./

GENESIS - Learning Outcome & Mini-project Summary Report



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

**Details**

Contents

[Contents 3](#_Toc69724499)

[1. MINI PROJECT -1: TEAM – SDLC 5](#_Toc69724500)

[1.1 Objectives and requirements: 5](#_Toc69724501)

[ Pros 5](#_Toc69724502)

[LOW LEVEL REQUIREMENTS: 7](#_Toc69724503)

[FEATURES AND COST 8](#_Toc69724504)

[1.2 Design: 9](#_Toc69724505)

[1.3 TEST PLAN 16](#_Toc69724506)

[1.4 IMPLIMENTATION SUMMARY: 18](#_Toc69724507)

[1.5 VIDEO SUMMARY: 19](#_Toc69724508)

[1.6 Git Link: 19](#_Toc69724509)

[ https://github.com/99003756/NTEAM5\_SDLC\_CALCULATOR.git 19](#_Toc69724510)

[1.7.1 Git Dashboard: 19](#_Toc69724511)

[1.7 Summary: 19](#_Toc69724512)

[1.8 Individual Contribution 28](#_Toc69724513)

[1.8.2 Summary 28](#_Toc69724514)

[1.8.3 Challenges faced and how were they overcome 29](#_Toc69724515)

[1.8.4 Future Scope (If applicable) 29](#_Toc69724516)

[Miniproject -2 [Individual] – Python 30](#_Toc69724517)

[2.1 Module 30](#_Toc69724518)

[2.2 Project title: Retrieve data from multiple Excel sheet 30](#_Toc69724519)

[2.3 Topic and Subtopics 30](#_Toc69724520)

[2.4 Objectives: 30](#_Toc69724521)

[Creating one Excel file with five Sheets, one sheet is the master sheet. Here in 4 Excel sheets some data should be same. By searching on the particular data we can get the total data in the mastersheet. 30](#_Toc69724522)

[2.5 Requirements: 30](#_Toc69724523)

[2.5.1 High Level requirement analysis 30](#_Toc69724524)

[2.5.2 Low Level requirement analysis 30](#_Toc69724525)

[2.7 Implementation Summary 33](#_Toc69724526)

[2.9 Git Link 33](#_Toc69724527)

[2.10 Summary 33](#_Toc69724528)

[2.10.1 Outcomes: 33](#_Toc69724529)

[2.11 Challenges faced and how were they overcome: 33](#_Toc69724530)

[3. Miniproject -3 Embedded C => [Team] 34](#_Toc69724531)

[3.1 Modules Used: 34](#_Toc69724532)

[Modules used in this project are Embedded Systems and Embedded C Programming and was implemented on the hardware STM32. 34](#_Toc69724533)

[3.2 Topic and Subtopics 34](#_Toc69724534)

[3.3 Objectives & Requirements 34](#_Toc69724535)

[Design: 37](#_Toc69724536)

[Implementation Summary 43](#_Toc69724537)

[Git Link 43](#_Toc69724538)

[Git Dashboard 43](#_Toc69724539)

[Summary 43](#_Toc69724540)

[Challenges faced and how were they overcome: 44](#_Toc69724541)

[Individual Contribution & Highlights 45](#_Toc69724542)

[Summary 45](#_Toc69724543)

[Miniproject -4 [Individual] – Kernel Programming and Device Drivers 46](#_Toc69724544)

[4.1 Module/s: 46](#_Toc69724545)

[4.2 Topic and Subtopics: 46](#_Toc69724546)

[4.3 Objectives & Requirements: 47](#_Toc69724547)

[4.3.1 Requirements: 47](#_Toc69724548)

[4.4 Implementation Summary: 47](#_Toc69724549)

[4.4.1 Hands-on Activity that are implemented are as follow: 47](#_Toc69724550)

[4.4.2 User space code: 48](#_Toc69724551)

[4.4.3 kthread examples: 48](#_Toc69724552)

[4.5 Git Link: 48](#_Toc69724553)

[4.6 Summary: 48](#_Toc69724554)

[4.7 Challenges faced and how were they overcome: 49](#_Toc69724555)

## **MINI PROJECT -1: TEAM – SDLC**

## **1.1 Objectives and requirements:**

**Pros and Cons**

## Pros

* + - 1. More operations possible.
      2. Efficient
      3. User friendly
      4. LCD display
    - Cons
      1. High cost
      2. Need to have some knowledge for operating calculators.

**SWOT ANALYSIS:**

**STRENGHTS:**

* It is very user friendly since it has limited functions.
* Dual power (battery and solar)
* High speed calculations
* High resolution LCD

**WEAKNESS:**

* High cost
* Not a water proof
* High maintenance cost

**OPPURTUNITIES:**

* The product will do very well in shop counters as the calculator is very user friendly and anybody can use it without prior knowledge of that calculator.
* Students up to class 10 will be attracted to this product because of its simplicity in operations and design.
* The product will also work in banking sectors and other government sectors where they want low price, minimum features, handy products

**THREATS:**

* Can be misused by students by over using from lower classes

**4W’S AND 1H ANALYSIS:**

WHAT: It is a simple electronic hardware device or software that are capable of performing the simple calculations such as addition, subtraction, multiplication, division, calculating power of number, exponential function, logarithmic function, permutation and combination, trigonometry, inverse-trigonometric functions, factorial of a number, binary to decimal conversion etc.

