# ACTIVITY 3-MINI PROJECT

PRODUCT – SMART CALCULATOR

## Define product:

This document specifies the functional requirements for a multi-function calculator program. The program is designed to act like a “handheld calculator” with the usual standard functions (add, subtract, multiply, divide). Additionally, the calculator will have the additional capability of performing functions like squares, cubes, power, maximum, factorial, check prime, average, swap. The program is designed to be as easy to use as a regular calculator.

## Ageing:

**Past:**

Mechanical calculators are used to perform basic arithmetic operations and they are comparable in size to small desktop computers.

**Present:**

Digital and scientific calculator are portable and used to perform calculations ranging from basic arithmetic to complex mathematics. Scientific calculators include trigonometric and statistical calculations.

**Future:**

Voice command input calculators can be developed.

Cost and Gradation**:**

Product cost will increase for every new technology but as the new technologies evolve, the product cost reduces.

Cost of standard calculators: Rs.300-Rs.500

Cost of digital calculators: Rs.500-Rs.800

Cost of scientific calculators: Rs.800-Rs.1000

## Requirements:

|  |  |
| --- | --- |
| ID | DESCRIPTION |
| 1 | The calculator has the following keys 0-9, +, -, \*, /, =, DEL, AC. and operations keys like power, squares, cubes, max, factorial, average, prime, swap. |
| 2 | In any situation, calculator must produce a correct result defined by the well-known arithmetic rules. |
| 3 | If the calculations are impossible the calculator must display information helping the user to resolve the erroneous situation |
| 4 | On encountering a division by 0 the display should read "Math Error" and typing the key “AC” should reset the calculator |
| 5 | On calculating the “factorial” value of a negative operand, the display should read "Wrong operand". |
| 6 | For operation key ‘max’ the fucntions identifies the maximum of two operands and display the highest value operand it is possible for both positive and negative operands |
| 7 | For operation key “swap” the two operands swaps their values. |
| 8 | On identifying a non-prime value, the display should read "Not a prime". |
| 9 | On erroneous operand or operation keys the display should read “Reset (AC) to continue” or “Clear (DEL) to continue” as appropriate. Of course, any situation can be cleared using the main reset key “AC”. |
| 10 | The program shall be written in standard C, as compiled by the GNU GCC compiler (). The program shall use only standard C library functions. The program shall be usable on any system which supports the compiler, and shall not require any particular hardware or software. |

### High-level and Low-level requirements:

|  |  |
| --- | --- |
| H\_01\_L\_01 | The program must have the required keys in their respective types-hence impossible results must be ask by showing some hint or symbols. |
| H\_02\_L\_01 | The correct results must be produced within less amount of time -any device with proper compile shall use this program. |
| H\_03\_L\_01 | The system must have the display to show the respective results |
| H\_04\_L\_01 | Indefinite form |
| H\_05\_L\_02 | Imaginary number |

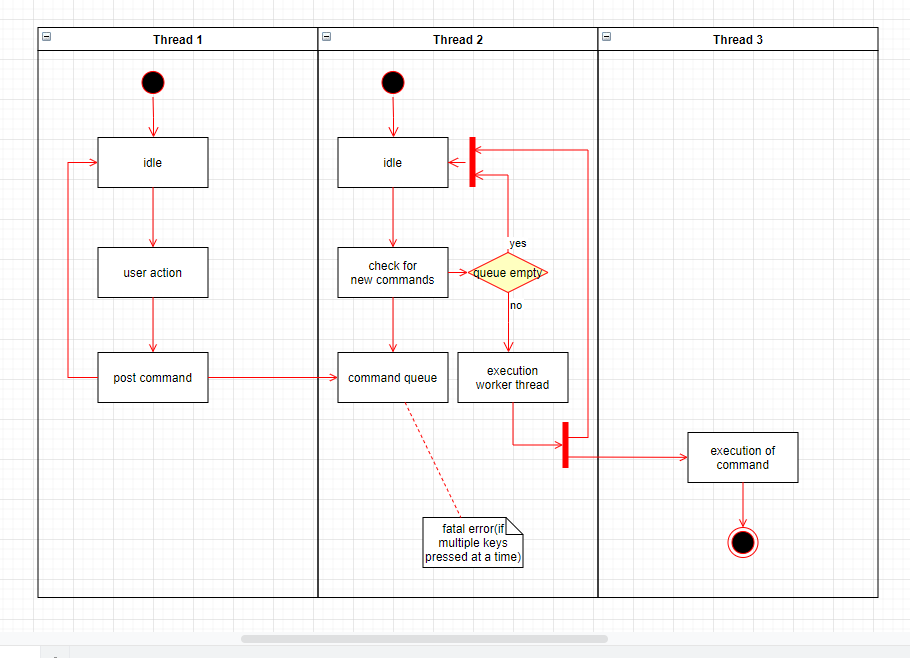
## Design:

This work was centred on design and implementation of simple scientific calculator. This work mainly focused principally on numbers and arithmetic operation. Interestingly, the end result of simple calculator

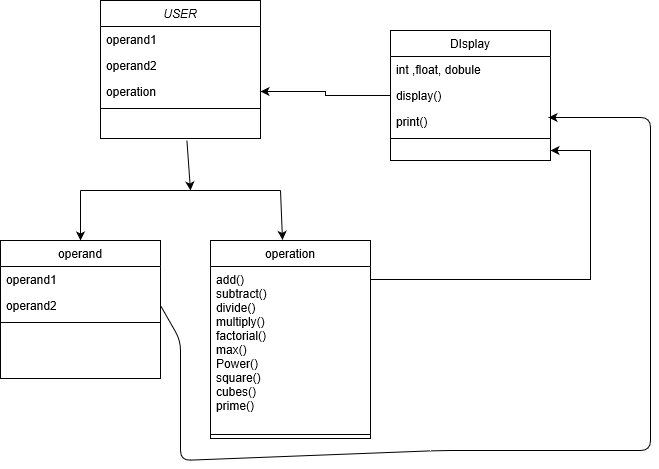
System was its ability to process number and operators and provides a useful result.

### High level design:

Activity Diagram:

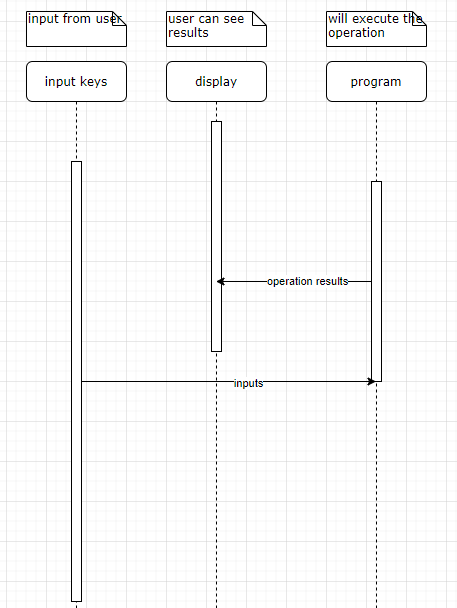


CLASS DIAGRAM:



### Low level design:

Sequence diagram:



## Test Plan:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Description** | **Precondition** | **Expected I/p** | **Expected o/p** | **Actual o/p** |
| High\_1 | Input from user | Accept the input from key | Correct input | Will show the inputs in display | Same as expected o/p |
| High\_2 | Maths operations | Correct Calculation | Correct input symbol as pressed by the user. | Will do correct calculation | Same as expected o/p |
| High\_3 | display | Display output | proper display of output which can be interpreted by human. | will display results clear visibility |  |
| Low\_1 | Format of result | Format of the output can be more than one type | No expected input | Will show error or fatal error | Operation has performed |
| Low\_2.1 | Indefinite number | If the numerator is zero | decimal with number 0 | Error or NA |  |
| Low\_2.2 | Imaginary number | if give the input for root of a negative number. | Root of a negative number | Error or NA |  |
| EX-1 | Addition of 2 numbers | User input | 7 & 3 | 10 | 10 |
| EX-2 | Subtraction of 2 numbers | User input | 5 & 2 | 3 | 3 |
| EX-3 | Division of 2 number s | User input | 3 & 2 | 1 | 1 |
| EX-4 | Multiplication of 2 numbers | User input | 5\*7 | 35 | 35 |
| EX-5 | Square of a number | User input | 8 | 64 | 64 |
| EX-6 | Cubes of a number | User input | 10 | 1000 | 1000 |
| EX-7 | Power of 2 numbers | 1st input is number and 2nd input is exponential factor | 4 & 6 | 4096 | 4096 |
| EX-8 | Factorial of a number | User input | 3 | 6 | 6 |
| Ex-9 | Average of 2 numbers | User input | 5 & 7 | 6 | 6 |
| EX-10 | Swap of 2 numbers | User input | (5,4) | (4,5) | (4,5) |

SWOT Analysis:

|  |  |  |  |
| --- | --- | --- | --- |
| **Strength** | **Weakness** | **Opportunities** | **Threats** |
| Fast calculation | Damaged under 30mm water | Healthy programming practice | Fault result |
| Water proof | not robust and axile | chance to show your skills in c and c++ | Hardware acceleration |
| large calculation | Basic knowledge needed | Business standard ideas | Imaginary and indefinite number or infinite number display problem |
| All arithmetic operations | Need to know prerequisites | Arithmetic operations and logics |  |

## Activity 3 – Agile Aspects

### Theme:

**T1. Accounting.**

**T2. Student.**

### Epic:

**E1.**

* Simple arithmetic operations and large calculations.
* Addition
* Division
* Subtraction
* Multiplication

**E2.**

* All arithmetic operations
* Complex calculation
* Equation solving

### User Stories:

**User Story**1.

**Description:**

* 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 Story2.**

**Description:**

* 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 AC (clear everything) key.
* Then the operation should be cancelled.
* The display should show 0 show that it will be ready for next operation.

