––./

GENESIS - Learning Outcome & Mini-project Summary Report



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **To be Approved** | **Remarks/Revision Details** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Details**

Contents

[Contents 3](#_Toc69543865)

[Mini Project -1 SDLC (System Development life cycle) =>[Team] 5](#_Toc69543866)

[1.1 Modules Used: 5](#_Toc69543867)

[1.2 Topic and Subtopics: 5](#_Toc69543868)

[1.3 Objectives: 5](#_Toc69543869)

[1.3.1 SWOT Analysis: 5](#_Toc69543870)

[1.3.2 4W and 1H: 5](#_Toc69543871)

[1.4 Requirements: 6](#_Toc69543872)

[High Level Requirements: 6](#_Toc69543873)

[Low Level Requirements: 6](#_Toc69543874)

[1.5 Design: 7](#_Toc69543875)

[1.5.1 High Level Diagram of the Calculator 7](#_Toc69543876)

[1.5.2 Permutation and Combination 8](#_Toc69543877)

[1.5.3 Arithmetic operations: 9](#_Toc69543878)

[1.6 Test Plan: 10](#_Toc69543879)

[1.6.1 High Level Test Plan 10](#_Toc69543880)

[1.6.2 Low Level Test Plan 10](#_Toc69543881)

[1.7 Implementation Summary: 11](#_Toc69543882)

[1.8 Video Summary: 11](#_Toc69543883)

[1.9 Git Link 11](#_Toc69543884)

[1.10 Git Dashboard 12](#_Toc69543885)

[1.10.1 GitHub Repo 12](#_Toc69543886)

[1.10.2 Badges 12](#_Toc69543887)

[Summary 12](#_Toc69543888)

[1.10.2 Git inspector summary 13](#_Toc69543889)

[1.10.3 Build 15](#_Toc69543890)

[1.10.4 Code quality 16](#_Toc69543891)

[1.10.5 Git issues 18](#_Toc69543892)

[1.10.6 Unit Testing 18](#_Toc69543893)

[1.11 Individual Contribution & Highlights 19](#_Toc69543894)

[1.12 Summary 20](#_Toc69543895)

[1.13 Challenges faced and how were they overcome 20](#_Toc69543896)

[1.14 Future Scope (If applicable) 21](#_Toc69543897)

[Mini project -2 Embedded C => [Team] 22](#_Toc69543898)

[2.1 Modules Used: 22](#_Toc69543899)

[2.2 Topics and Subtopics: 22](#_Toc69543900)

[2.3 Objectives & Requirements: 22](#_Toc69543901)

[2.4 Components Used: 22](#_Toc69543902)

[2.5 Requirements 23](#_Toc69543903)

[2.5.1 High Level Requirements: 23](#_Toc69543904)

[2.5.2 Low Level Requirements: 23](#_Toc69543905)

[2.6 Design: 23](#_Toc69543906)

[2.6.1 Code snippet for Power Window 23](#_Toc69543907)

[2.6.2 Circuit for Power Window 24](#_Toc69543908)

[2.6.3 Code snippet for Sunroof Control 24](#_Toc69543909)

[2.6.4 Circuit for Sunroof Control 24](#_Toc69543910)

[2.7 Test Plan 25](#_Toc69543911)

[2.7.1 High Level Test Plan 25](#_Toc69543912)

[2.7.2 Low Level Test Plan 25](#_Toc69543913)

[2.8 Implementation Summary 26](#_Toc69543914)

[2.9 Summary 26](#_Toc69543915)

[2.10 Challenges faced and how were they overcome: 27](#_Toc69543916)

[Mini Project -3 Python Programming (Individual) 28](#_Toc69543917)

[3.1 Modules Used 28](#_Toc69543918)

[3.2 Topic and Subtopics 28](#_Toc69543919)

[Core-Topics 28](#_Toc69543920)

[Sub-Topics 28](#_Toc69543921)

[Sub-Topics 28](#_Toc69543922)

[Sub-Topics 28](#_Toc69543923)

[Sub-Topics 28](#_Toc69543924)

[Sub-Topics 28](#_Toc69543925)

[Sub-Topics 29](#_Toc69543926)

[Sub-Topics 29](#_Toc69543927)

[3.4 Objectives: 29](#_Toc69543928)

[3.4.1 Introduction 29](#_Toc69543929)

[3.4.2 4W & 1H 29](#_Toc69543930)

[3.4.3 SWOT Analysis 30](#_Toc69543931)

[3.5 Requirements: 31](#_Toc69543932)

[3.5.1 High Level Requirement Analysis: 31](#_Toc69543933)

[3.5.2 Low Level Requirement Analysis: 31](#_Toc69543934)

[3.6 Design 32](#_Toc69543935)

[3.6.1 High Level Diagram 32](#_Toc69543936)

[3.6.2 Low Level Diagram 32](#_Toc69543937)

[3.7 Test Plan: 33](#_Toc69543938)

[3.7.1 High Level Testing: 33](#_Toc69543939)

[3.7.2 Low Level Testing: 33](#_Toc69543940)

[3.8 Implementation Summary 35](#_Toc69543941)

[3.9 Git Link 35](#_Toc69543942)

[3.10 Git Dashboard 35](#_Toc69543943)

[3.11 Git Summary 36](#_Toc69543944)

[3.12 Summary 36](#_Toc69543945)

[3.13 Challenges faced and how were they overcome 36](#_Toc69543946)

[3.14 Future Scope 37](#_Toc69543947)

[Mini Project -4 Kernel Programming and Device driver =>[Individual] 38](#_Toc69543948)

[4.1 Modules Used 38](#_Toc69543949)

[4.2 Topic and Subtopics 38](#_Toc69543950)

[4.3 Objectives 38](#_Toc69543951)

[4.4 Requirements 38](#_Toc69543952)

[4.5 Test Plan 39](#_Toc69543953)

[4.6 Implementation Summary: 39](#_Toc69543954)

[4.7 Git Link: 40](#_Toc69543955)

[4.8 Git Dashboard 40](#_Toc69543956)

[4.9 Summary 40](#_Toc69543957)

[4.10 Challenges faced and how were they overcome 40](#_Toc69543958)

# Mini Project -1 SDLC (System Development life cycle) =>[Team]

## **1.1** **Modules Used:**

Modules used in this project are SDLC and C programming.

## **1.2** **Topic and Subtopics:**

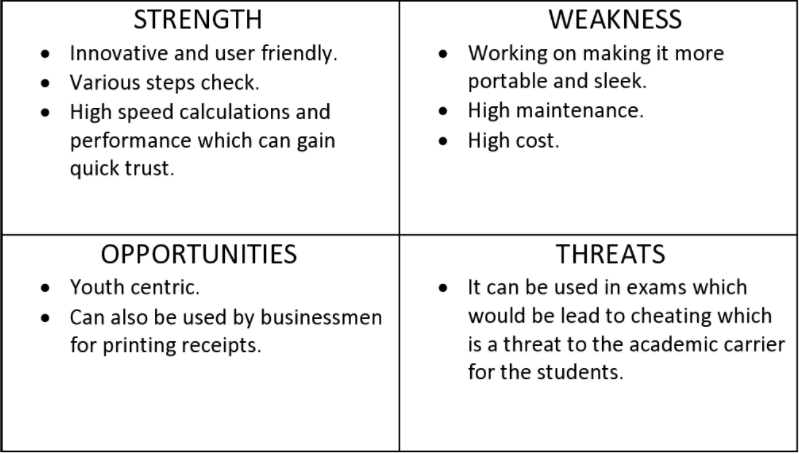
* ***V-Model***
* C Programming
* Makefile
* Unit Testing
* Version Control (via GitHub)
* ***Agile-Methodology***
* Theme
* Epic
* User- Story

All the core-topics and sub-topics are implemented through V-Model.

## **1.3 Objectives**:

Designing a basic calculator that performs basic functions as well as some specific functions as per requirements.

