./

GENESIS - Learning Outcome & Mini-project Summary Report



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **To be Approved** | **Remarks/Revision Details** |
| 1 | 07-12-2020 | Pranav V Bharadhwaj |  |  |  |
|  |  | Dimanth G S |  |  |  |
|  |  | Ruchitha Pagadala |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Details**

Contents

[TEAM ACTIVITY – SIMPLE CALCULATOR 5](#_Toc58232951)

[**1)** **Introduction** 5](#_Toc58232952)

[**1.1) Using Agile** 5](#_Toc58232956)

[**1.2) Using V Model** 6](#_Toc58232975)

[**2)** **Requirements:** 7](#_Toc58232978)

[**2.1) High Level Requirements:** 7](#_Toc58232979)

[**2.2) Low Level Requirements:** 7](#_Toc58232980)

[**3)** **SWOT Analysis** 8](#_Toc58232981)

[**4) 4W1H:** 8](#_Toc58232982)

[**5.1) Class diagram** 9](#_Toc58232983)

[**5.2) Deployment diagram** 10](#_Toc58232984)

[**5.3) Object diagram** 10](#_Toc58232985)

[**5.4) Package diagram** 11](#_Toc58232986)

[**5.5) State diagram** 11](#_Toc58232987)

[**5.6) Composite diagram** 12](#_Toc58232988)

[**5.7) Use case diagram** 12](#_Toc58232989)

[**5.8) Sequence diagram** 13](#_Toc58232990)

[**5.9) State Machine diagram** 13](#_Toc58232991)

[**5.10) Interaction diagram** 14](#_Toc58232992)

[**6.1) High level Testing** 14](#_Toc58232993)

[**6.2) Low Level Testing** 15](#_Toc58232994)

# TEAM ACTIVITY – SIMPLE CALCULATOR

## **Introduction**

## 

## A calculator is a mobile app that performs arithmetic operations on numbers. The simplest calculators can do only addition, subtraction, multiplication, and division. More sophisticated calculators can handle exponential operations, roots, trigonometric functions, and hyperbolic functions. Internally, some calculators perform these functions by repeated processes of addition. Portable, battery-powered calculators are popular with engineers and engineering students. The calculator we have designed will have

## Simple Calculations like addition, subtraction, multiplication, division and modulo division.

## Simple Operations like nth power of a number, square root of a given number, factorial of a number and multiplicative inverse of a number.

# Using Agile

## **Theme**

Calculation

## **Epic1**

Simple Calculation

## **User Stories1:**

## As an accountant.

## I want to add 2 numbers.

## I want to see the result with less time.

## **Test Case:**

## Given 2 numbers 3 and 5.

## When I add them.

## Result should be 8 with no time.

## **User Stories2:**

## As a primary school teacher.

## I want to teach students about basic addition, subtraction, multiplication ,division, power ,modulo ,square root, factorial, inverse, currency, length, time.

## I want to get the results in less time.

**Epic2:**

Easy Set up and usages.

## **User Stories:**

* As a Student.
* I want to reset the calculator
* So that I can start fresh.

## **Test Case1:**

* Given I am in the middle of an operation.
* When I press CE (clear everything) key.
* Then the operation should be cancelled.
* And the display should show 0 show that it will be ready for next operation.

## **Test Case2:**

* Given that the display status bar on the display shows ‘M’.
* When I press CE key.
* Then the memory should get cleared and it should not display ‘M’.
* And the display should show ‘0’ for further operations.

# Using V Model

## **Ageing**

|  |  |
| --- | --- |
| Time | Gradation |
| Before | Analog-User had to interact more |
| Now | Simple and digital |
| Future | Voice enabled input |

## **Costing**

|  |  |
| --- | --- |
| Type | Cost |
| Standard | Rs. 250/- |
| Digital | Rs. 450/- |
| Simple | Rs. 890/- |

## **Requirements:**

## **2.1) High Level Requirements:**

|  |  |
| --- | --- |
| ID | Description |
| HL\_01 | The calculator has the following keys: 0..9, ., +, -, \*, /, ±, =, C, CE |
| HL\_02 | Calculator should display correct output |
| HL\_03 | The calculator is developed using standard C language and should run on all machines supporting gcc compiler. |
|  |  |
|  |  |