WHEN:

* Useful during exams, for getting complex calculation in very less time
* Calculation of bills in malls, shops, and Restaurants

WHERE:

* Exam hall
* Shop Counters
* Colleges and schools, Banking sectors.

HOW:

* Write the code for all the requirements.
* Write the code for all the requirements
* Saves human power
* We save our valuable time by using the

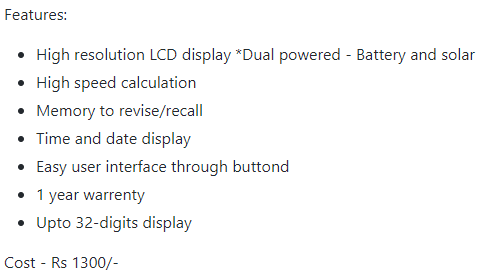
**HIGH LEVEL REQUIREMENT:**

|  |  |  |
| --- | --- | --- |
| **ID** | **DESCRIPTION** | **STATUS** |
| HLR\_01 | ARITHMETIC | IMPLIMENTED |
| HLR\_02 | REALTION | IMPLIMENTED |
| HLR\_03 | AREA CALCULATION | IMPLIMENTED |
| HLR\_04 | VOLUME CALCULATION | IMPLIMENTED |
| HLR\_05 | BODMAS CALCULATION | FUTURE |
| HLR\_06 | TRIGNOMETRIC CALCULATION | IMPLIMENTED |
| HLR\_07 | LOGICAL CALCULATION | IMPLIMENTED |
| HLR\_08 | EXPONENTIAL | IMPLIMENTED |
| HLR\_10 | FACTORIAL | IMPLIMENTED |
| HLR\_11 | DIFFERENTIATION | FUTURE |
| HLR\_12 | INTEGRATION | FUTURE |

## **LOW LEVEL REQUIREMENTS**:

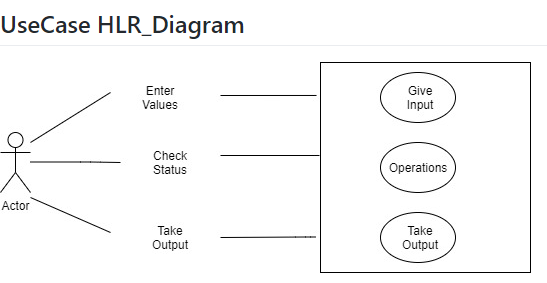
|  |  |  |
| --- | --- | --- |
| ID | DESCRIPTION | STATUS |
| LLR\_01 | IF CHOISE X=3; CHOISE Y =2  CLACULATES “X^Y” | IMPLIMENTED |
| LLR\_02 | IF INPUT =H  CALCULATES H FACTORIAL | IMPLIMENTED |
| LLR\_03 | IF INPUT ENTER VARIABLE  CALCULATES H DIFFERENTIATION | IMPLIMENTED |
| LLR\_04 | IF INPUT = ENTER VARIABLE  CALCULATES INTEGRATION | IMPLIMENTED |
| LLR\_05 | CALCULATES ADDITION, SUBRACTION, MULTIPLICATION, DIVISION, SQUARE ROOT | IMPLIMENTED |
| LLR\_06 | CALCULATES LESS THAN AND GREATER THAN OPERATIONS | IMPLIMENTED |
| LLR\_07 | CALCULATES AREAS FOR CIRCLE, RECTANGLE, SQUARE, TRIANGLE BY TAKING INPUTS LIKE RADIUS, LENGTH AND BREADTH | IMPLIMENTED |
| LLR\_08 | CALCULATES VOLUMES FOR CUBE, SYLINDER, CONE, SPHERE NY TAKING INPUTS LIKE HEIGHT, RADIUS, LENGTH | IMPLIMENTED |
| LLR\_09 | INPUT IS ENTIRE CALCULATED BASED ON BODMAS RULE | FUTURE |
| LLR\_10 | CALUCLATES THE SIN, COS, TAN, COSEC, COT, SEC VALUES BY TAKING INPUT AS ANGLES | IMPLIMENTED |

