and categories. | The product that was chosen for deletion will be removed from product list page and categories. | Pass |
| 15 | managerAddProduct | Click ‘manage’ button, click ‘add product’ button, input correct details for title, description, price, stock, image, alternative name, and category for product, click ‘submit’ button | The product the manager created will be added to product list page and selected category. | The product the manager created will be added to product list page and selected category. | Pass |
| 16 | managerEditProduct | Click on any of the products displayed, click ‘edit product’, input new details for product ensuring the input values are accepted, click ‘update’ button | The details of the product will be updated to the new values entered by the manager. | The details of the product will be updated to the new values entered by the manager. | Pass |
| 17 | purchaseProduct | Click ‘Indoor Plants’, click on any of the products displayed, click ‘Add to Cart’ button, click ‘Pay with Card’ button, Input payment deails | User is redirected to a order thank you page | User is redirected to the order thank you page with the “Thanks for placing your order” message displayed and is returned an order number | Pass |
| 18 | viewOrders | Click username in navbar, click ‘Orders’, click ‘View order’ listed under Action | User is directed to a Order Details page listing all their details | User is directed to the Order Details page listing their payment details, shipping address, billing address and invoice | Pass |
| 19 | printOrder | Once in Order Details click ‘Print Order’ | User should be able to print their order details on a singular page or save as a PDF | User is able to print their order details on a singular page and save as a PDF | Pass |
| 20 | createVoucher | Django Admin – Click ‘vouchers’, click ‘+Add’, input voucher details in fields, click ‘active’ | A message displayed “Voucher object” was added successfully should be displayed | “Voucher object” was added successfully message is displayed | Pass |
| 21 | addToWishlist | Click ‘Indoor Plants’, click on any of the products displayed, click the heart icon displayed next to the product | User is redirected to their wishlist page with the product wishlisted displayed | User is redirected to their wishlist page with the product wishlisted displayed | Pass |
| 22 | printWishlist | Once in Wishlist, click ‘print wishlist’ | User should be able to print their wishlist on a singular page or save as a PDF | User is able to print their wishlist details on a singular page or save as a PDF | Pass |

Plant Oasis Iteration 3 – by Daniel Cullen, Hubert Bukowski & Jibril Abdulazeez



At the beginning of iteration 3 we had our final workshop to prepare us for the work that we had to do during the last phase of the prototype. We joined a call together and discussed the main features that we wanted to complete and discuss what parts of the website we would go through to ensure there were no issues or bugs. We also created a timetable of how much time we thought we would be able to put into the iteration during the final weeks.



The focus that we had for iteration 3 of the project was ensuring that any of the features that we had already implemented into the website worked and that there no apparent bugs or issues with the code that we had until that point. We were not focused on adding big new features, we wanted to add to or improve on our existing features if we felt that there was anything lacking.

Some changes did happen to the website before the testing began, a wish list was added to allow the customer to add products to a list that they could later go back to without having to add the products to their cart. A feature that was improved upon was the comment system, previously an admin would have had to manually access the admin part of the website to activate comments, but a new page was made that displayed all the inactive comments which took the admin straight to the activate comments page. We believe that this made the experience for admin a lot smoother when activating comments from the website. The manager user was also implemented which had less capabilities than the admin but still had a lot of extra control over the different features compared to a regular user, the manager was given the ability to add/delete products, delete blogs and edit previously made products, these abilities are shared by the admin user too.

Due to time constraints and the priority of testing and bug fixing there were many features or improvements that we were unable to implement into the finished prototype of the website. Some examples of these missing features or improvements would have been the sort by feature for the products which would have allowed the user to sort the products by price, availability and date added. An improvement that was not added to the website was the ability to allow users to change their blog from a draft to a published blog, this can only be done through the admin side of the website which the regular users and manager cannot access, in the future this would be a nice feature to add which would improve the user experience. Another big feature that we were unable to successfully implement in the time frame that we had was the recommendation system, this would have collected data on the products that a user purchased and would have recommended them similar items that they may have been interested in purchasing. We decided that our time would have been better spent on testing and bug fixing instead of on a feature that we were unsure would have been complete in time.