## **2.2) Low Level Requirements:**

|  |  |
| --- | --- |
| ID | Description |
| LL\_01 | If the calculations are if not possible the calculator must display information helping the user to resolve the issue |
| LL\_02 | On encountering divide by 0 operation and operations on imaginary numbers calculator should display error message |
| LL\_03 | Should exit when the user tries to enter the choice more than the Specific choice given. |

## 

## **SWOT Analysis**

|  |  |
| --- | --- |
| **Strength**  1. Perform basic arithmetic operations  2.Able to perform calculations for both  positive and negative numbers  3. large number calculations | **Weakness**  1.basic knowledge needed  2.Continuous input should be provided |
| **Opportunities**  1. Complex functions can be further enhanced  2.Bigger arithmetic calculations can be done | **Threats**  1.Fault results for invalid inputs  2.infinite number display problem  3. imaginary numbers can’t be displayed |

# 4W1H:

**What:**

A simple calculator is a type of electronic calculator, usually but not always handheld, designed to calculate problems in science, engineering, and mathematics.

**Why:**

Because Modern days calculator includes high end engineering and mathematical computation capabilities.

**Where:**

It is mainly used in educational and professional settings.

**When:**

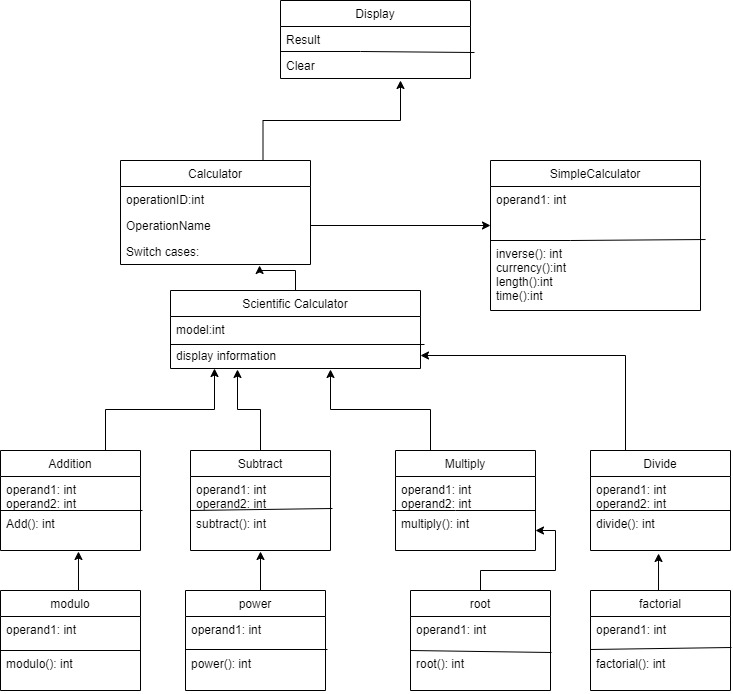
Simple calculators are used widely insituations that require quickaccess to certain mathematical functions especially those that were once looked up in mathematical tables such as trigonometric functions or logarithms**.**

**How:**

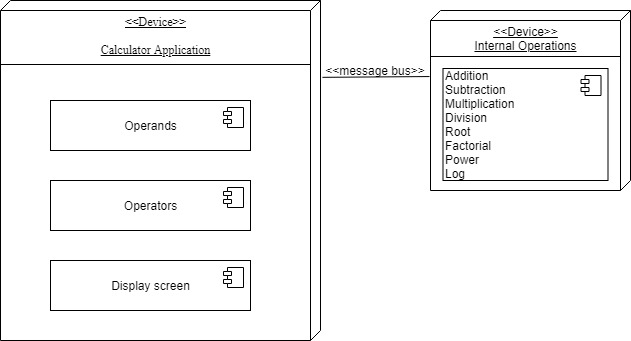
The software inside simple calculator is capable of doing basic functions of arithmetic operations, logarithms, currency, inverse, length, modulo, power and time.