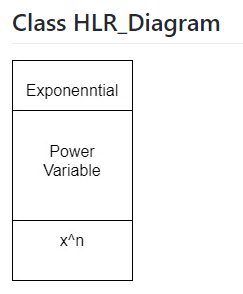
## **FEATURES AND COST**

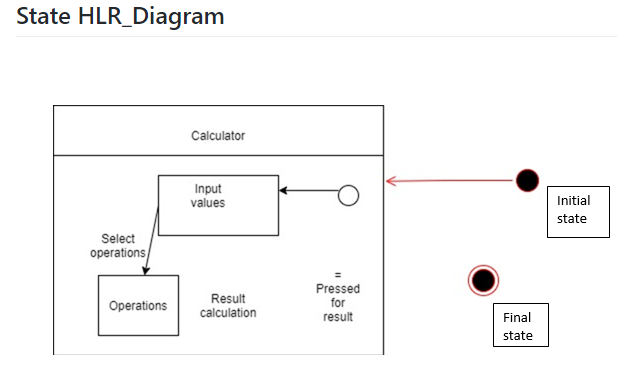


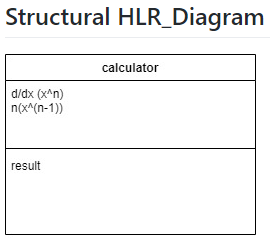
## **1.2 Design:**

**high Level Design:**

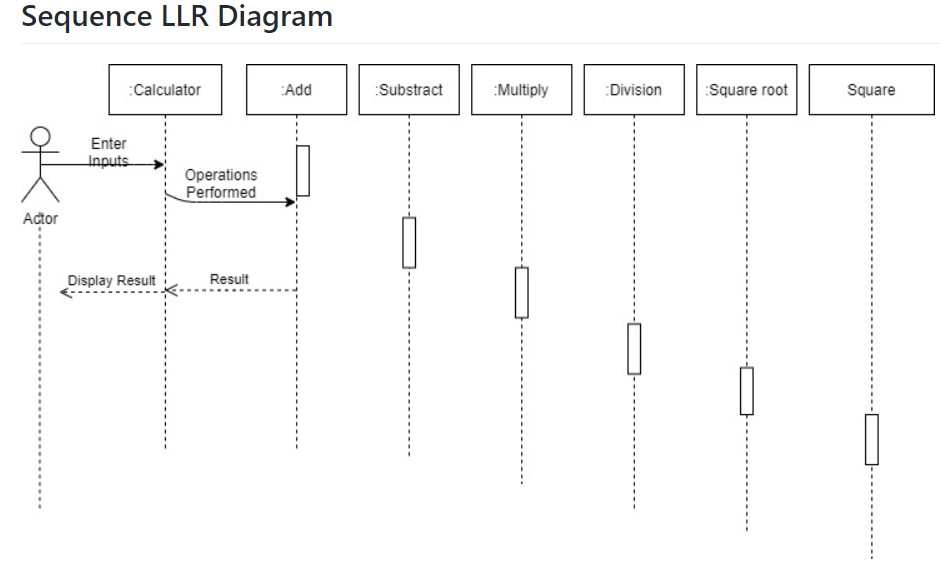
****

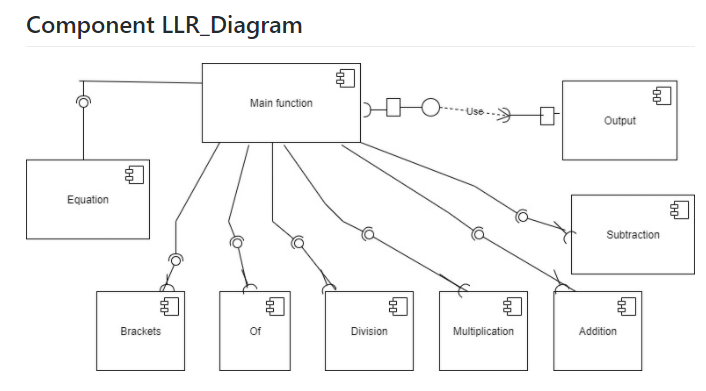
****

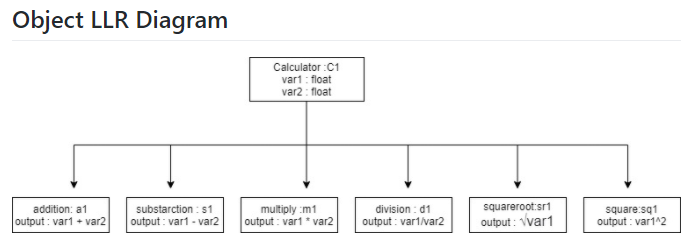
****

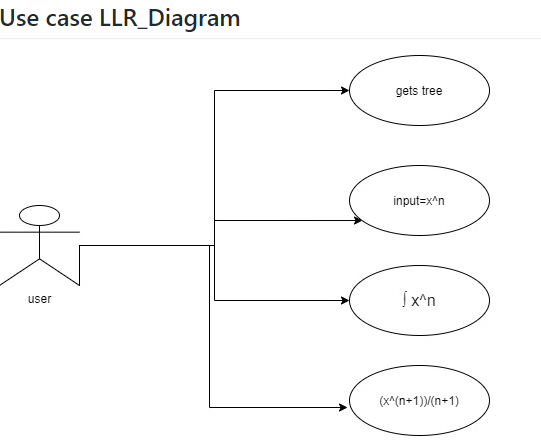
****

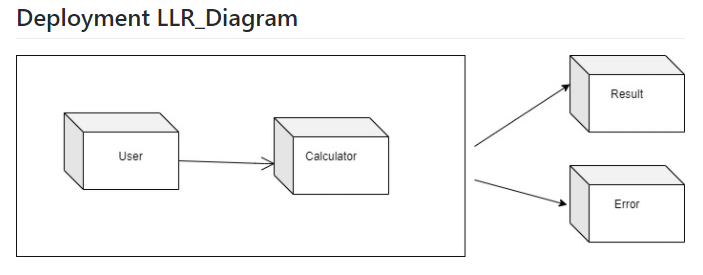
**Low Level Diagrams:**

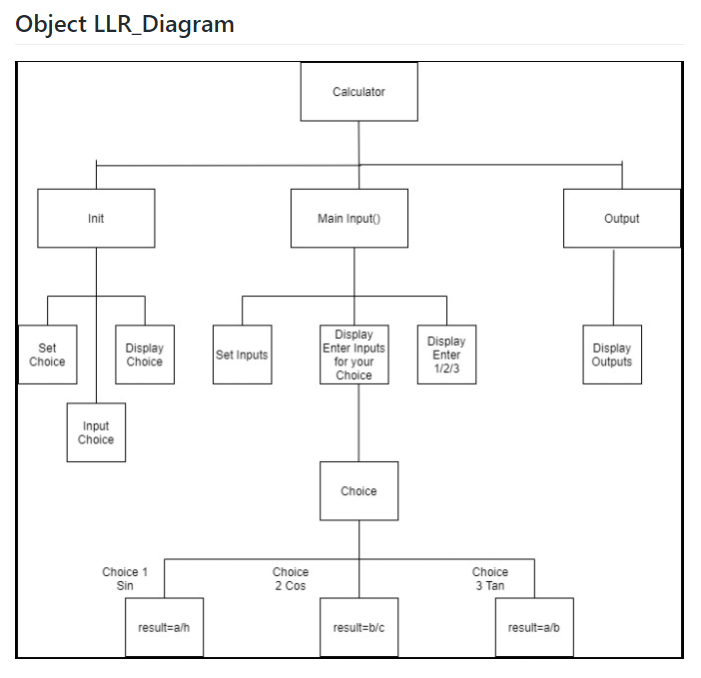
****



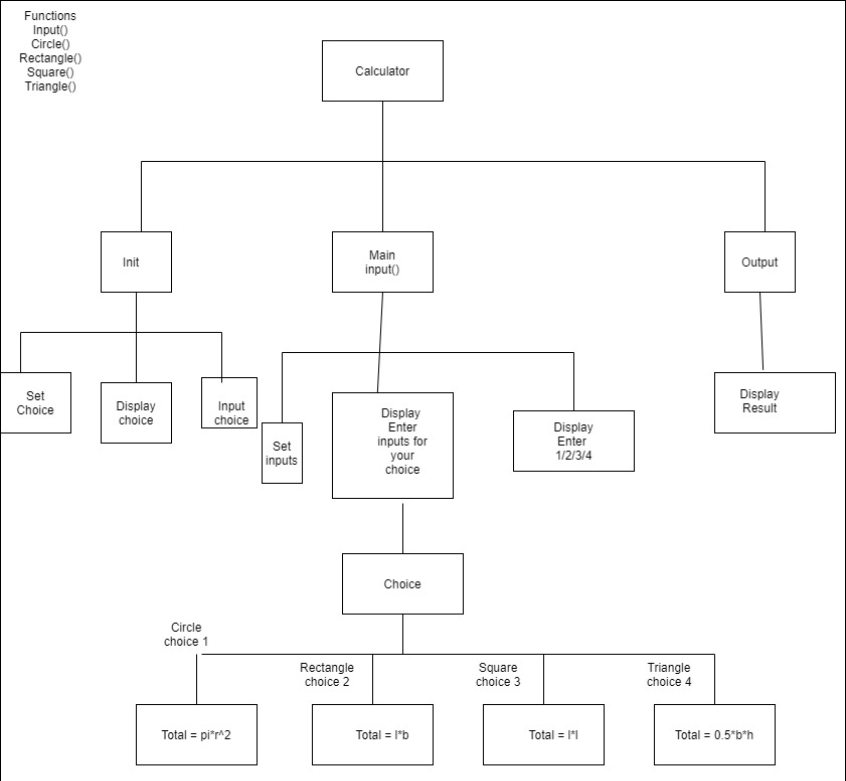








**ACTIVITY LLR\_DIAGRAM**



## **1.3 TEST PLAN**

HIGH LEVEL TEST PLAN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TEST ID | DESCRIPTION | EXP IN | EXP OUT | ACTUAL OUT | TYPE OF TEST |
| H\_01 | Performs Arithmetic Operations | N1 = 6;  N2 = 4 | Performs  Arithmetic Operations | Calculates addition, subtraction, multiplication, division, square, square root | Scenario based |
| H\_02 | Performs Relational Operations | N1 =4; n2 = 5 | Performs Relational Operations | Calculates greater than, smaller than, equals to, not equals to | Scenario based |
| H\_03 | Performs Area Operations | 2 | Performs Area Operations | Calculates area of Triangle ,Rectangle, circle, square | Scenario based |
| H\_04 | Performs volume calculations | 3 | Performs volume calculations | Calculates area of cylinder, cube, cone and sphere | Scenario based |
| H\_05 | Performs trigonometric calculations | 4 | Performs trigonometric calculations | Calculates trigonometric calculations based on inputs | Scenario based |
| H\_06 | Performs exponential | 2,3 | Performs exponential | 8 | Scenario based |
| H\_07 | Performs factorial | 3 | Performs factorial | 6 | Scenario based |
| H\_08 | Performs differentiation | X | Performs differentiation | 1 | Scenario based |
| H\_09 | Performs integration | 1 | Performs integration | x | Scenario based |

