

# Test Plan

**Goal:** Create test cases for the following simple application:

<https://ahfarmer.github.io/calculator/>

## Test Cases - Q1

**General Precondition:** Access the website <https://ahfarmer.github.io/calculator/> from a computer

### **Test if the buttons are working by clicking**

Steps: 1. Choose a button to click 2. Click on it Test Data: 1	<b>Expected:</b> It affects the display, showing the value chosen
---	--

### **Test if the buttons are working by keyboard**

Steps: 1. Choose a button on keyboard to press 2. Press on it Test Data: 1	<b>Expected:</b> It affects the display, showing the value chosen  <b>Result:</b> Defect, it should work through keyboard
---	--

### **Test if AC (All Clear) is working**

Steps: 1. Choose <b>All Clear</b> to click 2. Click on it Test Data: All Clear	<b>Expected:</b> It clears the display
---	---

### **Test if +/- is working for positive numbers**

Steps: 1. Click on a number 2. Click on +/- Test Data: +/-	<b>Expected:</b> It turns a number into negative
---	---

**Test if +/- is working for negative numbers**

Steps: 1. Click on a number 2. Click on +/- 3. Click on +/- Test Data: +/-	<b><u>Expected:</u></b> It turns a number into positive
--	--

**Test if % is working**

Steps: 1. Click on a number 2. Choose % to click 3. Click on it Test Data: All Clear	<b><u>Expected:</u></b> It turns a number into percentage
--	--

**Test if / is working**

Steps: 1. Click on a number 2. Choose / to click 3. Click on another number 4. Click on =	<b><u>Expected:</u></b> It divides the first number with the second, displaying the quotient
---	---

**Test if / is working for the same value**

Steps: 1. Click on a number 2. Choose / to click 3. Click on the same number 4. Click on = 5. Click on =	<b><u>Expected:</u></b> It divides the first number with itself 3 times, displaying the quotient (cube root)  <b><u>Result:</u></b> Defect, it should outcome $\sqrt[3]{x}$
---	--

**Test if + is working**

Steps: 1. Click on a number 2. Choose + to click 3. Click on another number	<b><u>Expected:</u></b> It adds the first number with the second, displaying the quotient
--	--

**Test if + is working for the same value**

Steps: 6. Click on a number 7. Choose + to click	<b><u>Expected:</u></b> It adds the first number with itself 3 times, displaying the summation
--	---

8. Click on the same number 9. Click on = 10. Click on =	<b>Result:</b> Defect, it should outcome $3x$ or $x+x+x$
--	--

#### Test if $\times$ is working

Steps: 1. Click on a number 2. Choose $\times$ to click 3. Click on another number 4. Click on =	<b>Expected:</b> It multiplies the first number with the second, displaying the product
--	--

#### Test if $\times$ is working for the same value

Steps: 11. Click on a number 12. Choose $\times$ to click 13. Click on the same number 14. Click on = 15. Click on =	<b>Expected:</b> It multiplies the first number with itself 3 times, displaying the product (cube)  <b>Result:</b> Defect, it should outcome $X^3$
---	---

#### Test if $-$ is working

Steps: 1. Click on a number 2. Choose $-$ to click 3. Click on another number	<b>Expected:</b> It adds the first number with the second, displaying the quotient
--	---

#### Test if $-$ is working for the same value

Steps: 16. Click on a number 17. Choose $-$ to click 18. Click on the same number 19. Click on = 20. Click on =	<b>Expected:</b> It adds the first number with itself 3 times, displaying the summation  <b>Result:</b> Defect, it should outcome $x/3$ or $x-x-x$
--	---

### Test the range of numbers

<p>Steps:</p> <ol style="list-style-type: none"><li>1. Click on a number till reach the limit of the display</li><li>2. Check if the number overpassed the canvas</li></ol>	<p><b><u>Expected:</u></b></p> <p>That the display follow the reached number by scrolling</p>
---	---

### Test the range of numbers with zooming out the page

<p>Precondition: setup the smaller size of the screen (zoom it out)</p> <p>Steps:</p> <ol style="list-style-type: none"><li>1. Click on a number till reach the limit of the display</li><li>2. Check if the number overpassed the canvas</li></ol>	<p><b><u>Expected:</u></b></p> <p>That the display follow the reached number by scrolling</p>
---	---

### Test the screen responsiveness (responsive web page)

<p>Precondition: setup the screen for different devices (Toggle device toolbar -&gt; DevTools)</p> <p>Steps:</p> <ol style="list-style-type: none"><li>1. Click on a number till reach the limit of the display</li><li>2. Check if the number overpassed the canvas</li></ol>	<p><b><u>Expected:</u></b></p> <p>That the display follow the reached number by scrolling</p>
--	---

## General Flow for QA - Q2

Following a flowchart with the steps where each task will be addressed to, I show you my QA Test Plan proposal:

### **Phase 1: With Requirement Documents**

- Validate the requirements (for example, user stories) by giving comments, coming up with some points
- Evaluate with a detailed analysis the documentation and description of a new feature in order to determine if it's possible to create Test Cases from it
- If it's possible, report a bug/improvement for Documentation