1. **UML Diagrams**

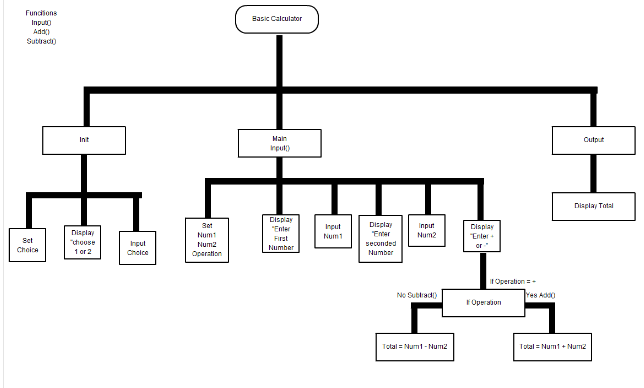
## **5.1) Class diagram**



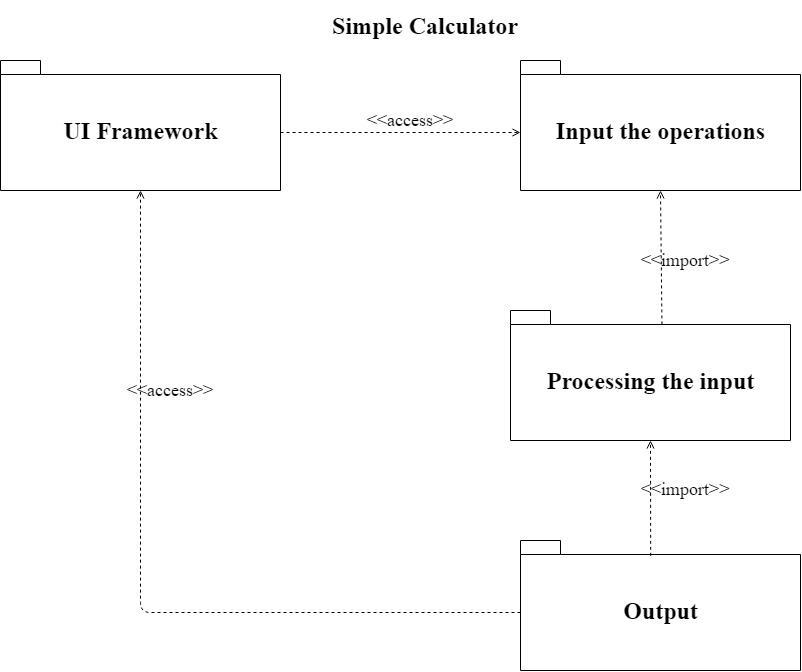
## **5.2) Deployment diagram**



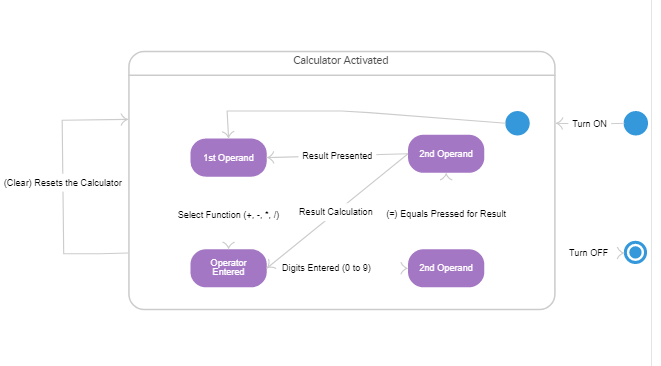
## **5.3) Object diagram**



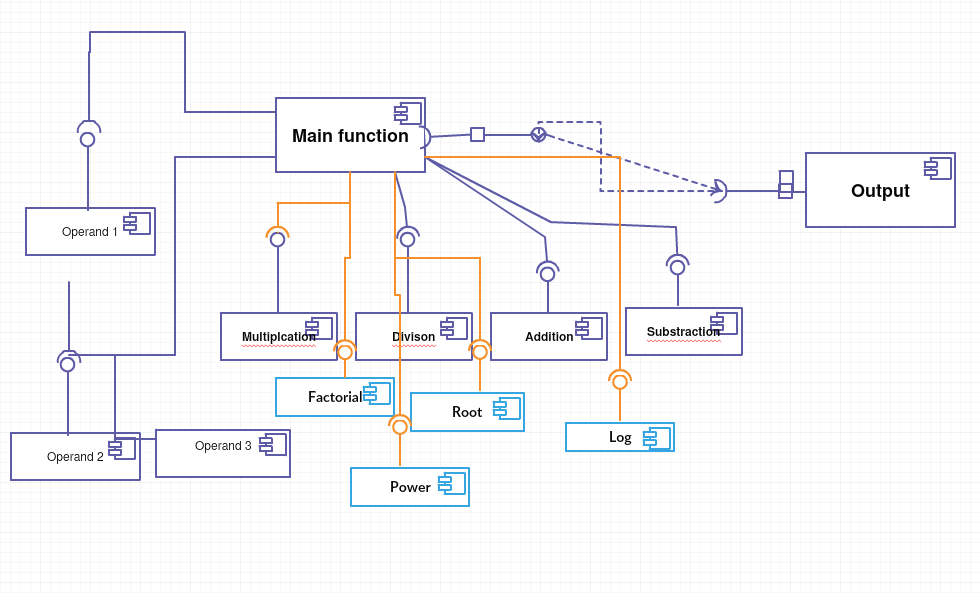
## **5.4) Package diagram**



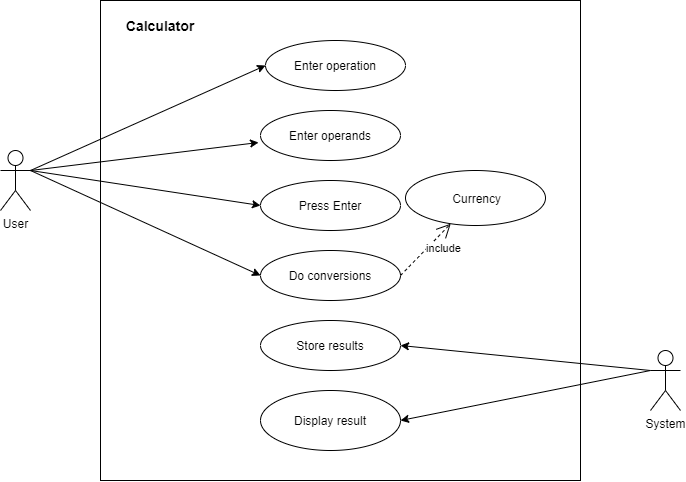
## **5.5) State diagram**



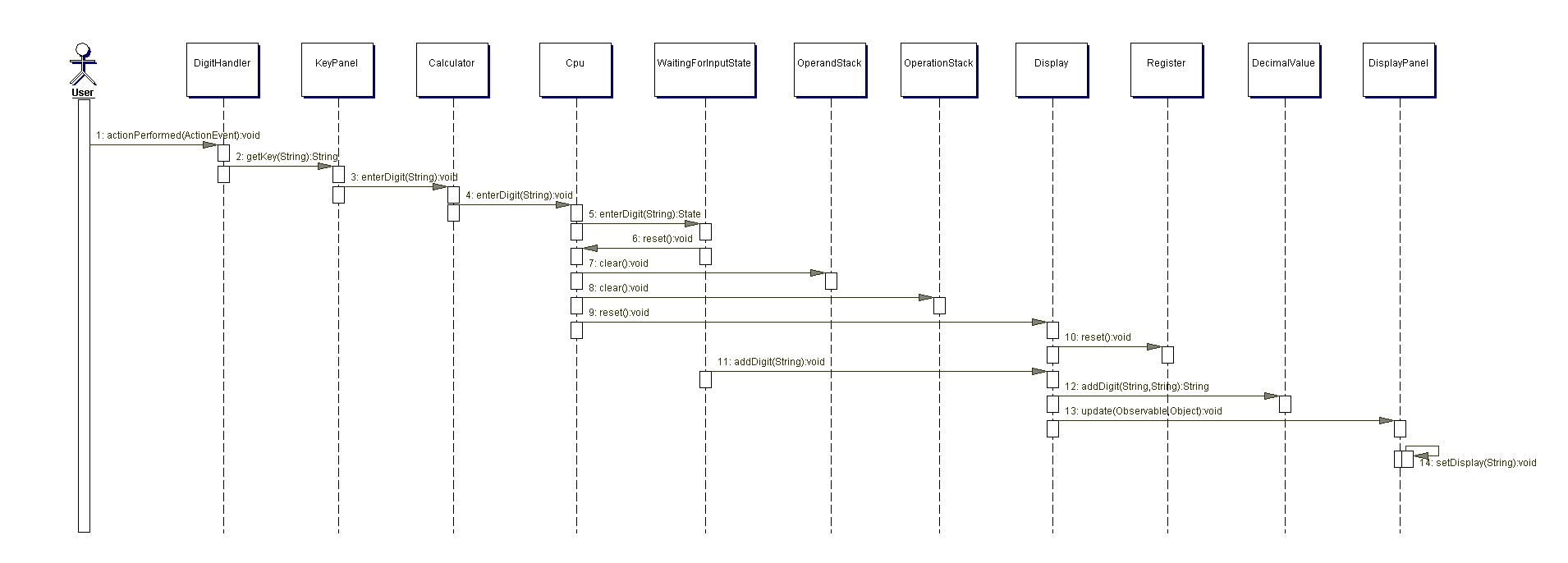