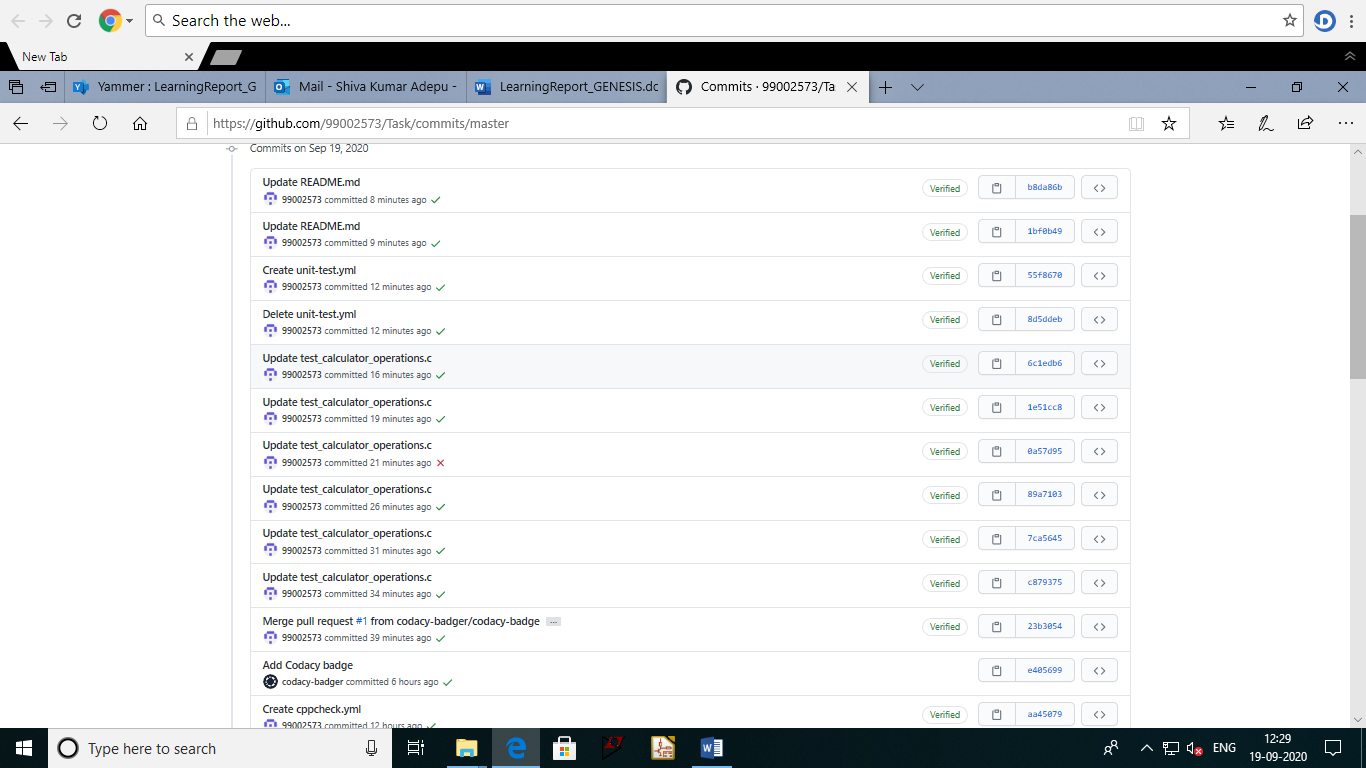
**Test Case2:**

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

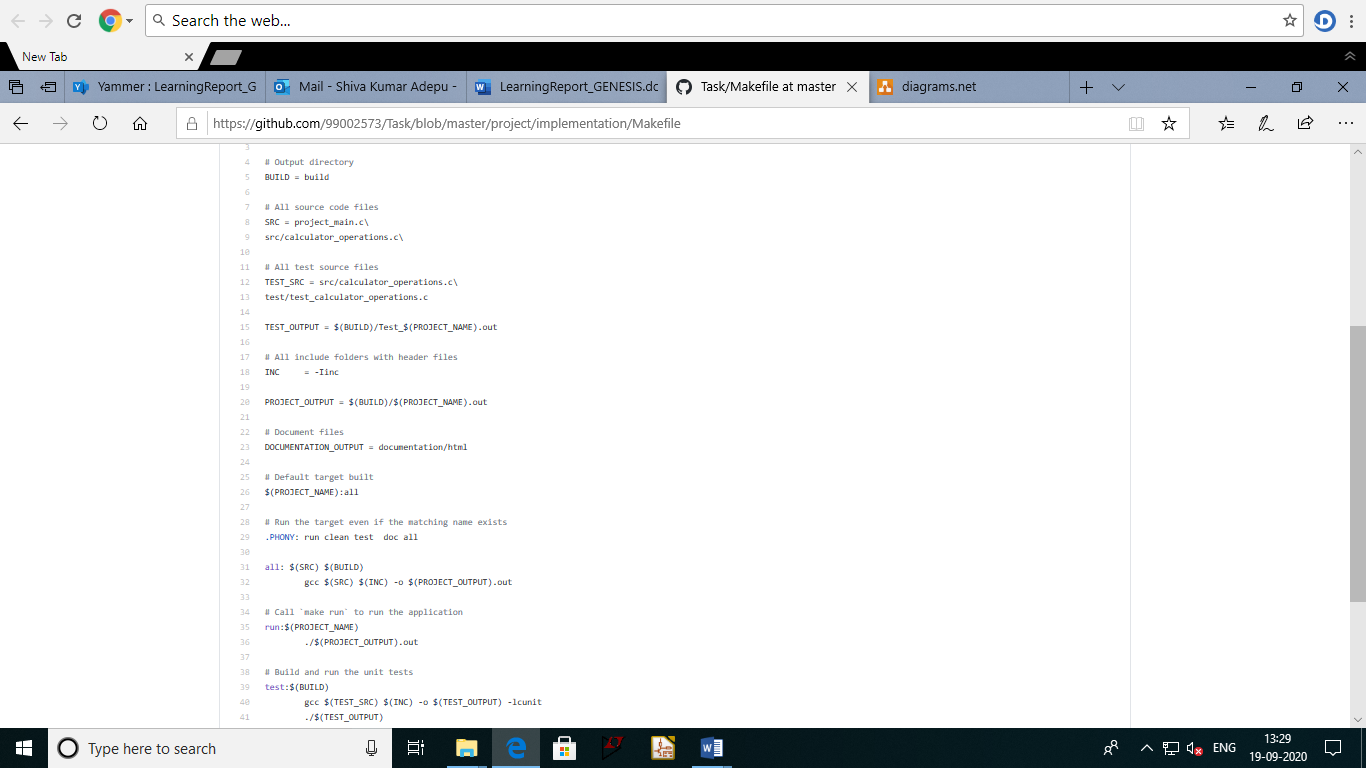
