

CreateTech

Software Quality Assurance

Task 2: Test Conduction

Ensuring that a product is up to standard and can help customers derive satisfaction from it takes months as there are many problems that can pop up during the production of them and to ensure that most or all these problems are worked out. The principle of conducting tests with a picked sample group to ensure that the issues are mitigated is quite important as not only does it highlight any errors that may be present but to ensure that unforeseen factors to consider as the product will be used by the wider public.

CreateTech as over the past few weeks developed an application that would have the purpose of offering people the luxury of adequately saving money through supplying recipes that have been carefully curated to cater to the tightest budgets especially those student pockets amidst fluctuating food prices and volatile inflation alongside assisting in finding the cheapest options from supermarkets where the app may be launched.

The last task involved careful discussions on how the app would be tested and what element would we be testing to ensure that when we have acquired the necessary feedback, we can move towards fixing the issues that our users found on the app and ensure the final product that we present is not only up to end user standards but also be the desired result from the group's goals which that were set.

Before explaining the testing scenario and process and results, all people that were involved in the testing of the prototype had given prior consent to becoming willing participants and have allowed CreateTech to conduct and utilize the test result in whichever way to extrapolate findings that would serve the purpose of advancing the project.

Whitebox Testing

Whitebox testing is rigid analysis process of evaluating the system development and architecture of a prototype or product. CreateIT, as mentioned in earlier stages of documentation, naming the ICT System Evaluation and Mobile Application ideation tasks, has been constructed within the Visual Studio framework using C#. External plugins such as Visual Code Interaction namespace have also been incorporated into our system. With our Whitebox testing phase, CreateIT was reviewed by not only CreateTech but external candidates, being other students of IT who knew of the basic algorithms and principles in programming as well as Graphical Design to give

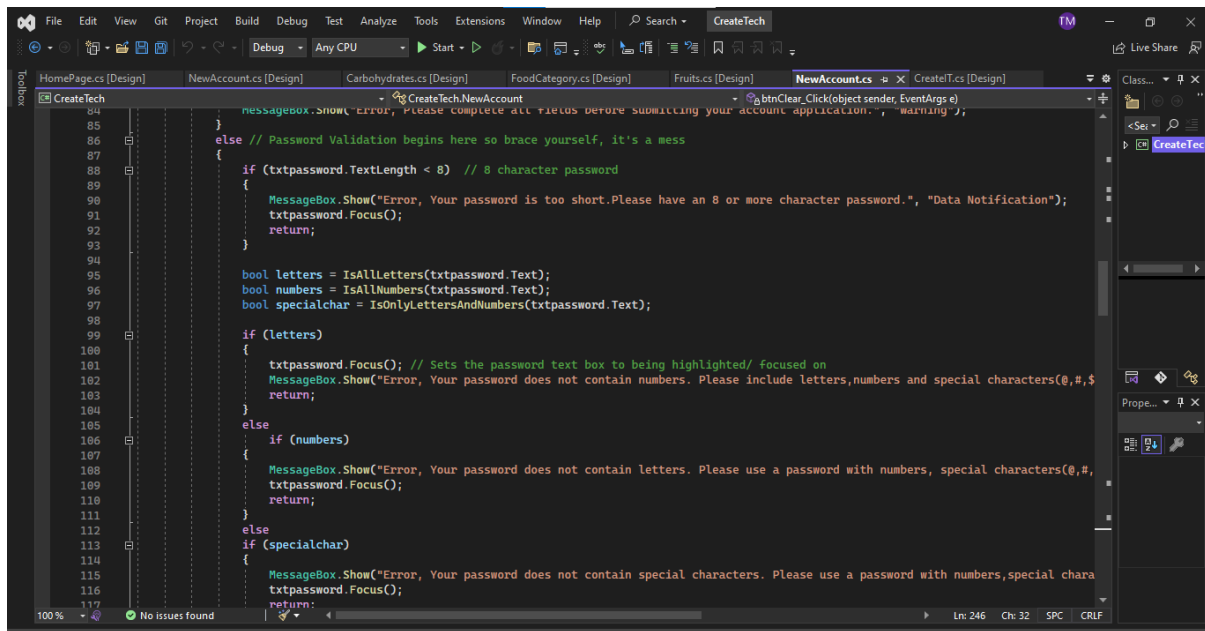
CreateTech feedback on the approach that is currently being applied to the problem at hand. The following section of the document will converge the concepts that were discussed in the *Software Quality Assurance Task 1: Planning the Test*, with the result we managed to receive from our internal and external sources.