### **Phase 2: Planning**

- Definition of the types of tests that will be performed (it can be more or less complex) on Web, API, Mobile, etc
- Functional(Acceptance Testing) and non-functional testing
- Select the modules/features that will be automated
- Code/modify the Existing Automation (depending on the stack)
- Performance Testing
- Manual testing
- Definition if there will be Regression Testing (Full or Partial -> Smoking Test )
- Define time estimation (Manual and Automated tests)
- Tests Cases (Writing and Creation)
- Infrastructure for testing and environment

### **Phase 3: Feature or bug fix released for testing**

- Initializing all the process of performing the Testing Plan for a specific product/sprint
- Perform Tests Cases
- Running Regression Test Suite
- Report bugs and defects always relating it to the test case performed to keep the traceability
- It can be many iterations through the development process and the type of task, for example: bugfix, new features, hotfix for any defect got in regression tests,etc

## Phase 4: Final Validation

- Map all the tasks and their status through a management tool. Ex: Confluence (Jira)
- Results and Reports
- A document or presentation for bringing the outcome of the tests for each sprint
- Bring the aspects to be evaluated, for example: downsides, upsides, impediments (Test environment status, Breaking Changes), improvements, etc

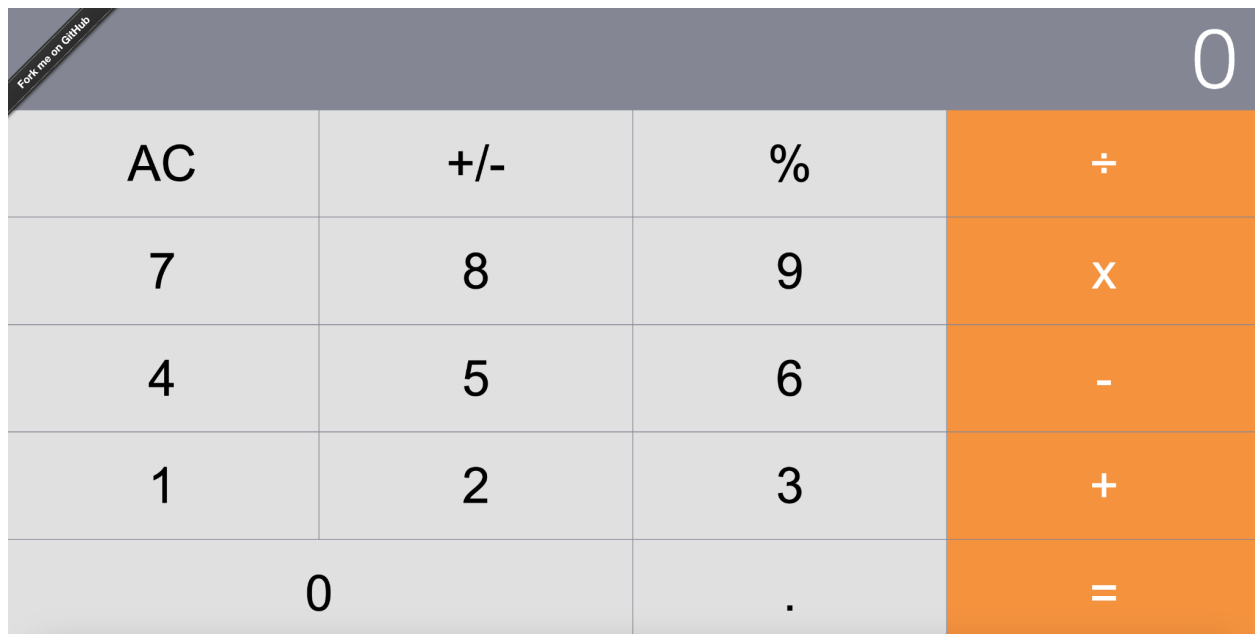
## Example of Test Plan

### Introduction

This sprint, there will be a simple calculator. The behavior of the application must follow what was defined by Product.

This TP describes the test plan for the feature testing by the Quality Assurance team.

### Application



The image shows a calculator application interface. It features a dark grey header bar with a 'Fork me on GitHub' logo on the left and a large '0' on the right. Below the header is a grid of buttons. The first three columns are light grey, and the fourth column is orange. The buttons are arranged as follows:

AC	+/-	%	÷
7	8	9	x
4	5	6	-
1	2	3	+
0	.	=	

## Goal

This document describes the test planning for the new application - Calculator. Hereby, there will be the following goals:





- Identify all the information already available about it as well as the flowchart and Use Cases understanding to be tested
- List all the recommended test requirements
- Recommend and describe the test strategies to be used
- List the modules/features from it to be tested and sorted (prioritization)

## Scope

This Test Plan outlines Unit, Integration and System Testing that will be performed in the application. It is critical that all the modules from the system must be tested, as well as the system performance in this moment beforehand