We tested the new additions using selenium ide and made test cases for them like the previous tests that were conducted before hand. We also reran the previous ide files that were made when testing iteration 1 and iteration 2 to ensure that the old features were still working in iteration 3. We also went through the entire website to ensure that all of the visuals were working as they should, and that the website was easy to navigate.

Test cases:

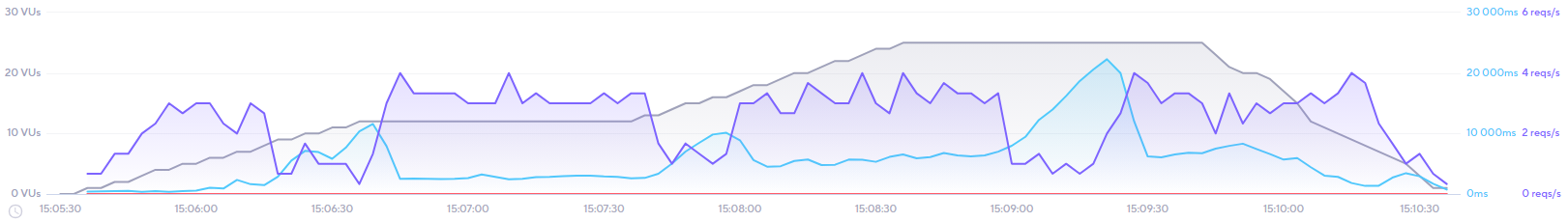
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case Matrix** | | | | | |
| **Test Case No** | **Test Case Name** | **Input** | **Expected Outcome** | **Actual Outcome** | **Result (Pass/Fail)** |
| 1 | CommentMonitoring | Click the ‘Blogs’ link, click ‘read more’ on a blog, insert name, email and body of your comment, click ‘submit’ | A box will appears under the comment section telling the user that their comment is being monitored before it will appear on the blog | A box will appears under the comment section telling the user that their comment is being monitored before it will appear on the blog | Pass |
| 2 | ActivateComment | User must be an admin, click the ‘manage’ drop down menu, click the ‘activate comment’ button, click the ‘activate’ button on one of the inactive comments, click the comment that should be activated, click the ‘active’ button at the bottom of the specified comment | The comment will be activated and will appear in the comment section of the blog it was written for | The comment will be activated and will appear in the comment section of the blog it was written for | Pass |
| 4 | clearWishlist | User must be logged in, Click Wishlist icon, click ‘Clear Wishlist’ button, Confirm | The Wishlist is cleared and a notification telling the user as such is displayed near the top of the screen. | The Wishlist is cleared and a notification telling the user as such is displayed near the top of the screen. | Pass |

**Performance Tests**

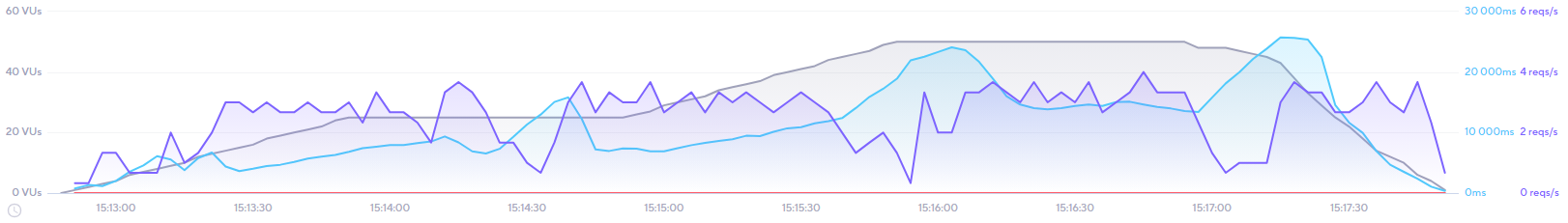
We attempted to use docker to run our Performance tests but unfortunately couldn’t get it to work. We instead opted for hosting our website on PythonAnywhere and running our tests on that instead. This is regrettably not as reliable, as now we must factor in the additional overhead time provided by PythonAnywhere Servers. We decided to use k6.io to run our performance tests. The performance tests included Stress, Load, Spike and Soak tests. We also ran each test in sets of two, one 25 user test, and another 50 user test. This is to get a more accurate reading on the absolute stress the website can handle.