The points: 1 and 2 are focused on the system code. After scouting through our C# code for errors and potential glitches, we noticed that majority of our code was not necessarily using the most efficient approaches to implementing certain functionality. An example of an issue we found is that throughout our application, the main Form (page/screen) keeps track of the user that is currently logged in all the time as a public global variable but what was inefficient about this is that the other forms such as the home screen would have to access the user information on the main form through another declared global variable. This may not be seen as a system glitch, but it does cause the resources used by our system to become more clustered in small unneeded pieces. Another issue we encountered is not disposing of variables or resources that had exhausted their life purpose, such as one-time counters keeping track of the number of something.

Point 3, Testing for Device Adaptability: CreateIT is as of now limited to the portrait orientation mode. Unfortunately, we could not find a way to interact with the gyroscope of devices and it was quite an ambitious leap for a first-year project as we would need to recreate every single component on the screen in live-time.

Point 4, Testing for Navigation Response: The transition between screens of our application as of now is decent, only restrained by the fact that the system does have a “flickering” effect when transitioning between one form to the other, this may also occur from time to time when loading options such as the user dashboard.

Point 5, Password Management: Our password management system is solidly formed to ensure that users not only enter the correct password or else they may have to wait to attempt again after a period but upon account creation, our system only allows a password that has 8 or more characters, including but not limited to: letters, numbers and special characters. The password is hidden away from users as the normal convention unless they want to view it for clarification when typing. Please view the C# code snippet of how our system validates and verifies passwords and emails:

The image is a screenshot of the Visual Studio Code editor interface. The top menu bar includes File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, and Help. The status bar at the bottom shows '100%' zoom, 'No issues found', and file encoding 'Ln: 246 Ch: 32 SPC CRLF'. The main editor window displays a C# file named 'CreateTech.NewAccount.cs'. The code is for a 'btnClear_Click' event handler. It starts with a warning message box. Then, it checks if the password length is less than 8 characters. If so, it shows an error message and focuses the password field. Next, it checks if the password contains only letters, only numbers, or only letters and numbers. If it contains only letters, it shows an error message and focuses the field. If it contains only numbers, it shows an error message and focuses the field. If it contains only letters and numbers, it shows an error message and focuses the field. The code is as follows:

```
84: MessageBox.Show("Error, Please complete all fields before submitting your account application.", "Warning");
85:
86: else // Password Validation begins here so brace yourself, it's a mess
87: {
88:     if (txtpassword.Text.Length < 8) // 8 character password
89:     {
90:         MessageBox.Show("Error, Your password is too short. Please have an 8 or more character password.", "Data Notification");
91:         txtpassword.Focus();
92:         return;
93:     }
94:
95:     bool letters = IsAllLetters(txtpassword.Text);
96:     bool numbers = IsAllNumbers(txtpassword.Text);
97:     bool specialchar = IsOnlyLettersAndNumbers(txtpassword.Text);
98:
99:     if (letters)
100:     {
101:         txtpassword.Focus(); // Sets the password text box to being highlighted/ focused on
102:         MessageBox.Show("Error, Your password does not contain numbers. Please include letters, numbers and special characters(0, #, $", "Data Notification");
103:         return;
104:     }
105:     else if (numbers)
106:     {
107:         MessageBox.Show("Error, Your password does not contain letters. Please use a password with numbers, special characters(0, #, $", "Data Notification");
108:         txtpassword.Focus();
109:         return;
110:     }
111:     else if (specialchar)
112:     {
113:         MessageBox.Show("Error, Your password does not contain special characters. Please use a password with numbers, special characters(0, #, $", "Data Notification");
114:         txtpassword.Focus();
115:         return;
116:     }
117: }
```

Point 6, Resource Security Management: Our system as of yet does not maintain information of user interactions therefore as much as this may be a positive in the eye of the customer, for our corporation, we are unable to receive feedback on system usage where we can focus our updates on improving certain system configurations. We will implement a mechanism such as this in later stages of the development process.