## **5.6) Composite diagram**



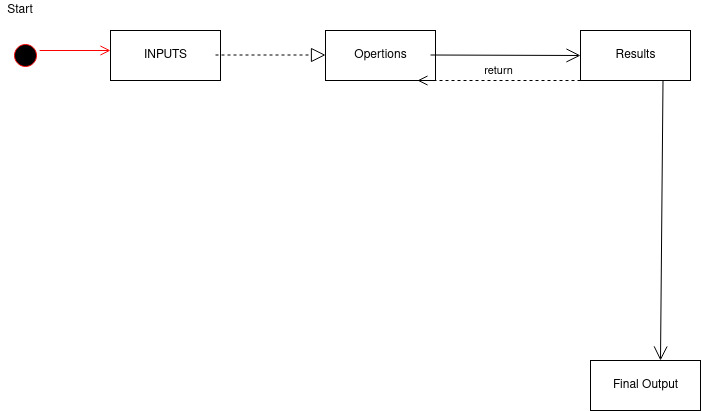
## **5.7) Use case diagram**



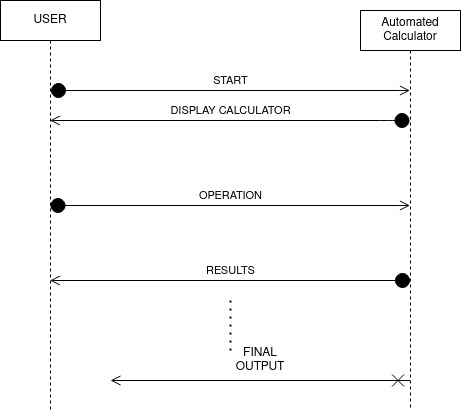
## **5.8) Sequence diagram**



## **5.9) State Machine diagram**



## **5.10) Interaction diagram**



1. **Test Plan**

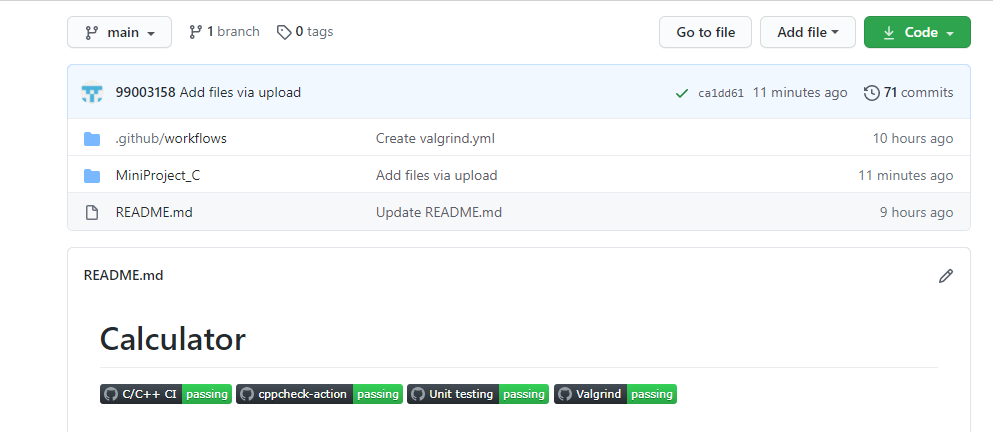
## **6.1) High level Testing**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Description | Precondition | Expected Input | Expected Output | Actual Output |
| HL\_01 | Input method | Accept the input from key | Correct input | Display accepted input |  |
| HL\_02 | Perform math operation | Correct calculation | Correct operands and operators are clicked by the user | Display result of performed operation |  |
| HL\_03 | display | Display output | Display output and appropriate error message | Display result with clear visibility |  |