## **1.3.1 SWOT Analysis:**



### 1.3.2 4W and 1H:

**Who:** Student and businessman.

**What:** Smart Scientific Calculator.

**When:** For fast and effective way to complete calculations.

**Where:** Statics comparing the previous data with present data.

**How:** Easy to user interface.

## **1.4 Requirements:**

### High Level Requirements:

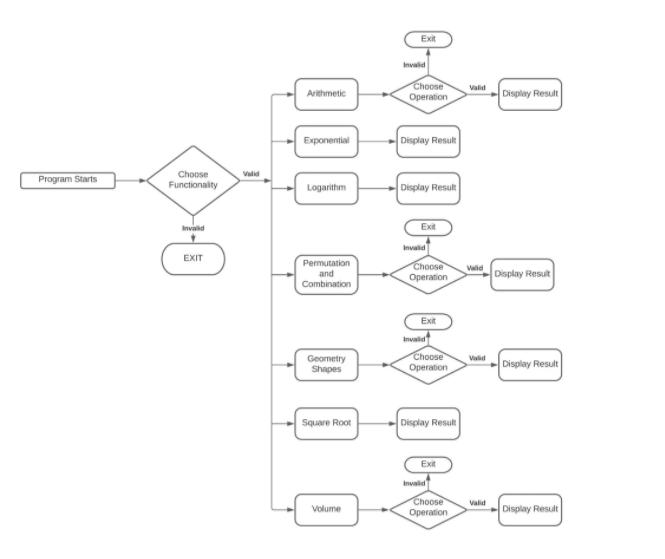
|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Status** |
| 01 | 14-digit screen input. | Implemented. |
| 02 | Dedicated MRC (Memory Recall and Clear). | Implemented. |
| 03 | Dedicated check keys. | Implemented. |
| 04 | Permutation and combination functions are performed. | Implemented. |
| 05 | Volume for cone, sphere, cylinder. | Implemented. |
| 06 | Area for square and rectangle. | Implemented. |
| 07 | Arithmetic operations are performed. | Implemented. |

### Low Level Requirements:

|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Status** |
| 01 | Dedicated ON/OFF switch. | Implemented. |
| 02 | Grand total key. | Implemented. |
| 03 | Decimal key. | Implemented. |
| 04 | Basic math operations keys. | Implemented. |

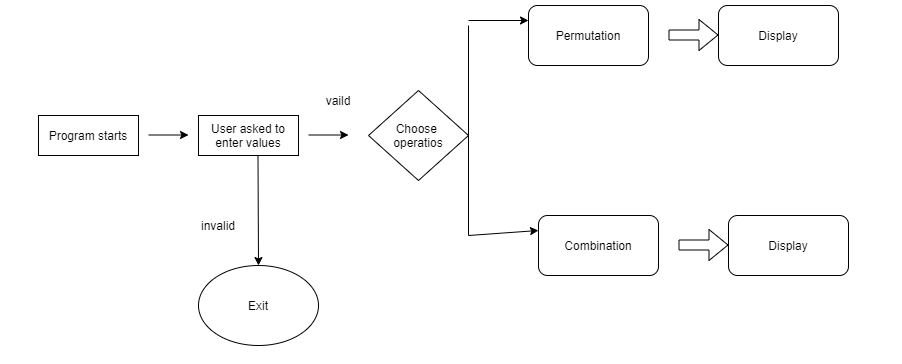
## **1.5 Design:**

### 1.5.1 High Level Diagram of the Calculator

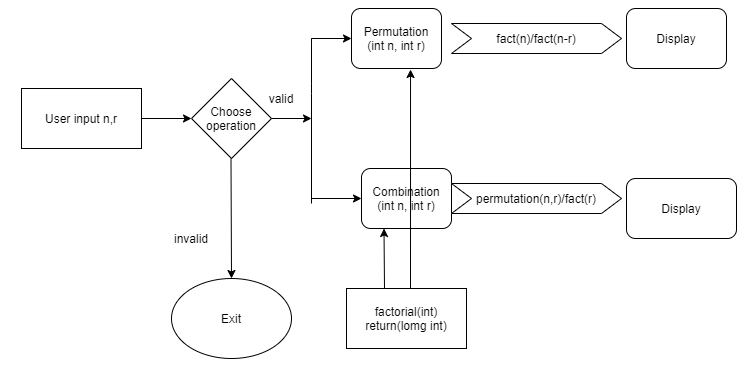


### 1.5.2 Permutation and Combination

#### 1.5.2.1 High Level Requirement:

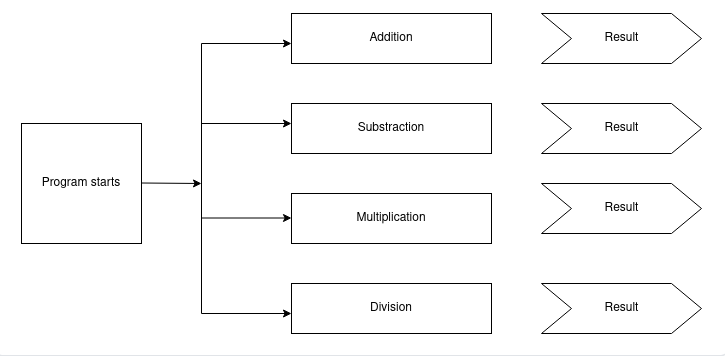


#### 1.5.2.2 Low Level Requirement:

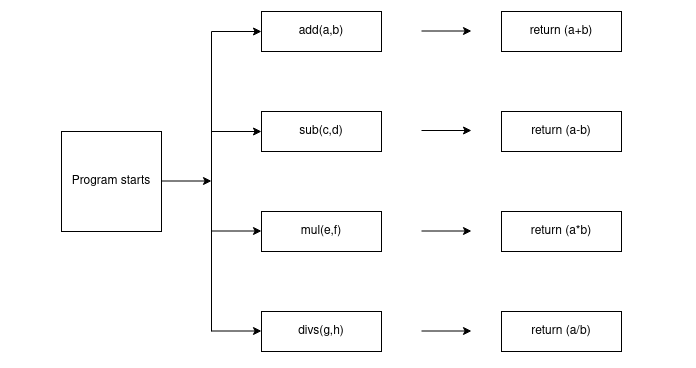


### 1.5.3 Arithmetic operations:

#### 1.5.3.1 High Level Requirement:

****

#### 1.5.3.2 Low Level Requirement:



## **1.6 Test Plan:**

### 1.6.1 High Level Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Description** | **Exp IN** | **Exp OUT** | **Actual OUT** |
| H\_01 | For arithmetic operations, the numbers taken will give positive result | Positive | Positive | Positive |
| H\_02 | For permutation and combination, the values taken will result in zero | N to 0 | 0 | 0 |
| H\_03 | For factorial, the numbers taken will give positive output | 20,10 | 200 | 200 |

### 

### 1.6.2 Low Level Test Plan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Description** | **Exp IN** | **Exp OUT** | **Actual OUT** | **Type of tests** |
| L\_01 | Addition: All the numbers we add will give positive values | 10, 20 | 30 | 30 | Requirement based |
| L\_02 | Subtraction: All the numbers we add subtract give positive values | 20, 10 | 10 | 10 | Requirement based |
| L\_03 | Multiplication: All the numbers we multiply will give positive values | 20, 10 | 200 | 200 | Requirement based |
| L\_04 | Division: All the numbers multiply divide will give positive values | 20,10 | 2 | 2 | Requirement based |
| L\_05 | Permutation: If n will be zero then result will also be zero | N=0 | 0 | 0 | Requirement based |
| L\_06 | Combination: If n will be zero then result will also be zero | N=0 | 0 | 0 | Requirement based |
| L\_07 | Factorial: When we give the input, it gives positive output | 5 | 120 | 120 | Requirement based |

## 

## **1.7 Implementation Summary:**

It is a basic calculator that will allow users to perform operations in Mathematics Addition, Subtraction, Multiplication, Division, Trigonometry, Factorial, Area, Volume etc. However, the input has to be in the form "number1 operator1 number2 operator2 number3" (i.e 2+4\*10). The input values can be from any integer to even a number with decimals. Moreover, this calculator is smart enough to operate multiplication/division before addition/subtraction, in another word it is implemented with the order of precedence logic.

Implementation folder had all source files, header files, test files for different features of the calculator such as Basic Arithmetic, Square root, cube root, exponent, logarithm, etc.

Here, **inc** folder holds all the header files with “.h” extension which contains prototype of all functions, structure definition, macro definition and definition of all the enumerators.

The **src** folder holds all the source files with “.c” extension which has definitions of all the functions whose prototype is defined in header files.

Other than these folders, there is also a **unity** folder which holds prototypes and definition of the standard unity test case functions.

Also, there is a **Makefile** which builds, debugs using valgrind, check static and dynamic code quality, performs overall unit testing for all the codes together with the execution of single commands based on different defined targets.