**Stress Tests**

25 Virtual Users



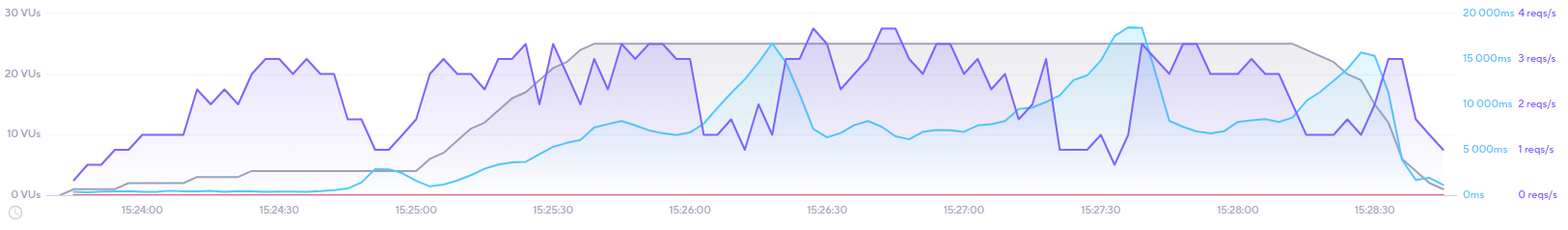
50 Virtual Users



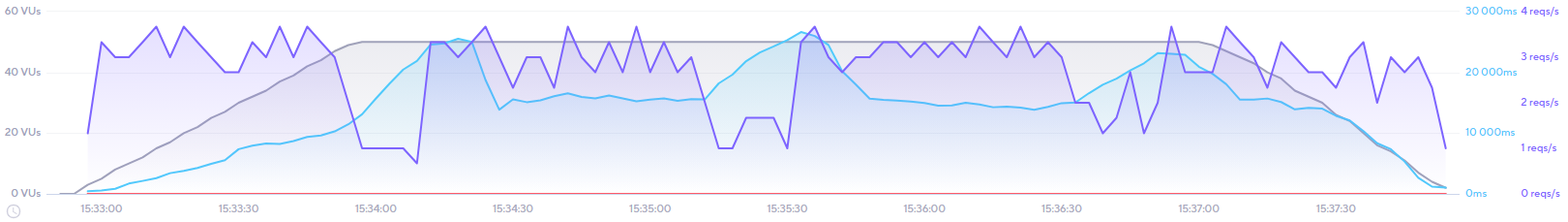
As you can see from the graphs provided by k6.io, When scaling our stress test, the response times increase quite a bit. The website seems to handle 25 users, having an average response time of 5,000ms. We can also see that just as there is about to be a big influx of users, the Response time can increase drastically peaking at over 21,000ms, which we believe is possibly due to PythonAnywhere GET request overhead. The 50 Virtual users test was much harsher on the website, with the average response time now being over 11,000ms, more than double the previous test. Despite this, both tests hadn’t crashed once, as seen by the static red line at the bottom of both graphs.