**Low level test plan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TEST ID | DESCRIPTION | EXP IN | EXP OUT | ACTUAL OUT | TYPE OF TEST |
| L\_01 | Adding two numbers | 5+2 | 7 | 7 | Requirement based |
| L\_02 | Subtract two numbers | 5-3 | 2 | 2 | Requirement based |
| L\_03 | Divide two numbers | 10/2 | 5 | 5 | Requirement based |
| L\_04 | Multiplying two numbers | 2\*3 | 6 | 6 | Requirement based |
| L\_05 | Square of two numbers | 5\*5 | 25 | 25 | Requirement based |
| L\_06 | Greater than or less than | 3>5 | False | False | Requirement based |
| L\_07 | Equals to or not | 1=5 | false | false | Requirement based |
| L\_08 | Not equals to or not | 2!=3 | False | False | Requirement based |
| L\_09 | Square root | Square root (4) | 2 | 2 | Requirement based |
| L\_10 | Calculates area of a square | S1=2; s2=4 | 8 | 8 | Requirement based |
| L\_11 | Calculates volume of the square | A1=4; a2=6 | 24 | 24 | Requirement based |
| L\_12 | Calculates area of a triangle | H=3, b= 4 | 6 | 6 | Requirement based |
| L\_13 | Calculates volume of the square | H=3, b= 4, l=3 | 18 | 18 | Requirement based |
| L\_14 | Calculates area of a circle | R=4 | 50.265 | 50.265 | Requirement based |
| L\_15 | Calculates volume of the cone | B=4, h= 9 | 12 | 12 | Requirement based |
| L\_16 | Calculates area of a rectangle | L=6, w=4 | 24 | 24 | Requirement based |
| L\_17 | Calculates volume of the cube | A=2 | 8 | 8 | Requirement based |
| L\_18 | Calculates exponential | 2^3 | 8 | 8 | Requirement based |
| L\_19 | Calculates factorial | H=3 | 6 | 6 | Requirement based |
| L\_20 | Calculates trigonometric functions  (For sin) | Angle = 0 | 0 | 0 | Requirement based |
| L\_21 | Calculates trigonometric functions  (For cos) | Angle = 0 | 1 | 1 | Requirement based |

## **1.4 IMPLIMENTATION SUMMARY:**

* 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.
* The **test** folder holds the ***test\_calculator\_operations.c*** file for cumulative testing of the source codes based on requirements, scenario and boundary.
* 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.

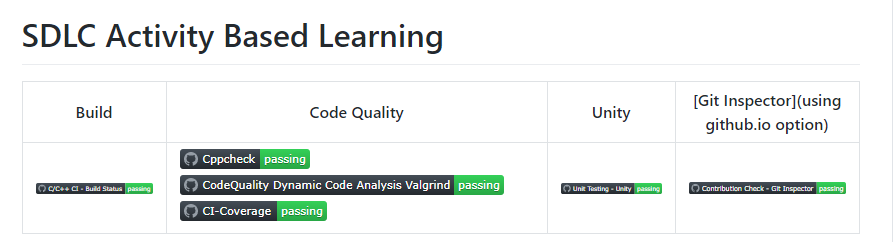
## **1.5 VIDEO SUMMARY:**

“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”

### 1.6 Git Link:

## https://github.com/99003756/NTEAM5\_SDLC\_CALCULATOR.git

### 1.7.1 Git Dashboard:



### 1.7 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.

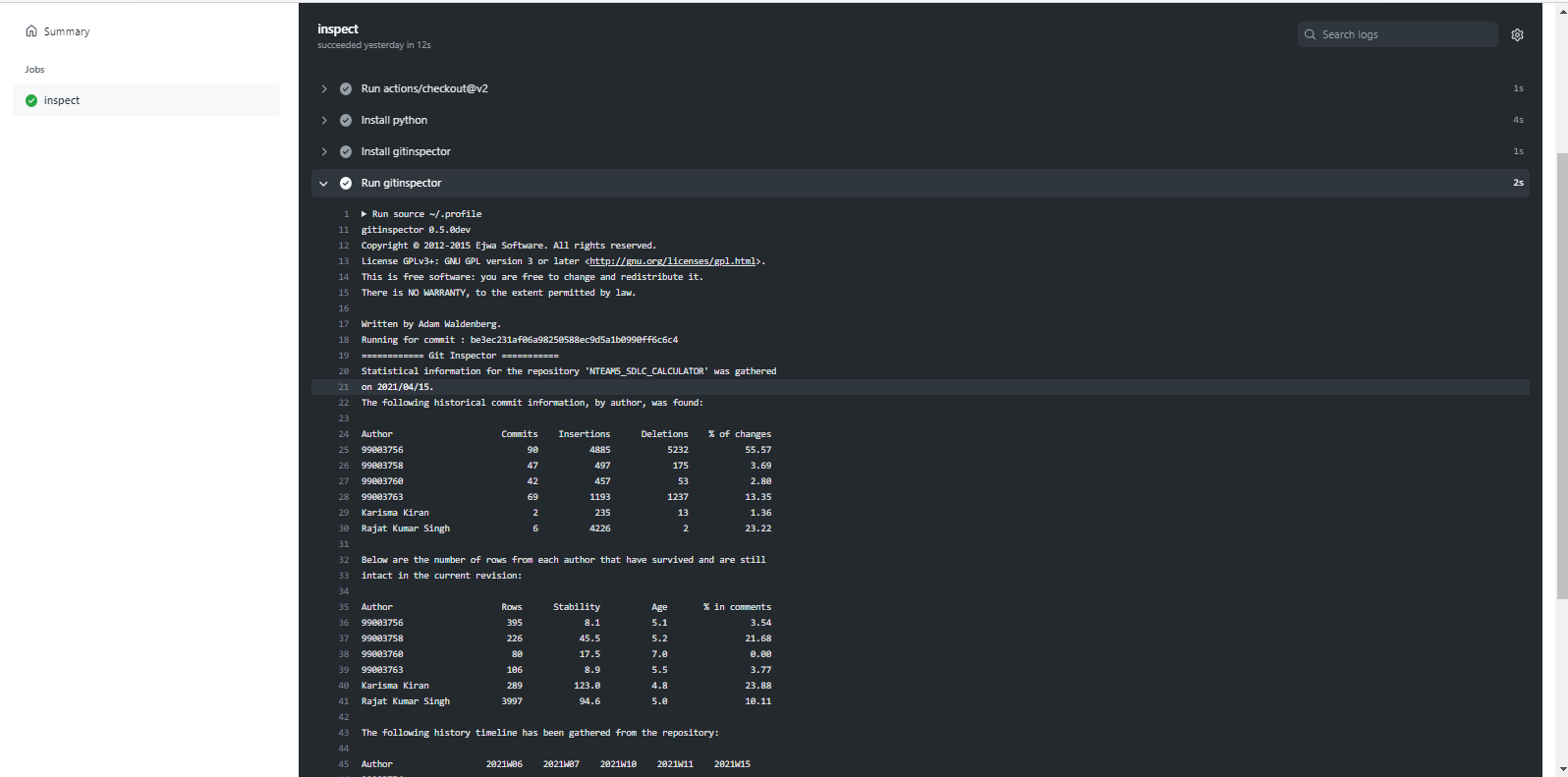
So, for this we first analyzed all the other calculators from low-end feature low cost calculator to high-end feature high-cost calculator and prepared a list of features to include in our modified cost-effective more featured calculator.