## 

## **1.8 Video Summary:**

“Please upload a short video on the repo for the walkthrough of the project (Team/Individual) less than 7min and less than 30MB File Size. Start is the Standard opening slide with title of miniproject + Team members followed by the walkthrough”

## 

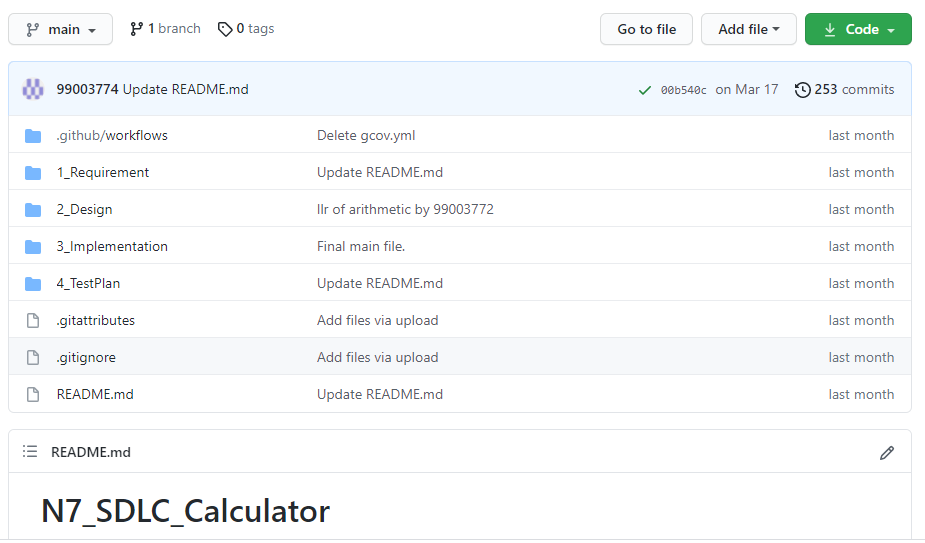
## **1.9 Git Link**

<https://github.com/99003774/N7_SDLC_Calculator.git>

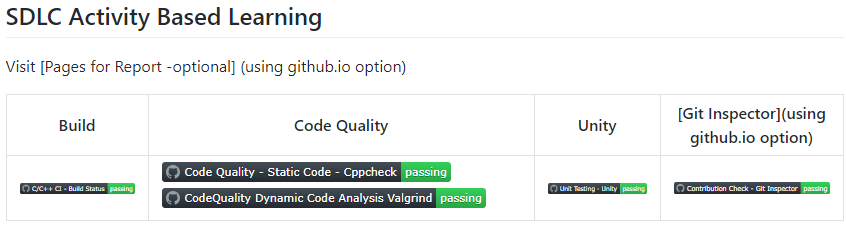
## 

## **1.10 Git Dashboard**

### 1.10.1 GitHub Repo



### 1.10.2 Badges



### Summary

In this project, we mainly-focused on how to design a calculator which is bit different and cost-effective as compared to other calculators present in the market.

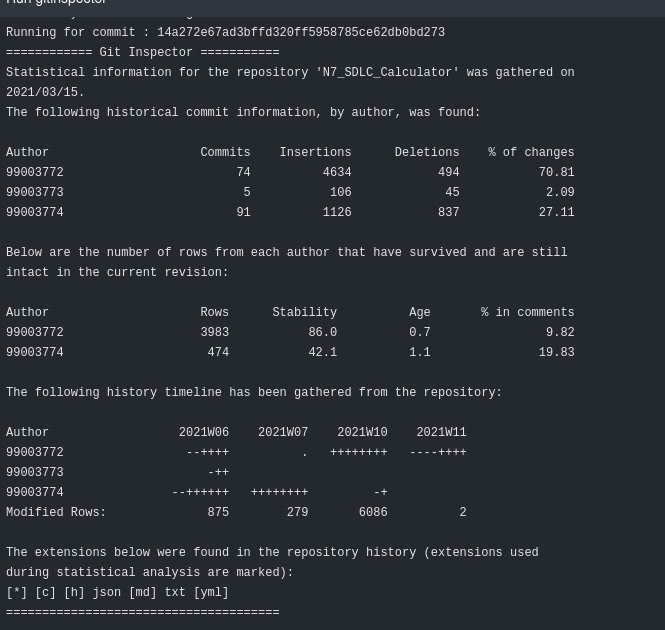
So, for this we first analyzed all the other calculators from low-end feature low cost calculator to high-end feature high-cost calculator and prepared a list of features to include in our modified cost-effective more featured calculator.

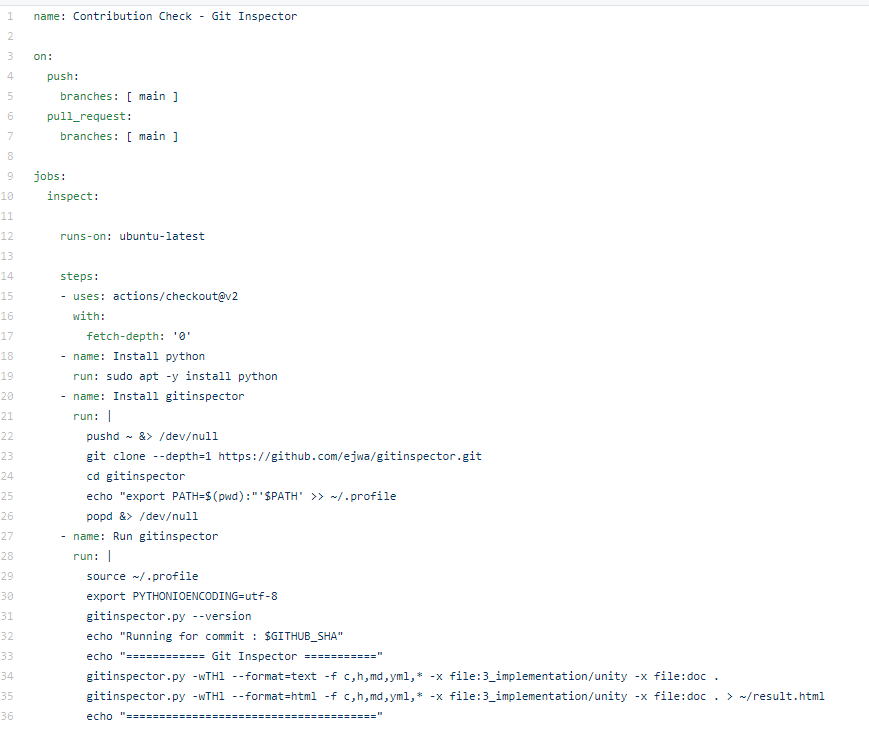
Features included in the calculator are basic arithmetic operations (addition, subtraction, multiplication, division), Permutation and Combination, Geometric Calculations.

This calculator is implemented through C programming. This calculator will be mainly used by the school, college students, businessman, engineers for various purpose. Also, this calculator will be cost effective and if implemented over hardware it cost around 800 – 1000 INR.