**Load Tests**

25 Virtual Users



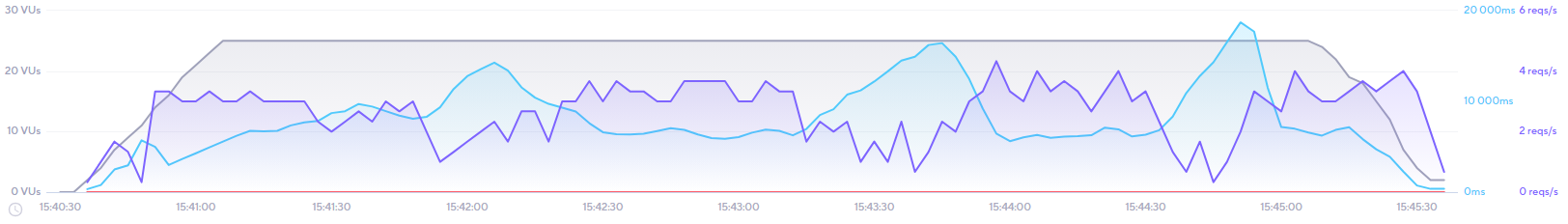
50 Virtual Users



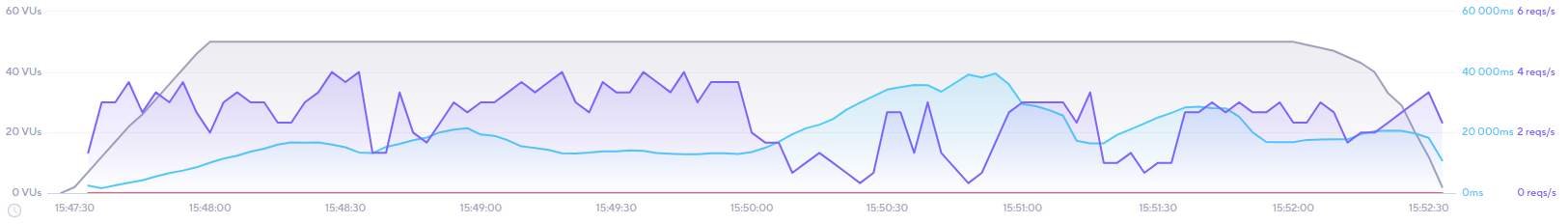
In the load tests we can see that the higher user volume test had some extra response time over the 25-user test. This could tell us that the website possibly has some difficulty handling many users at a time. Despite this the line signifying website crashes has not spiked a single time in both tests, showing that despite the higher response time, the tests had not managed to bring the website down. In addition, we must once again factor in the PythonAnywhere GET response overhead.