Point 7, User-Inclusivity: CreateIT uses a dark mode aesthetic for the numerous reasons, one of them being that dark mode is superior, a principle we believe at CreateTech and another being that we understand the strains white-light can emit on the iris of users with sensitive vision as well as the mass populist of users. A downside we viewed is that we were not able to incorporate a feature that also caters to the blind. We plan on having a voice assistant that will aid those who require this in the nearby future. A good side though is we found that as our application may include the ingredients on delivery, those who are mobility disabled do not need to worry about that aspect of their grocery shopping.

Point 8, Language and Localization: Our application is currently presented in English as a medium of interaction, but we are aware that not all our users will be proficient in English. We have tried to maintain a simplistic but helpful graphical user interface along with the generic prompts and emojis when doing certain actions such as cancelling or signing out. CreateIT, further down the line will also cater to indigenous recipes and ingredients that may only be found in specific native regions rather than only being on a broad national/ international scale.

Point 9, Testing for human errors: While conducting our test, we found that our users were quite excited to test for the system weak points or breaches, after all, it is human nature to see “what happens if I do this”. With that in mind, we viewed our test cases, noting where we may need to reinforce our defences, CreateIT does not only need to protect people from hackers and external attacks, as well as Themselves. One possible suggestion we took to mind is the possible integration of

a “Forgot your password” feature which would email the given email, if it exists within our database (which is a text file as of now) with a unique one-time pin to reset the password.

Black box testing

We conducted a Blackbox testing session based under the previous submission scenario with minor deviations from the written word, the black box test included 4 students from Nelson Mandela University who study in different qualifications from I.T to ensure that the participants were to be unknowing end-users with no prior knowledge of Software-development or any other I.T field.

Testing Scenario

The test was conducted in an enclosed room that allowed for two people, the participants and the person who is conducting the test the participants were tasked with completing different that are related to the application which were:

1. Sign-up and Sign-in to be able utilize the application.
2. Choosing ingredients that were to be used and asking the app to suggest a recipe.
3. Searching for an ingredient to buy it.
4. Creating a shopping list for the ingredients.

The participants were given 10-15 minutes to interact with the application and were given assistance in parts they struggled with and did not understand after which they were asked ten questions about their experience with the application. And lastly asked about suggestions about improvement or modifications the app would do to ensure that it is a comfortable and satisfactory experience.

Testing Questions

1. How was your sign-up experience?
2. How was the experience signing in?
3. How was the response time of the app?
4. Was the language used on the app confusing or did you understand it?
5. How was experience of navigating the app like?
6. Was the app quick to respond to your touch or keyboard strokes?
7. Did the app respond in the desired time and complete your task?
8. How your experience when searching for an ingredient?
9. What can you suggest making the app a more efficient experience?

Results

The results I acquired may not be of a large sample size but the answers I received were quite satisfactory everyone's experience was different which is what we had hoped for this allows for proper analysis of different data as well as suggestions

based on the experience. I have elected to use an average based reporting method by compiling the most common experience with the uncommon one.

Everyone in the testing session didn't struggle with signing and didn't require assistance from the test conductor, while 1 participant struggled to sign-in even though they had entered the correct details while the rest of the group didn't struggle and found the experience comforting, with one of the 3 saying "it was simple as I had all the needed information I had to input."

The testing group was asked to complete the tasks raised at the top and decided to ask them whether overall the response time from the app was satisfactory and all the participants required assistance around this time as it took a few seconds for the page to load and in some instances 2 of the 4 participants reported that the pages were not loading at all.

I followed this up by asking about navigating the application throughout this part they needed assistance especially when it came to adding ingredients to the shopping carts even though they struggled they found that the application was quite straightforward and could choose options correctly. The participant all understood the language that we used for the app and did not require any translation nor further explanation on certain terms used.

The application was able to match choices made correctly and preventing the chances of mismatched options, they were able search the app for certain items as the app directed you to a grocery website.

The participants were able to offer valuable feedback and suggestions on how to make the app a more pleasurable experience, one participant gave feedback regarding the response time and hammered the need to make sure we capture the end-user's attached to the application.