**Usability testing**

The basics of usability testing is the act of getting testers to see how easy something is to use. A typical usability testing session consists of users who are asked to complete various tasks using the product being tested. The benefits of usability testing include;

* Saves time for both the company producing the product and its users (otherwise other more time-wasting testing methods would need to be used, or more issues would be caught once the product is released)
* Provides a better user experience
* Demonstrates user’s satisfaction with the product

In the case of the ‘MeNU’ app two users were asked to sign up, navigate through the pages of the app and finally make an order. This was done on two separate occasions whereby some suggestions from testers were added to the app.

**January testing**

In January 2020 a preliminary version of the app was created for testing, two users were given the app and a form to fill out asking what they liked/disliked about the app and if they had any improvement suggestions.

**Testing results**

|  |  |
| --- | --- |
| Tester | Sam Milward |
| Comments |  |
| I like the idea of showing the location of the meals on the map. The navigation drawer is also a very nice addition to the app as I find it helps with flow around it. It seems odd to have the navigation drawer on one page only. Perhaps it would be worthwhile adding it to each of the pages in which it navigates to. | |
| Suggested Changes |  |
| Add the navigation drawer to the pages, settings, basket, my meals and current orders. | |

|  |  |
| --- | --- |
| Tester | Liam Franey |
| Comments |  |
| I like the idea of the app, also the fact that every feature within the application is designed with a purpose in mind that contributes to the overall goal of the project. | |
| Suggested Changes |  |
| The meal vendor should have a method of notifying users the maximum number of dishes they can order of a specific food. I also think chefs should be able to set sale times for each individual meal, instead of all meals. | |

**Changes made**

* Added the navigation drawer to the pages, settings, basket, my meals and current orders. In retrospect this makes a lot of sense and adds to the flow of the app.
* Added a max number of dishes on display feature.
* Chefs can now set sale times for each individual meal.

**March testing**

In march 2020 an updated version of the app was created, and subject to testing by two users

**Testing results**

|  |  |
| --- | --- |
| Tester | Sam Milward |
| Comments |  |
| The app has progressed a lot since I last saw it. I’m glad to see my suggestions were added into the app, as a result the flow between different pages is clearer and more streamlined. | |
| Suggested Changes |  |
| Perhaps it would be worthwhile changing the “eat at home” option when ordering meals to something less ambiguous. The ability to add more than one order to the basket would be useful, as users may wish for different meals. | |

|  |  |
| --- | --- |
| Tester | Liam Franey |
| Comments |  |
| Its good to see my suggestions implemented into the app. The flow between the pages is much better now that the navigation drawer has been added to the other pages. | |
| Suggested Changes |  |
| Specifying the total price of an order instead of only the price of a single meal. The ability to search for meals as well as chefs on the search page could be useful. | |

**Changes made**

* Made orders separate, so that users can make multiple orders from different chefs at the same time.
* The words “eat at home” were changed to “eat in” on the meal ordering page to specify the fact that the user will be eating at the chef’s house.
* Orders now display their total price, i.e. accounting for the number of meals, instead of the price of a single meal.
* On the searching page, an option to search for either meals or users/chefs was added. Previously this was only users/chefs.

**Unit testing**

Unit testing is a type of software testing in which individual components/units of a software are tested. This allows for validation of the functionality of the individual components. There are various benefits to unit testing;

- It increases confidence in changing/maintaining code. When well written unit tests are run on every system change, errors can be caught quickly preventing these changes from being added to version control.

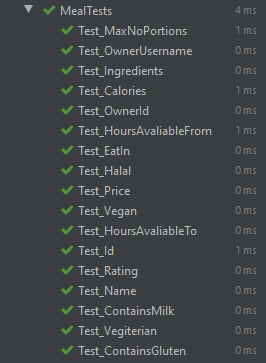
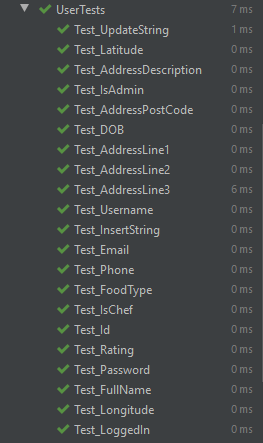
- It makes code more reusable. This is as a result of the forced format which code is forced into in order to make unit testing possible.

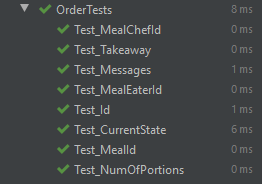
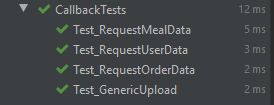
- Generally, the cost of fixing an error is lessened in terms of time and effort for unit tests in comparison to normal debugging. This is because unit tests make the location of the error immediately apparent.

- It provides a basic documentation of the system. Reading unit tests gives developers a better understanding of the features being tested within the software system.

As a result of these benefits, using unit testing for the system made sense. Unit tests focused on the classes of the system due to their extensive use throughout it. Unit testing on the GUI code was overlooked, since unit testing is difficult to perform on GUI based code, which is normally tested through GUI testing.

Below are the results of the final run of the unit tests. These were run each time a major change to the system was made for the reasons clarified above.

**References**

<https://www.keycdn.com/blog/usability-testing>