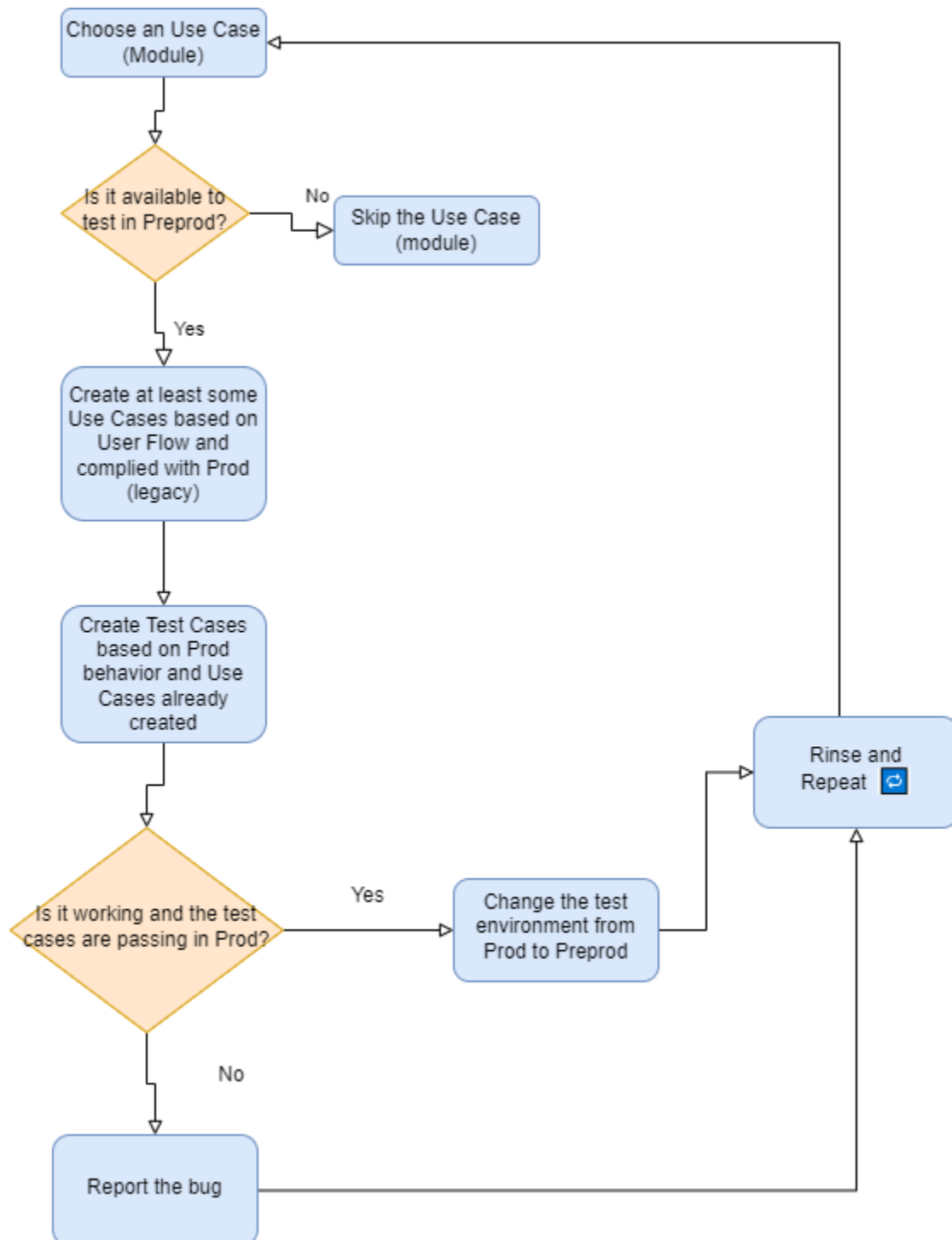
## Test Strategy

This section will define the main workflow, prioritization order, types of testing, tools and techniques on testing to fulfill properly the test scope already defined.

	All test case covered and passed.
	One or more test steps failed. Bug is assigned.
	Features that may be blockers
	Testing

- Test the UI for the buttons
- Test the display for different devices
- Test the functionality for each button
- Test the functionality of the operators (acceptance testing)
- Boundary testing for the values (threshold, limit of characters)

## Workflow





## **Delivery Deadline (Time Estimate)**

To be defined

## **Test Environment**

Tests are to be performed in a pre-production environment containing all the product functionalities that will be released with this feature. Tests are done at system level.

## **Types of Testing**

Functional Testing (Manual) : It will cover all the functionalities/modules from web (Frontend)

Functional Testing (Automatized) : It will cover all modules/methods and it aims to code and modify the source code that already runs in QA.

Non Functional Testing: It will be performed after manual and automated functional testing

## **Testing Techniques**

Performance Testing : Assure that platform supports many requests at the same time (concurrently) and its resources usage, for example scaling it up to evaluate responsiveness and stability under a particular workload

**Tools:** Python, Locust, Postman and JMeter

Security Testing : It will test the vulnerability level from platform, a process intended to reveal flaws in the security that may include elements of confidentiality, integrity, authentication, availability, authorization and non-repudiation

Regression Testing: Assure that all functionalities (already implemented) will work and will keep on as before, without breaking changes. In this case, a test suit will be arranged to cover all the modules

**Tools:** MySQL, Python, Robot Framework and Selenium