### 

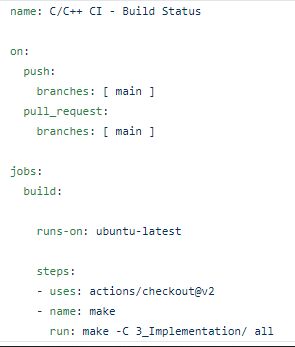
### 1.10.2 Git inspector summary



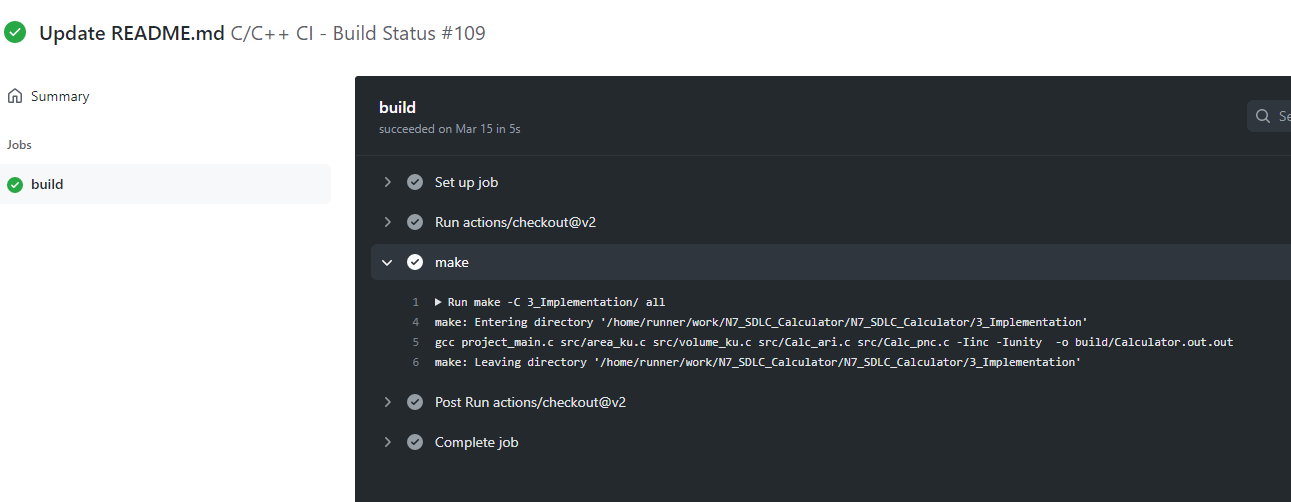


### 1.10.3 Build

#### 1.10.3.1 Setup for Build



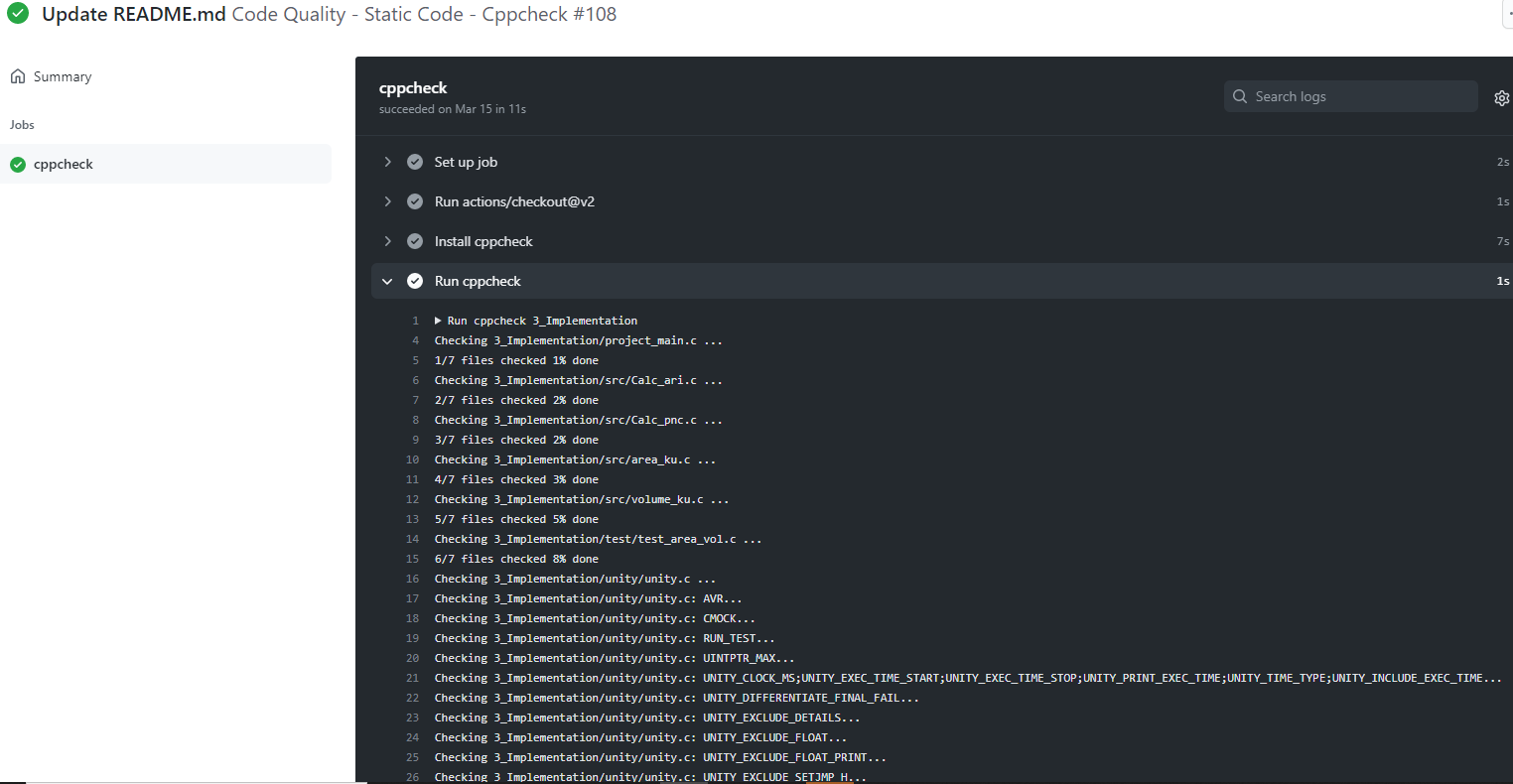
#### 1.10.3.2 Outcome of the Build



### 1.10.4 Code quality

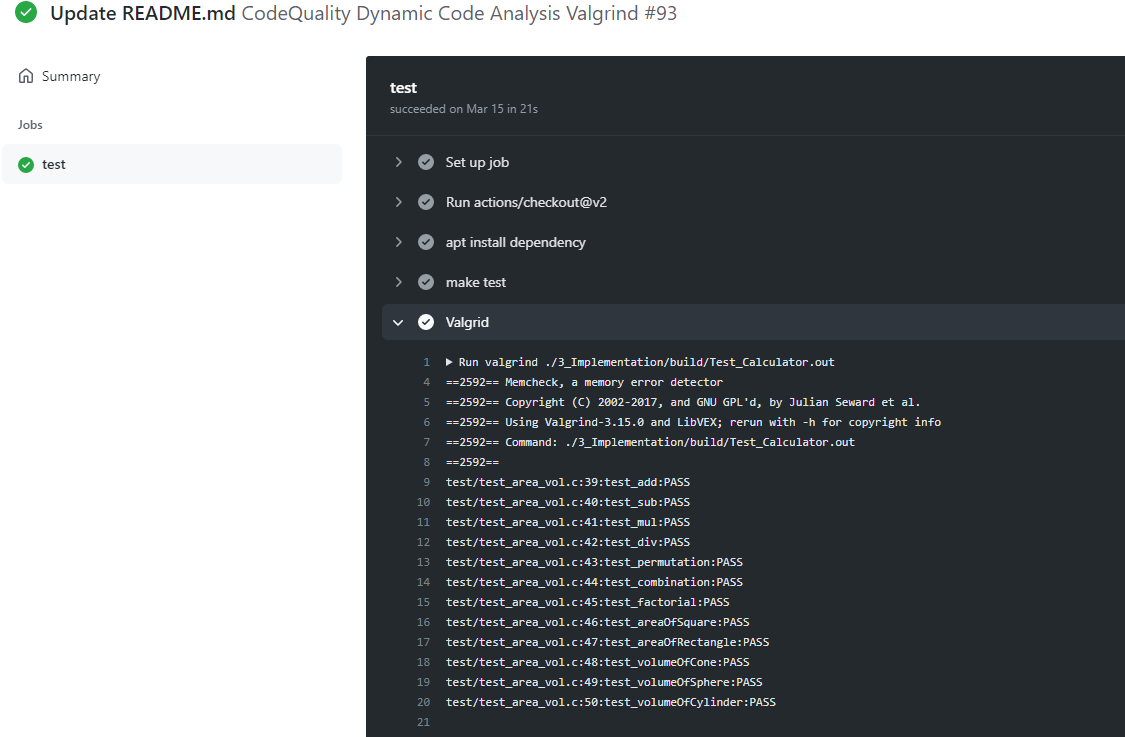
#### 1.10.4.1 Setup for Static Code Quality