Features included in the calculator are basic arithmetic operations (addition, subtraction, multiplication, division), Combinatorics (Permutations and Combinations), Mensuration (Area and Volumes), trigonometric Calculations.

This calculator is implemented through C programming. This calculator will be mainly used by the school, college students, scientists, businessman, engineers for various purpose. Also, this calculator will be cost effective and if implemented over hardware it cost around 1300 INR.

#### **1.7.1 Git inspector summary**

Git Inspector



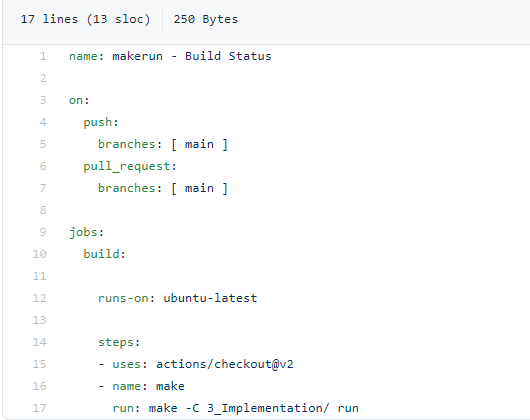
**Setup for Build**



#### 1.7.2 Build

#### Build status:

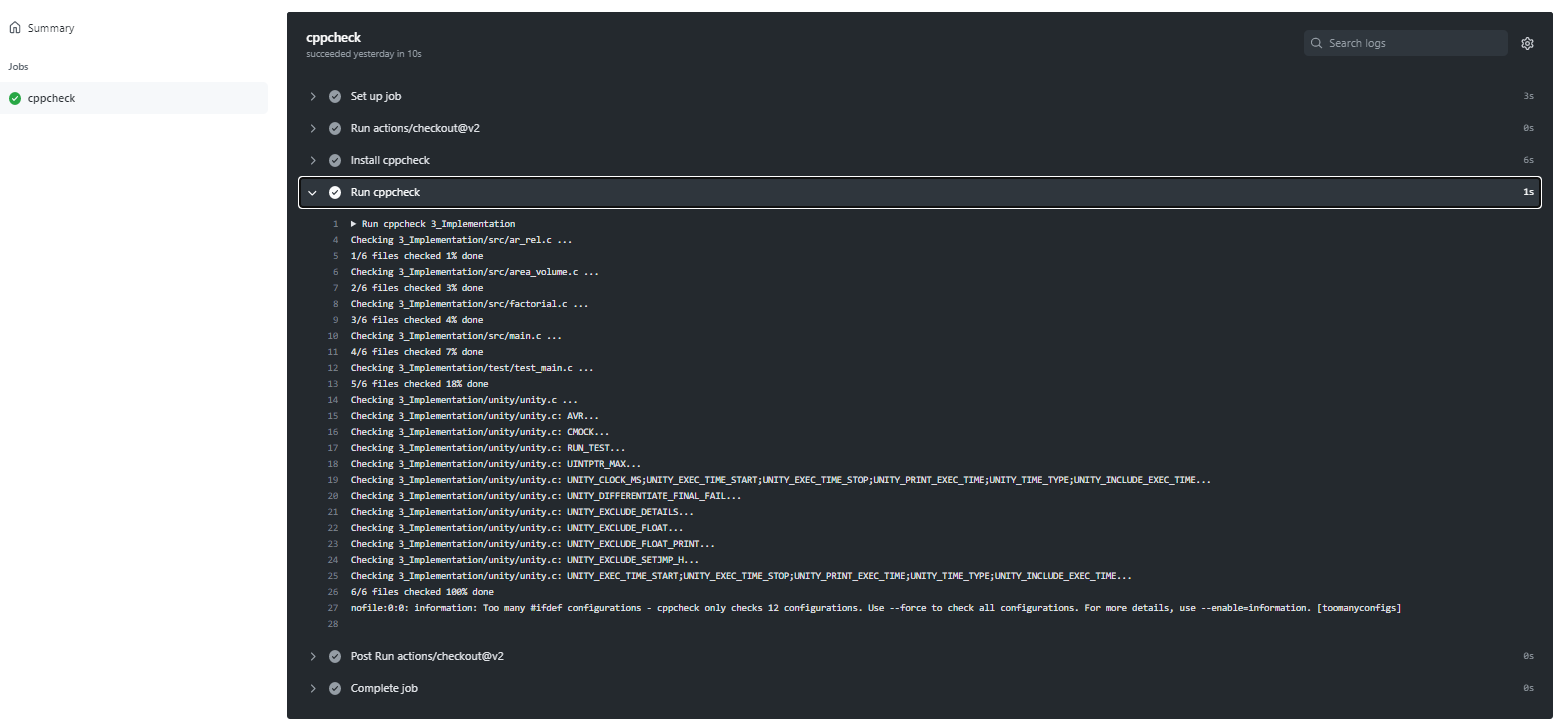