# 6.2) Low Level Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Description | Precondition | Expected Input | Expected Output | Actual Output |
| LL\_01 | Divide by 0 | denominator for the division operation is zero | Give input for the denominator as zero | Error message |  |
| LL\_02 | Exit | Check condition | Correct input | When user tries to enter choice more than specific choice |  |

1. **Git Badges**



1. **Git References**

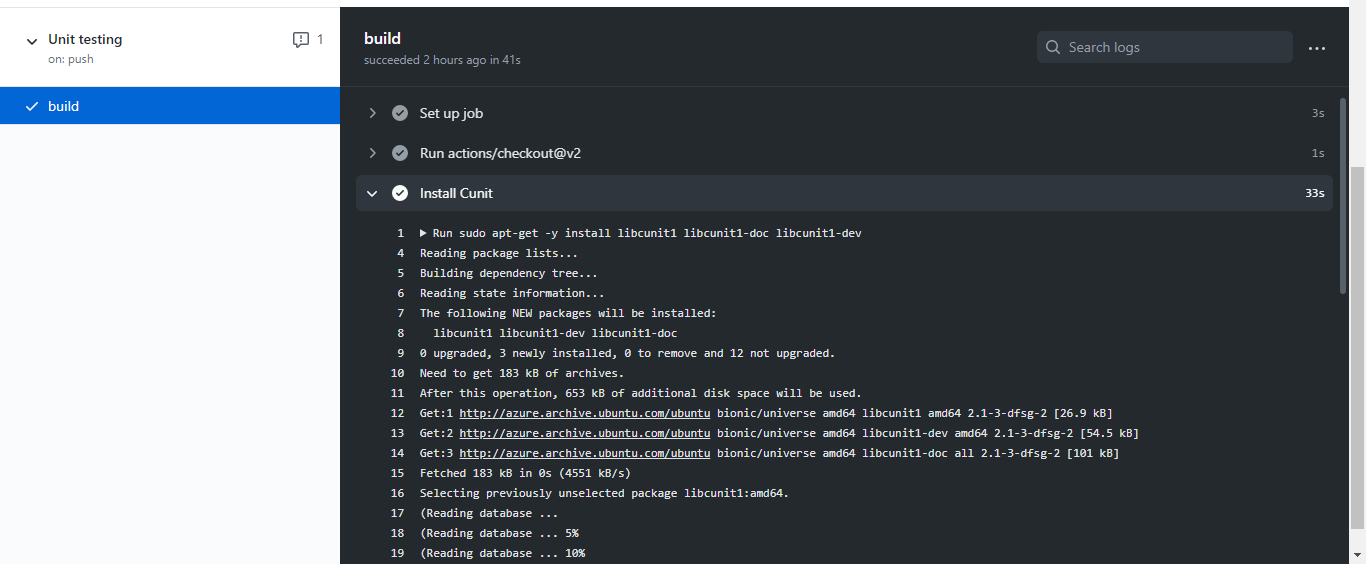
![C/C++ CI](<https://github.com/99003158/Calculator/workflows/C/C++%20CI/badge.svg>)

![cppcheck-action](<https://github.com/99003158/Calculator/workflows/cppcheck-action/badge.svg>)

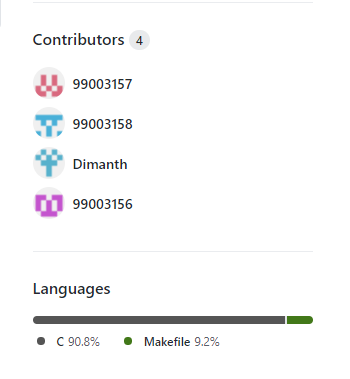
![Unittesting](<https://github.com/99003158/Calculator/workflows/Unit%20testing/badge.svg>)

![Valgrind](<https://github.com/99003158/Calculator/workflows/Valgrind/badge.svg>)

1. **Git Workflows**



1. **GIT Contributors**



1. **References**

[C Program to Create Simple Calculator (tutorialgateway.org)](https://www.tutorialgateway.org/c-program-to-create-simple-calculator/)

[C Program to Make Simple Calculator (codescracker.com)](https://codescracker.com/c/program/c-program-make-calculator.htm)