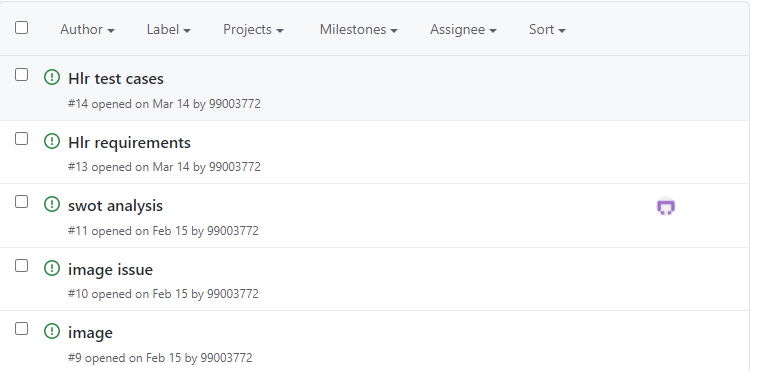


#### 1.10.4.2 Outcome of the Dynamic Cody Quality



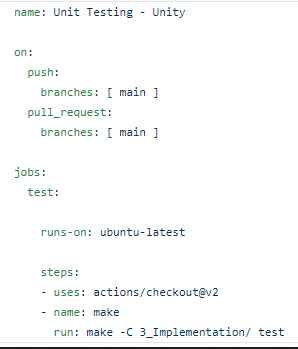


### 1.10.5 Git issues

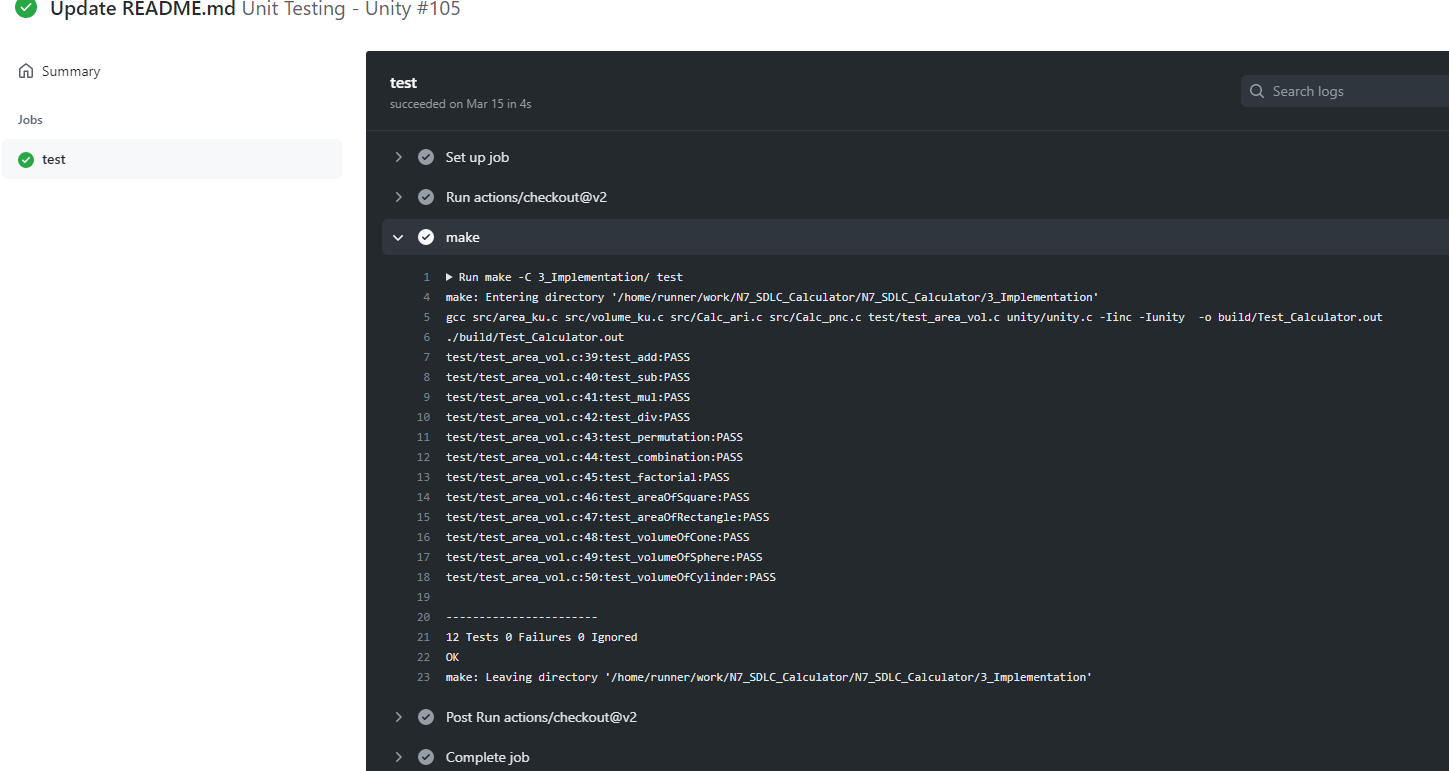


### 1.10.6 Unit Testing

#### 1.10.6.1 Setup for Unity Testing



#### 1.10.6.2 Outcome of the Unity Testing

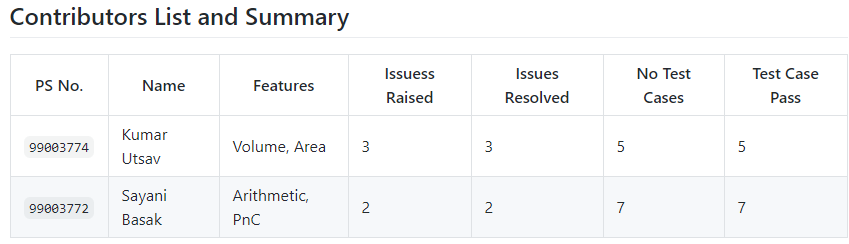


## **1.11 Individual Contribution & Highlights**



* Arithmetic operations, Permutation and Combination features are implemented.
* Test case for the same is implemented.
* High level and low-level test cases is implemented for the same.
* Issue raised and the issue was solved.
* Helped during the workflow’s implementation of the project.
* I have written my code in main.c in src folder under implementation. Accordingly, I have added main.h in inc folder under implementation. I have modified the test\_calculator\_operations.c where I have written my test cases.
* I have modified Makefile so that program should be able to build as well as it is able to perform unity based unit testing.

**Highlights**



## **1.12 Summary**

This project was our first step to work in coordination as a team while working towards our development of our individual skills.

Also, we’re unorganized at the beginning but after planning through V-Model we’re able to analyze and design each scenario and perfectly implemented whole project in the given duration. So, we also learned a great skill called time-management.

* Technical skills developed: -
  + Advanced C programming
  + Make File
  + Unit Testing through Unity
  + Version Control using GitHub
* Soft-Skills Developed: -
  + Team-work
  + Team-management
  + Time Management
  + Assertiveness

## **1.13 Challenges faced and how were they overcome**

* Initially I was unable to run the test cases but now it’s running.
* Differentiation of high level and low level.
* Committing to GitHub, pull and push in GitHub.
* Cpp check and Unity testing.
* Some workflow problems were there but now it has been solved.
* Makefile problem while running in windows but it worked with linux as there persisted an existence problem but it got resolved in linux.
* Git inspector was not working initially but now it’s working.
* Valgrind was not working previously because there was some issue in makefile but now it’s working fine after resolving makefile.

## **1.14 Future Scope (If applicable)**

1) Features like matrix operations can be added.

2) Features such as Physics operations (Acceleration, capacitive reactance, circular velocity, Coulomb’s Law, Gravitational force, projectile motion) can be added.

3) Features like AP, GP sum (AP sum, GP sum) can be implemented in the future.

4) The price of the product is less than other products with same features that are available in the market.

5) The product will also work in banking sectors and other government sectors where they want low price, minimum features, handy products.

### 

# Mini project -2 Embedded C => [Team]

## **2.1 Modules Used:**

Modules used in this project are Embedded Systems and Embedded C Programming and was implemented on the hardware STM32.

## **2.2 Topics and Subtopics:**

* Driver API Development
* GPIO
* ADC
* SPI, UART, I2C
* External interrupt
* Debugging using STM Board
* Driver Development (Hardware Abstraction Level- HAL)

* + GPIO
  + ADC
  + External Interrupt
  + Debugging using STM Board

## **2.3 Objectives & Requirements:**

To implement different CAR Module features using STM32f407VG Microcontroller featuring 32-bit ARM-M4 with FPU core.

* Car Module Features:
* Power window feature [ Done by me]
* Sunroof control feature [ Done by me]
* Interior Lighting using PIR Motion detection sensor
* Door Lock feature
* Seat control feature
* Wiper Control feature

## **2.4 Components Used:**

* STM32f407VG Microcontroller
* Breadboard
* LED
* LDR Sensor
* Soil Sensor
* PIR Motion detection sensor
* RGB Color Sensor
* Potentiometer Sensor
* Ring Buzzer Sensor
* Jumper Wires

## **2.5 Requirements**:

### 2.5.1 High Level Requirements:

1. Controlling movement (upward and downward) of Door window using buttons.
2. Controlling movement (opening and closing) of Sunroof window using push button.

### 2.5.2 Low Level Requirements:

1. When button 1 is pressed i.e. when the input is 1, the Green LED will glow and the

window will move upwards.

When button 2 is pressed i.e. when the input is 1, the Red LED will glow and the window will move downwards.

When button 1 and button 2 are pressed/released i.e. when the input is 1 and 1 / 0 and 0, the LED will not glow and there will be no window movement (upwards/downwards).

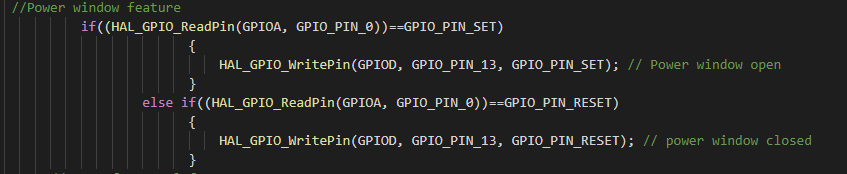
1. When the button 1 is pressed i.e. when the input is 1, the Green LED will glow and the

Sunroof window will open.

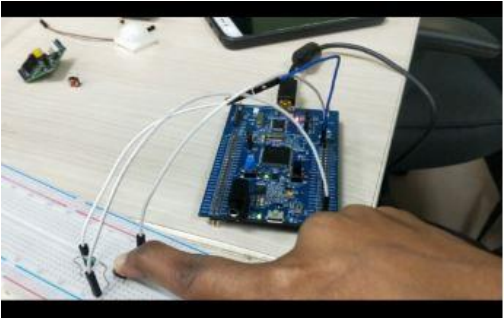
When the button 1 is released i.e. when the input is 0, the Red LED will glow and the Sunroof window will close.