**Soak Tests**

25 Virtual Users



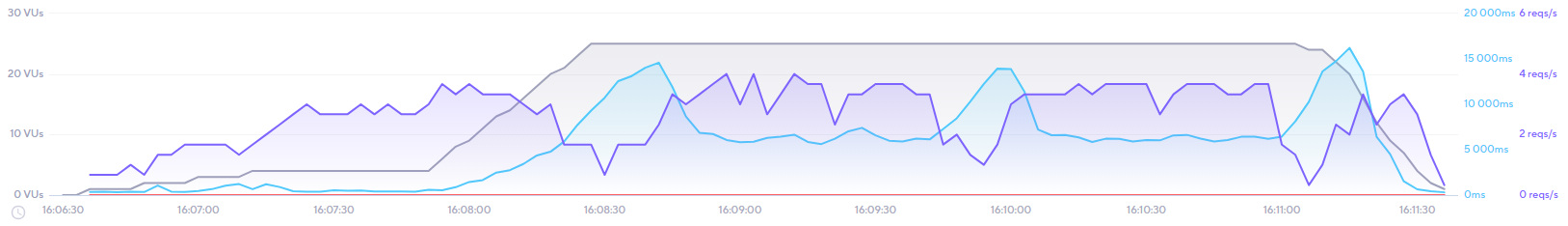
50 Virtual Users



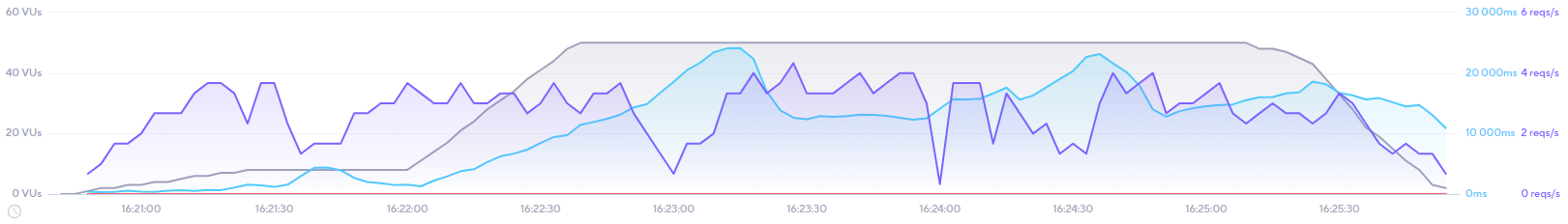
The soak test graphs are interesting. Despite the first graph having less agents involved, it’s responded time spikes we much higher relative to the number of users in both tests, with some spikes having an almost triple response rate. Despite this, the test was completed successfully, having an average response time of 7,500ms. The second test, as stated previously, seems to have performed better to the relative number of users, with the response rate line never peaking above the number virtual users. Similarly, to the previously conducted tests, the response time increases just before users enter the site. Despite this, both tests had not crashed a single time as shown by the static red line.

**Spike Tests**

25 Virtual Users



50 Virtual Users



From both spike tests we can analyze that the tests ran with some very good respond rates at the start, until the virtual users started to fill the website, at which point the response rate rose, until it reached the peak of the spike, and stabilized again at around half the value of the spike. The response rate line did not exceed the user amount until after the test was slowing down, at which point the response rate of both tests stayed relatively high before dropping again. Once again, this test did not cause the website to crash as shown by the static red line located at the bottom of each graph.

**Conclusion:**

We believe that we have made some great achievements throughout the completion of our 2nd Year Project. Even though it was difficult having to work as team without the benefit of seeing each other in person or in a more practical environment to work we were able to create a working website that we are proud of. At the beginning of the project, we only had a homepage with a name and a navigation bar with buttons that did not do anything, but after each iteration the website began to become full of features that we worked hard to implement. Some of the features that we are glad to have successfully implemented were the product reviews, blog comments, likes/dislikes, wish list and the star ratings for the products.

There were many features that we were unable to complete or implement at all into the project either due to time constraint or priority. A few examples of these features are the recommendation system, the ability to filter products and a subscription payment. Due to the time constraint at the end of the project it was difficult to get the recommendation system to work so we decided that it would be best to leave it out of the website and focus testing the website and ensuring there were no bugs in the features that were completed. Due to the priority of testing and bug fixing of the website we also decided to leave the filter and subscription out of the completed project.

We are proud of the amount of work that we were able to achieve in the time frame that was given to us and the great teamwork that we had. We shared our screens a lot during the entire project to help each other with issues and get inspiration from the other members and we also had meetings to ensure that we were all getting our work done and that we were on track for each iteration. Azure devops was used to great extent as we pushed our code to the repo frequently to ensure that each member had up to date repos to work on when implementing their features.

We have ideas on how we could improve the website in the future, we would really like to implement the recommendation system, the filter, and the subscription service in a future iteration. We also have other ideas on what we would add such as a live chat which customers could use to get advice on their planting needs and questions. There were many other features that we thought of but did not have to time to implement such as a page to see what stage your online order is in such as ‘in transit’, an email subscription, a sort by feature for products and many more. We would also do more research on other plant websites to gain more inspiration for other features and design choices.

We are very pleased with the progress that we made in the timeframe that we had to complete this project, there may have been a few shortcomings due to time, but we were capable of creating a functioning site in a short period of time successfully which we are proud of.

**References:**

* Gupta, S., Das, S. and Shahzaib, M., 2021. *Building A Blog Application With Django | Django Central*. [online] Django Central. Available at: <https://djangocentral.com/building-a-blog-application-with-django/> [Accessed 25 April 2021]
* DEV Community. 2021. *How to add Like/Unlike button to your Django Blog*. [online] Available at: <https://dev.to/radualexandrub/how-to-add-like-unlike-button-to-your-django-blog-5gkg> [Accessed 25 April 2021]
* L, H., 2021. *Creating Comments System With Django | Django Central*. [online] Django Central. Available at: <https://djangocentral.com/creating-comments-system-with-django/> [Accessed 25 April 2021].