Application: **Web Calculator**

Test areas:

* My test areas are arithmetic operators

* It includes Addition, subtraction, multiplication, division, trigonometric expressions and some equations which are in positive scenarios

* The positive scenarios will give the information that the application is working as per requirement (arithmetic)

* In my script, I have tested whether the number keys in the application are enabled or not. It is the validation to check whether the application is user friendly or not.

* Mostly the calculator is used for the arithmetic purpose. So I tested by using some of the arithmetic operators like "Plus", "Minus", "Multiplication", and "Division".

* Students will also use the calculator to calculate some trigonometric equations. So tested for Sin and Cos values.

* Adhoc testing is very important to complete the testing in all aspects.

* The negative scenarios will give the information that the application is not accepting the things which are not as per requirement.

* It includes some negative scenarios like whether the application is accepting the alphabets and special characters like asterisk(*).

For example; I tested the "plus" operator in Addition test script.

* Here first I checked the launching of application on web successfully.

* Then tested the numbers on the application are enabled. It is to be tested, if the number keys on the application is disabled, it will be not used by the End-user properly.

* So enabling of the keys in the application is important to validate.

* Then tested with number keys with "click" function.

* It is validated because, end-user will use the application by clicking the number keys, operator keys etc to get the result.

* I used "If- condition" to know whether the statement is working properly otherwise it should throw the runtime error. If error occurs, then the tester can debug that.

* If it persists, the tester can raise a defect, so that the developer can fix it.

* In the script, implicit wait is used. To make the script to wait till the element is detected. So that the script will not fail due to "TimeOut".

The errors I find are:

* Miscommunication between the testers and developers.

* After fixing the defect, the tester may not push the updated code and validate the application by repeat the defect is not fixed.

* Testing before understanding the scope and requirements.

* Tester may validate the application without considering the end to end scenario.

* Tester may validate the application by assuming which are not in requirement.

* Creating the test scenarios by his/her assumptions rather than understanding the scope.

Test data and TDD:

* TDD is a development technique that focuses more on the implementation of all features.

* By focusing on the implementation of all features of the application, the test script is developed.

* By using the negative scenarios, the "Error guessing" will be facilitated and this helps in validating the application and provide the bug free application.

* Boolean condition is used to test both True and False conditions.