## **2.6 Design:**

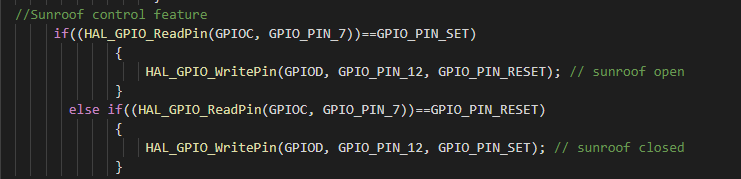
### 2.6.1 Code snippet for Power Window:



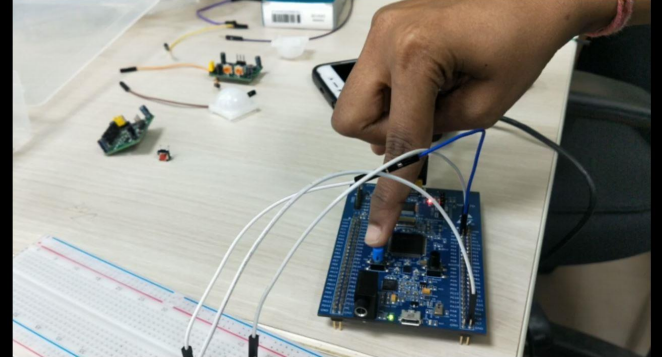
### 2.6.2 Circuit for Power Window:



### 2.6.3 Code snippet for Sunroof Control:



### 2.6.4 Circuit for Sunroof Control:



## **2.7 Test Plan:**

### 2.7.1 High Level Test Plan:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SL No** | **TEST\_ID** | **TESTING FUNCTION** | **EXPECTED INPUT** | **EXPECTED OUTPUT** | **SL No** |
| 1 | HLR\_1 | Power Window | Pressing Button 1 and 2 for Door Window movement control. | Door Window moves according to the switch pressed. | 1 |
| 2 | HLR\_2 | Sunroof Control | Pressing push button for sunroof window movement. | Open/ close of sunroof based on button press. | 2 |

### 2.7.2 Low Level Test Plan:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SL NO** | **TEST\_ID** | **TESTING\_FUNCTION** | **EXPECTED INPUT** | **EXPECTED OUTPUT** | **SL NO** |
| 1 | LLR\_1 | Power Window | When the input is 1 (button 1) i.e. when switch1 is pressed. | The Green LED will glow i.e. the GPIO pin 13 will be set and GPIO pin 14 will be reset. | 1 |
| 2 | LLR\_2 | Power Window | When the input is 1 (button 2) i.e. when switch2 is pressed. | The Red LED will glow i.e. the GPIO pin 14 will be set  and GPIO pin 13 will be reset. | 2 |
| 3 | LLR\_3 | Power Window | When the input is 1,1 (button 1, button 2) i.e. when both switch1, switch2 is pressed. | No LED will glow i.e. the GPIO pin 13 and GPIO 14 will be reset. | 3 |
| 4 | LLR\_4 | Power Window | When the input is 0,0 (button 1, button 2) i.e. when both switch1, switch2 is not pressed. | No LED will glow i.e. the GPIO pin 13 and GPIO 14 will be reset. | 4 |
| 5 | LLR\_5 | Sunroof Control | When the input is 1 i.e. when switch is pressed. | The GPIO pin 12 will be set. | 5 |
| 6 | LLR\_6 | Sunroof Control | When the input is 0 i.e. when switch is released. | The GPIO pin 12 will be reset. | LLR\_6 |

## **2.8 Implementation Summary:**

Multiple features of the car using STM32f407 discovery board has been implemented. Here we have assigned certain pins for GPIO input and output by board configuration in STM32CubeIDE. Then we have generated a high-level abstraction code and then we use certain GPIO HAL functions, ADC HAL functions and configured.

We have also used the same LED’s for different feature indications.

Features that I have implemented and pin configurations are as follows: -

* Power Window:
* PD13: Denotes the window(s) status (SET (1): Open, RESET (0): Closed)
* Sunroof Control:
* PD14: Denotes the window(s) status (SET (1): Close, RESET (0): Open)

## **2.9 Summary:**

In this project, the features that has been selected are namely:

* Power window
* Sunroof Control
* Interior Lighting
* Door Lock/Unlock
* Seat Control, and
* Wiper Control

Out of these six, the features that I have implemented are:

* Power Window, and
* Door lock/Unlock

For power window control, I have used a switch along with the discovery board. When the switch is pressed GPIO PIN PD13 is set and the green LED on the discovery board turns on indicating that the car window(s) is/are open. Similarly, when the switch is released the GPIO PIN PD 13 is reset and the green LED on the discovery board turns off indicating that the window(s) is/are close.

For sunroof control feature, I have used the discovery board along with a buzzer and a switch. When the switch is pressed, the GPIO PIN PD 14 is reset and the buzzer starts buzzing indicating that the door(s) is/are open. Similarly, when the switch is released, the GPIO PIN PD 14 is set and the buzzer stops buzzing indicating the door(s) is/are closed.

## **2.10 Challenges faced and how were they overcome:**

* At first, we were facing issues with the STN32 Discovery Board because of which we needed to change the board.
* There were even few issues with our code even which were eliminated after going through the entire code properly and debugging.
* The makefile that we created was not building the code but that was also eliminated after proper debugging

# Mini Project -3 Python Programming (Individual)

## **3.1 Modules Used**

Modules used in this project is Python.

## 3.2 Topic and Subtopics

### Core-Topics

* Basic Python

### Sub-Topics

* Data Types
* Arithmetic operations
* String operations
* Control structures

### Sub-Topics

* If-else statements
* While loops
* For loops
* Nested Loops
* Functions

### Sub-Topics

* Defining custom functions
* Pass by value
* Pass by reference
* Introduction to Library functions
* Data Structures

### Sub-Topics

* List
* Tuple
* Set
* Dictionary
* Exceptional Handling

### Sub-Topics

* Try, Except
* Finally
* pass keyword
* Date and time Library

### Sub-Topics

* Date comparison
* Execution time analysis
* Excel file library (openpyxl)

### Sub-Topics

* Open multiple excel files
* Reading multiple excel files sheet by sheet.
* Mastersheet creation
* Writing to excel file

All the core-topics and sub-topics are implemented through V-Model.

## **3.4 Objectives:**

To extract the data present in different spreadsheets in one excel file as required by the user.

### 3.4.1 Introduction

It is a Data Set problem that will allow users to retrieve data from data set. However, the input is in the form of the PS number, name, email id. The required output is all the candidate data.

### 3.4.2 4W & 1H

**What:**

\* We are preparing the master excel sheet to search and retrieve data from all the 5 excel sheets that are created.

\* It is used for easy search of a particular cell or data of a person

\* It provides information of every person details like bio, academics, health and personal details.

**When:**

\* Searching for person information

\* To get the contact information

\* To get the required details of that person educational qualification.

**Why:**

\* We are using to retrieve the data of an individual candidate from the excel workbook of 5 sheets where all the relevant data of 40 candidates is present.

\* We can easily access the details of that individual by giving some input such as name, Ps no and email id.

**Where:**

\* To check the information and bio of a person

\* Very useful during emergency times like health issues

\* We can also use it for knowing that person's bank details and other details related to his or her educational qualification.

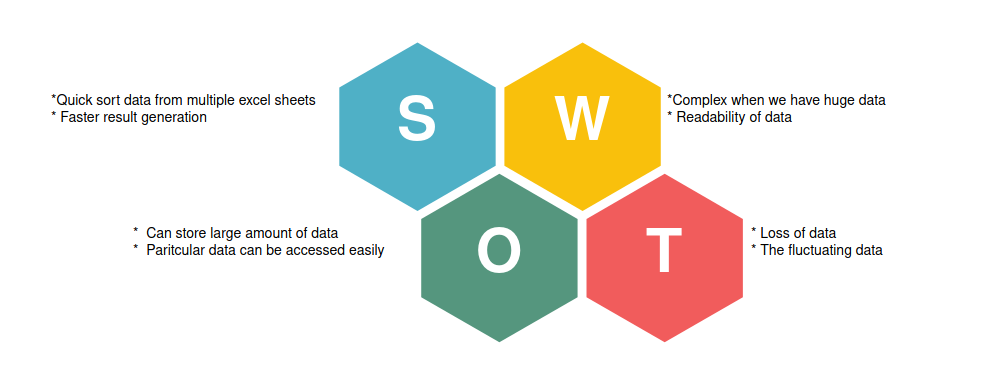
**How:**

\* Input: - We need to give 3 inputs such as Name, Ps No and Email Id.

\* Output: -We will get all the relevant information of that person whose name, Ps no and email id is given.

\* source: -All the relevant data will get copied in master sheet.