# CI Workflow for C Programming

## GIT:

## COMMITS:

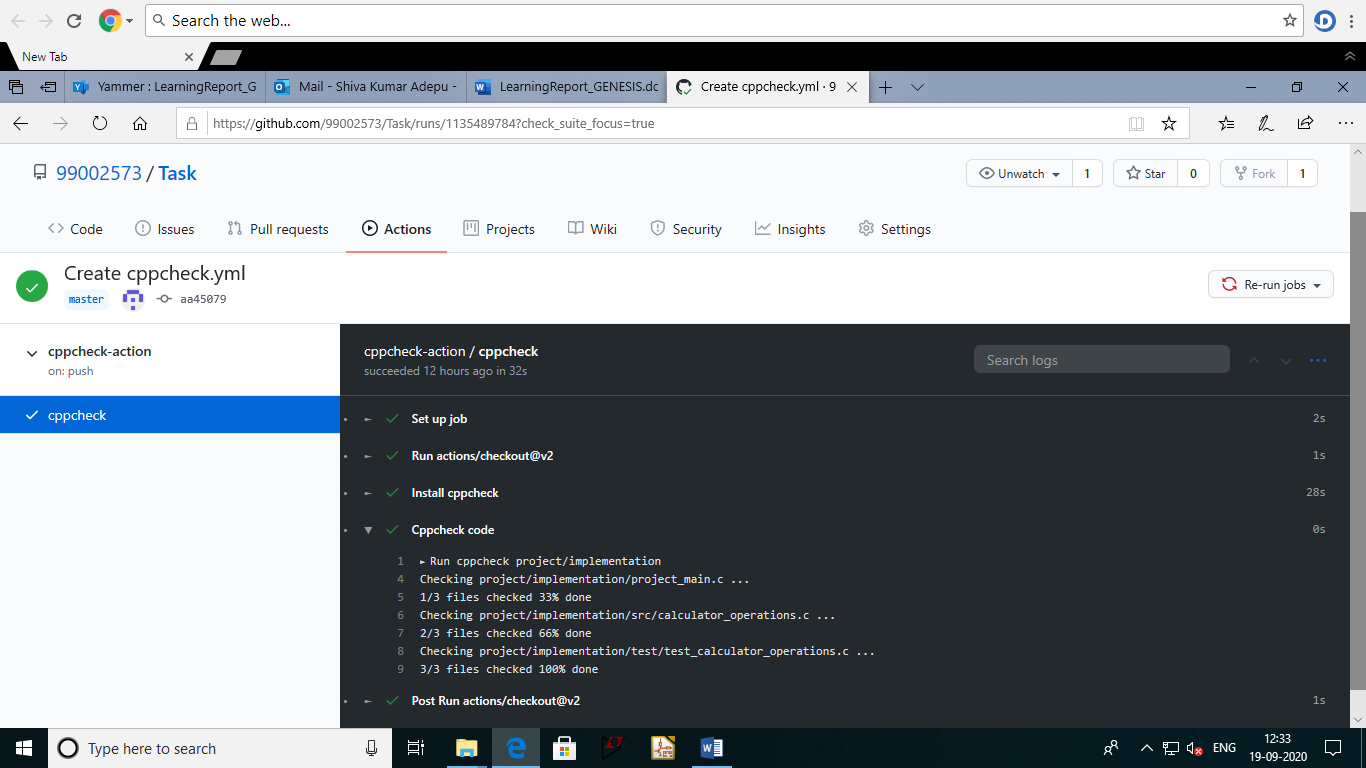


## MAKEFILE:

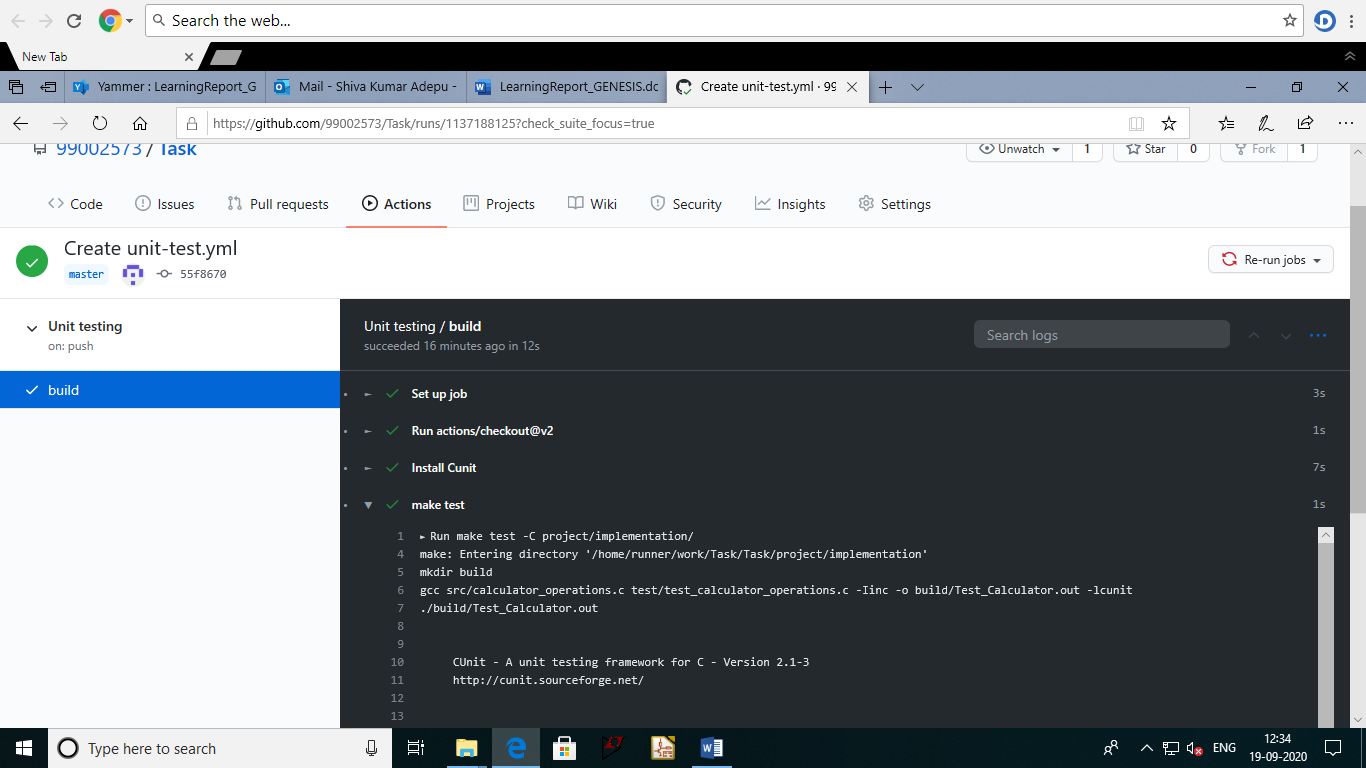


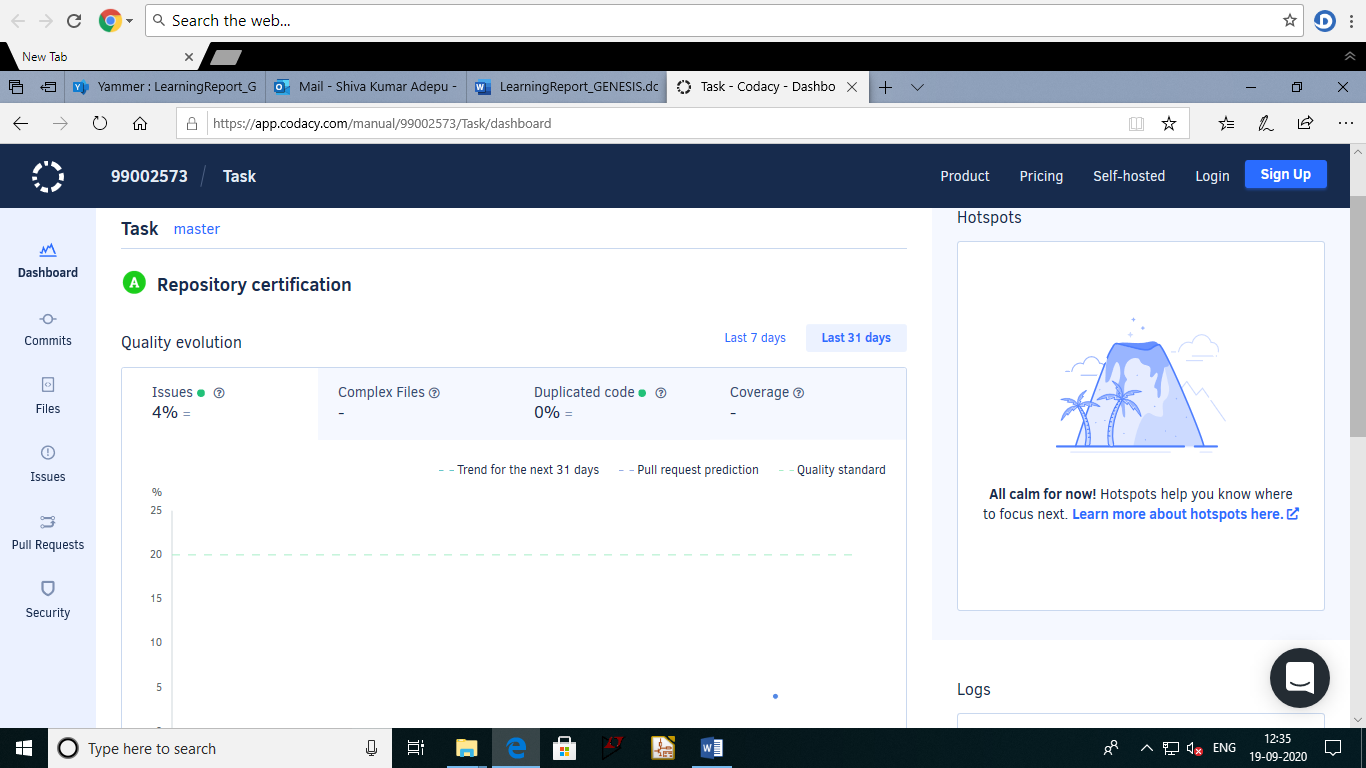
## BUILD:

## CPPCHECK:



## UNIT TESTING:



CODE QUALITY:

## GITHUB Link:

<https://github.com/99002573/Task>

referred: <https://github.com/stepin654321/MiniProject_Template/tree/master/MiniProject_C>