### 3.4.3 SWOT Analysis



## **3.5 Requirements:**

### 3.5.1 High Level Requirement Analysis:

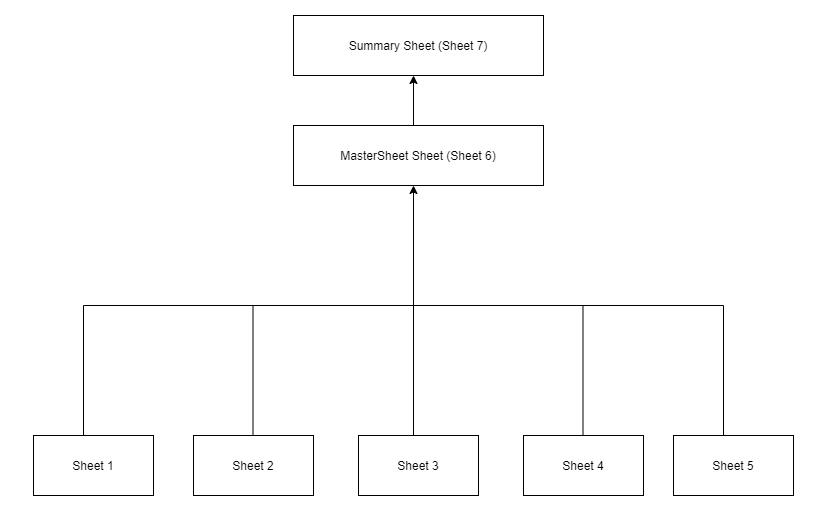
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requirements** | **Description** | **Status** |
| H\_01 | Search data from sheet | Search all data from sheets when user gives the name, PS No. and email id to be searched | IMPLEMENTED |
| H\_02 | write data into new Sheet | Write all the data from different sheets in one Master Sheet | IMPLEMENTED |
| H\_03 | Extract data from sheets using given input | Write new required data in the excel file | IMPLEMENTED |

### 3.5.2 Low Level Requirement Analysis:

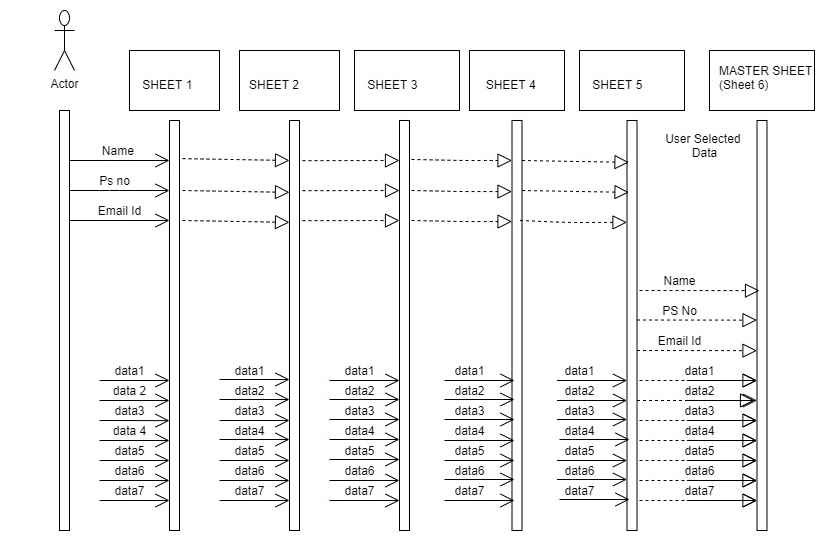
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requirements** | **Description** | **Status** |
| L\_01 | Data Collection | worksheets contain the data of company details and academic details of users input | IMPLEMENTED |
| L\_02 | Each Sheet Contains 10 Column and 40 Rows | Each Sheet showing 10X40 format | IMPLEMENTED |
| L\_03 | Excel file format | the workbook file should be of. xslx format | IMPLEMENTED |
| L\_04 | Inputs | User can give multiple inputs like name, PS No, name and email id at once | IMPLEMENTED |
| L\_05 | Reading Data | Reading all 5 worksheets from workbook Search for specific data based on user specific inputs | IMPLEMENTED |
| L\_06 | Searching Data | Search for specific data based on user specific inputs | IMPLEMENTED |
| L\_07 | Master Sheet Contains Created | Master Sheet Contains 40X40 Format | IMPLEMENTED |

## **3.6 Design**

### 3.6.1 High Level Diagram



### 3.6.2 Low Level Diagram



## **3.7 Test Plan:**

### 3.7.1 High Level Testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Id** | **Description** | **Expected Input** | **Expected Output** | **Actual Output** | **Type of Test** |
| 1 | To Access multiple workbooks stored in different locations | To Access multiple workbooks stored in different locations | Choice:2  Number of worksheets=2, C:\Users\99003731\Desktop\apps\New folder2\share\_axis.xlsx  C:\New folder1\share\_asian.xlsx | ‘<Workbook 1>’  ‘<Workbook 2>’ | ‘<Workbook1>’  ‘<Workbook2>’ |
| 2 | To Read multiple worksheets of a workbook | To Read multiple worksheets of a workbook | Choice:2  Number of worksheet =1,  Enter file path: D:\New folder\share\_adani.xlsx | ‘<ADANIPORTS>,  <Sheet 1>’ | ‘<ADANIPORTS>,  <Sheet 1>’ |
| 3 | To Search by date | Choice:1 (to take excel files path from file)  Choice1 :1 (To search by date)  Number of dates:2  Date :2/2/200  Date: 3/1/2000 | All the Data of 2/2/2000 and 3/1/2000 copied to master-workbook | All the data of 2/2/2000 and 3/1/2000 copied to master-workbook | Requirement based |

### 3.7.2 Low Level Testing:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Id** | **Description** | **Expected Input** | **Expected Output** | **Actual Output** | **Type of Test** |
| 1 | To access multiple worksheets by provide their path one by one | Choice: 2  Number of worksheets = 3  Enter file path: C:\Users\99003731\Desktop\apps\New folder2\share\_axis.xlsx  Enter File path:  C:\New folder1\share\_asian.xlsx  Enter File path:  D:\New folder\share\_adani.xlsx   |  | | --- | |  | | ‘<Workbook 1>’  ‘<Workbook 2>’  ‘<Workbook 3>’ | ‘<Workbook1>’  ‘<Workbook2>’  ‘<Workbook 3>’ | Requirement based |
| 2 | To access multiple worksheets whose paths are present in text file | Choice:1  File.txt | Starts reading every worksheet | Starts reading every worksheet | Scenario based |
| 3 | Searching data by only dates | Choice:1 (to take excel files path from file)  Choice1 :1 To search by date  Number of dates:2   1. 29/2/200   31/1/2000 | All the Data of 29/2/2000 and 31/1/2000 copied to master-workbook | All the data of 29/2/2000 and 31/1/2000 copied to master-workbook | Requirement based |
| 4 | Search Data by only ShareID | Choice:1 (to take excel files path from file)  Choice1 :2 To search by ShareID  Number of shares:2   1. HD855   AP575 | All the Data related to ShareID’s HD855, AP575 is copied to master-workbook | All the Data related to ShareID’s HD855, AP575 is copied to master-workbook | Requirement based |
| 5 | Search data by both date and ShareID | Choice:1 (to take excel files path from file)  Choice1 :3 To search by both ShareID  Number of dates: 2   1. 4/2/2000 2. 17/1/2000   Number of shares:1  HD855 | All the data on 4/2/2000,  17/1/2000 for  ShareID HD855 is copied to master-workbook | All the data on 4/2/2000,  17/1/2000 for  ShareID HD855 is copied to master-workbook | Requirement based |
| 6 | To Skip date which is not present in the sheets and search for all the other data | Choice:1 (to take excel files path from file)  Choice1 :1 To search by date  Number of dates:2   1. 29/2/200   26/1/2000 (Not a valid date) | Only data of 29/2/2000 copied to master-workbook | Only data of 29/2/2000 copied to master-workbook | Scenario based |
| 7 | To Skip ShareID which is not present in the sheets and search for all the other data | Choice:1 (to take excel files path from file)  Choice1 :2 To search by ShareID  Number of shares:2   1. AX700 (Invalid ShareID)   AP575 | Only data of AP575 copied to master-workbook | Only data of AP575 copied to master-workbook | Scenario based |
| 8 | To Skip date/ShareID which is not present in the sheets and search for all the other data | Choice:1 (to take excel files path from file)  Choice1 :3 To search by both ShareID  Number of dates: 2   1. 4/2/2000 2. 17/1/2000   Number of shares:1  AB878 | Empty Master-workbook | Empty Master-workbook | Scenario based |

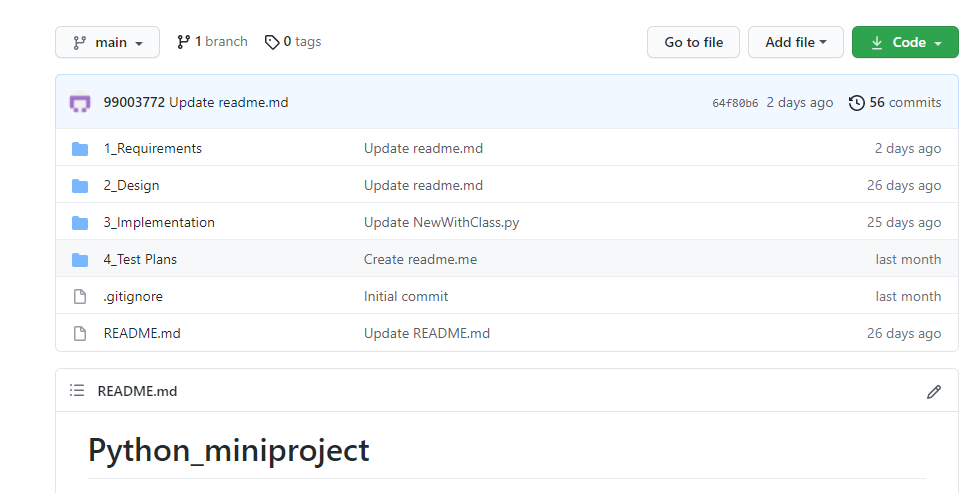
## **3.8 Implementation Summary:**

Aim of this project is to extract the data present in 5 different spreadsheets in one excel file as required by the user. The excel sheet consists of 5 different spreadsheets. The user defines the data that needs to be searched based on Area Name. The python program then reads the data corresponding to the data from different spreadsheets of excel. It then creates a master sheet and adds the data from all the sheets to it In the end, it will create a summary sheet which will count the individual number of date and the total number of data.

## **3.9 Git Link:**

<https://github.com/99003772/Python_miniproject.git>

## **3.10 Git Dashboard:**



## **3.11 Git Summary:**

There are four folders namely:

* 1\_Requirements:

Here there are two folders:

* HLR - It consists of the high-level requirement table.
* LLR - It consists of the low-level requirement table

There is also a readme.md file consisting of:

* + - * Introduction
      * 4W & 1H
      * SWOT Analysis
* 2\_Design

Here there are two folders:

* HLR - It consists of the high-level requirement UML diagram.
* LLR – It consists of the low-level requirement UML diagram.
* 3\_Implementation

Here there are two files:

* project.py file consisting the code.
* Mini\_project.xlsx file consisting of the raw data.
* Test Plan

Here there is a readme.md file consisting of:

* High Level Test Plan
* Low Level Test Plan

## **3.12 Summary:**

Python is one of the very powerful language present in the industry. However, despite the number of cons developer still use python to develop end-to-end, quality and robust web application. It is a great programming language with some excellent advantage such as easy use code lines, smooth maintenance and easy debugging.

Python is also used of automation and this is reflected in the mini project also. An excel file with 5 data sheets is given and based upon user input the data corresponding to the input will be searched from all 5-different sheet and will be written in the master sheet, and in the end, it will create a summary sheet which will count the individual number of date and the total number of data.

## **3.13 Challenges faced and how were they overcome:**

Some of the challenges faced are as follow:

* Working with, and handling excel sheet is a new topic and there were lots of issues initially.
* Understanding the concept of pandas and openpyxl modules and what module to use was challenging in the beginning, as many times excel sheet was getting corrupted.
* Writing the extracted data into the master sheet was also challenging.

## **3.14 Future Scope:**

* Searching in huge amount of data needed to be analyzed.
* Code can be more optimized and automatic generation of summarysheet or masterbook can be added.
* This code as an API or GUI can be used in analysis of data in schools, colleges, MNCs, banks, etc. for maintaining records.

# Mini Project -4 Kernel Programming and Device driver =>[Individual]

## **4.1 Modules Used:**

The modules used in this are Linux and Kernel Device Drivers.

## **4.2 Topic and Subtopics:**

Core- Topic

* System calls

Sub- Topics

* C programming
* Custom Kernel Image Generation
* Cross-Compilation (using Qemu)
* IPC (Inter- Process Communication)

Sub-Topics

* C programming
* Process
* Race Condition
* Semaphores
* Mutex
* Spinlock
* Wait Queue
* Threads
* Custom Modules

Sub- Topics

* Kfifo API
* List API
* Module development
* IOCTL

## **4.3 Objectives:**

* System calls -- echo back the given string
* System calls -- Traverse process list, print pid and ppid
* System calls -- Retrieve attributes of calling process

## **4.4 Requirements:**

|  |  |  |  |
| --- | --- | --- | --- |
| **SL No.** | **Requirements** | **Description** | **Status** |
| 1 | System call to echo string | The system call echo backs the given string. | Implemented |
| 2 | System to traverse process list and print pid | System call traverses through process list and print pid of all the running process. | Implemented |
| 3 | System to traverse process list and get attributes like state, priority and pid of the calling process | System call traverses through process list and retrieves state, priority and pid of calling process. | Implemented |

## **4.5 Test Plan:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Description** | **Expected Input** | **Expected Output** | **Actual Out** | **Type of Test** |
| HLR\_01 | Given string echo’s back | “Hello welcome to kernel” passed as a command line argument | “Hello welcome to kernel” (on serial console (Qemu)) | “Hello welcome to kernel” | Requirement based |
| HLR\_02 | PID and PPID of all the running process is to be printed | Invoking executable file of user-space code on serial console | PID, PPID of all the running process. (on terminal) | PID, PPID of all the running process. | Requirement based |
| HLR\_03 | Attributes such as state, priority, PID, PPID of calling process is to be printed | Invoking executable file of user-space code on serial console | PID, PPID, State, Priority of the calling process. (on terminal) | PID, PPID, State, Priority of the calling process. | Requirement based |

## **4.6 Implementation Summary:**

For System Calls: -

* Generate new zImage after adding system calls definition and prototype. Adding its definition file name in Makefile.

-> make ARCH=arm CROSS\_COMPILE=arm-linux-gnueabi- zImage

* Then mount the SD card and copy the output file of user space code in it and then unmount it by following commands: -

-> sudo mount –o loop, rw, sync rootfs.img /mnt/rootfs

-> sudo cp a.out rootfs.img /mnt/rootfs/home/root

->sudo umount rootfs.img /mnt/rootfs

* Then run Qemu using following command

-> qemu-system-arm -M vexpress-a9 -m 1024 -serial stdio \

-kernel zImage -dtb vexpress-v2p-ca9.dtb \

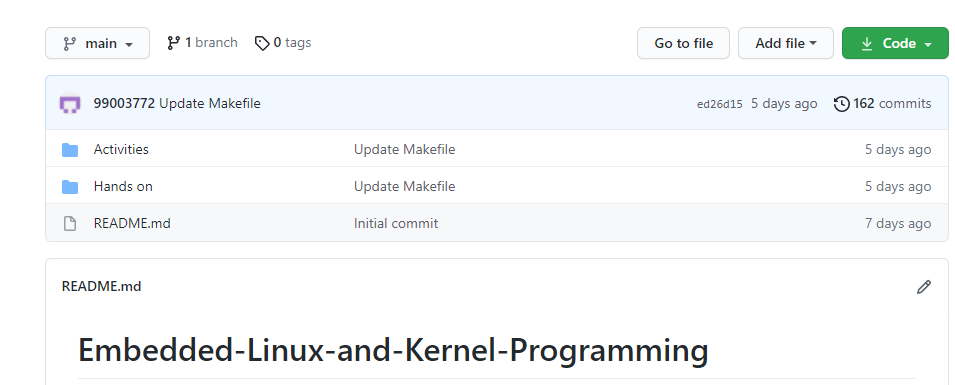
-sd rootfs.img -append "console=ttyAMA0 root=/dev/mmcblk0 rw"

* Then run the ./a.out file on Qemu and output is display either on VGA console or in serial console based on the system call. (give command line input only for first system call)
* System call displays the required output.

## **4.7 Git Link:**

<https://github.com/99003763/Embedded-linux-and-Kernel-Programming.git>

## **4.8 Git Dashboard**:



## **4.9 Summary:**

In this project custom system calls for a particular kernel is made by modifying internal syscalls.h, syscall.tbl , kernel /Makefile and its definition in c file in kernel folder of kernel source.

In user-space code of the system call a special system call number is mentioned to use the custom system call which is defined system call table (syscall.tbl). Finally, it’s test on serial console and VGA console according to expected input and output.

## **4.10 Challenges faced and how were they overcome:**

* Unable to directly access string in kernel space from user space and vice-versa – Using copy\_from\_user () and copy\_to\_user () solved this issue.
* Traversing through system process list was an issue- It was solved by using for\_each\_process () and task\_struct.
* Traversing through node list was issue that was resolved using list\_for\_each () method.