#### 

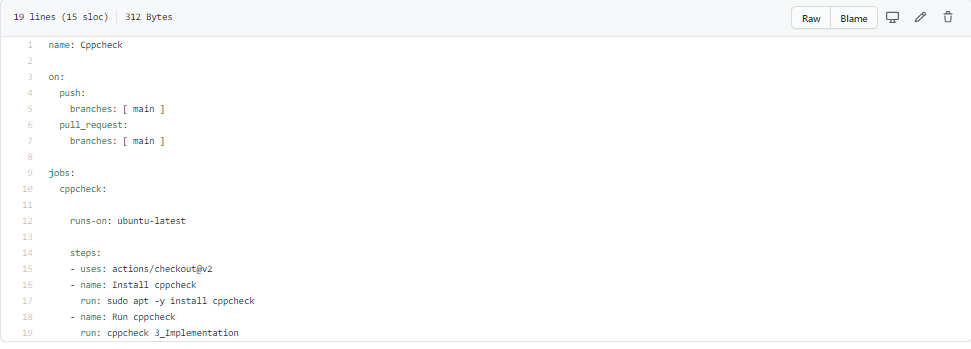


#### 1.7.3 Code quality and Issues or Bug Tracking

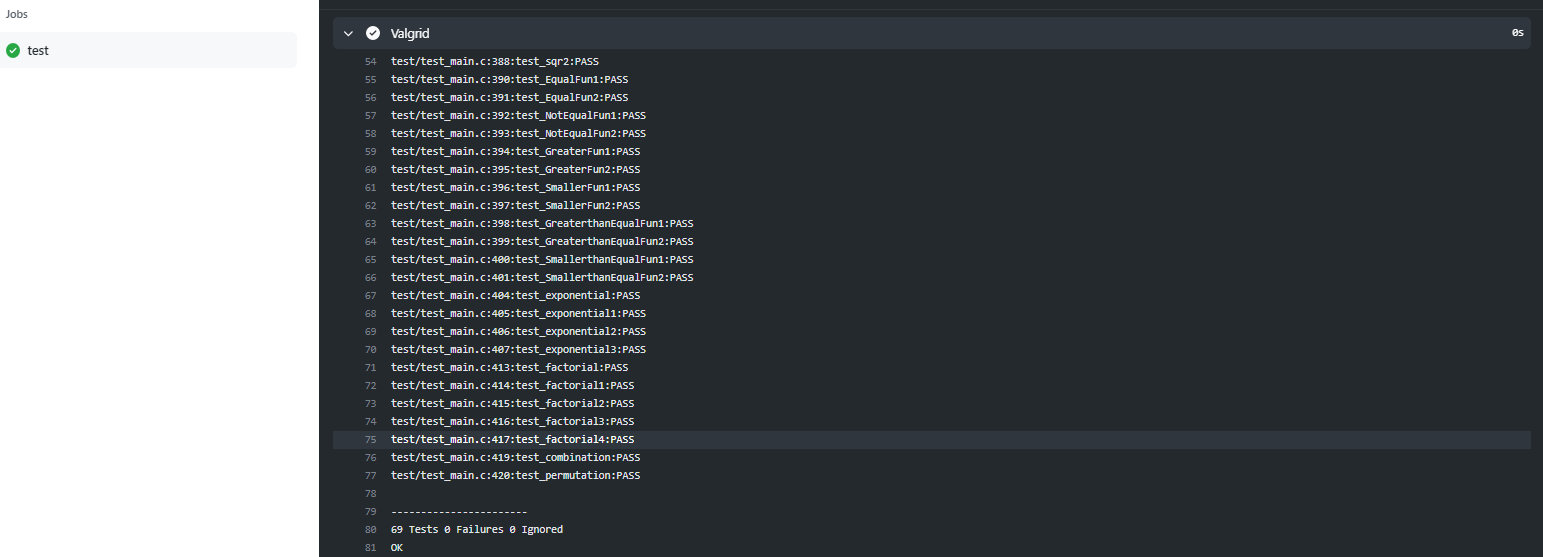
**CODE QUALITY:**

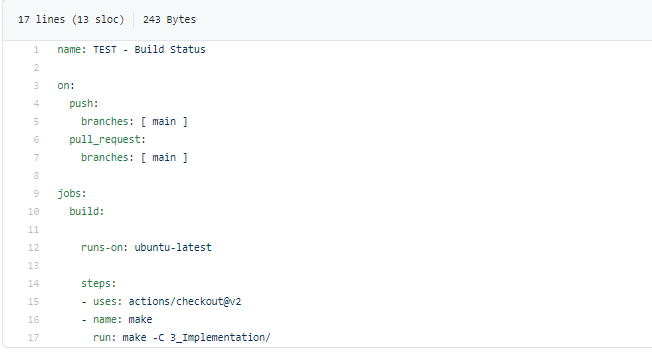
**STATIC CODE QUALITY:**



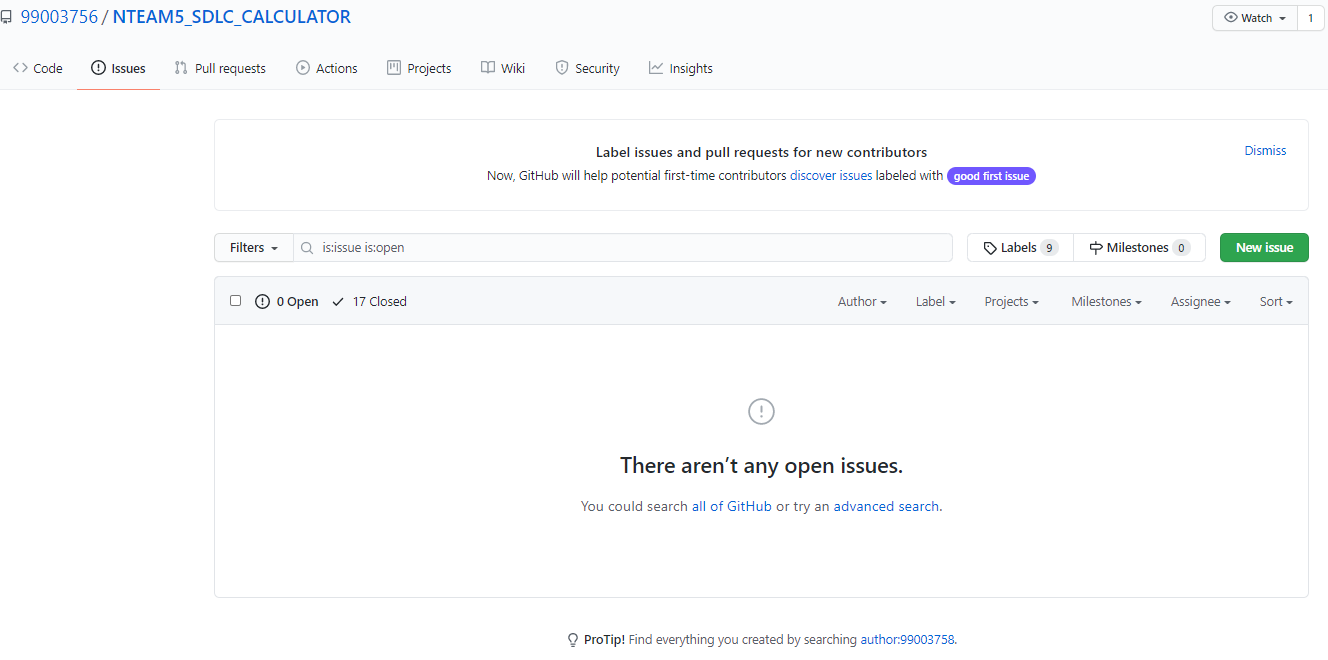


**DYNAMIC CODE QUALITY:**

****

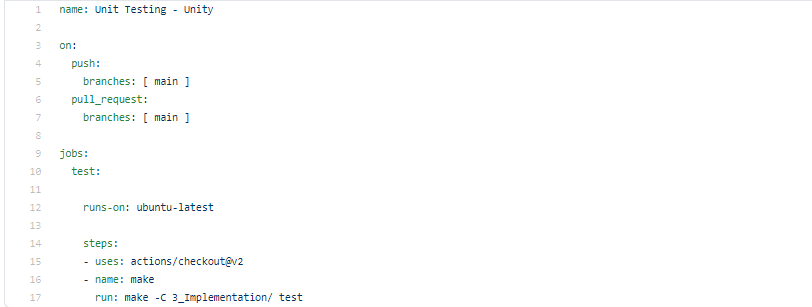
****

**1.7.4 GIT ISSUES:**

****

#### 1.7.5 Unit Testing





## **1.8 Individual Contribution**

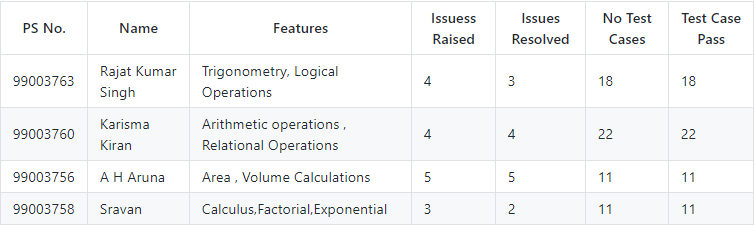
My Contribution to this project is that I have implemented the features of finding FACTORIAL and EXPONENTIAL for given inputs. I have implemented the feature of finding of INTEGRATIONS, DIFFERENTIATION for given inputs.

Also, I have performed the research analysis of different calculators present in market based on features and cost. I have done the SWOT analysis and 4W & 1H analysis in order to understand the pros and cons of our product.

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\_main.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.

**1.8.1 Contribution list and summary:**

****

### 1.8.2 Summary

The main motto is to design a calculator with certain features according to the specific requirements. The target customers for the designed calculator are students, shopkeepers, banking executives and engineers.

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.

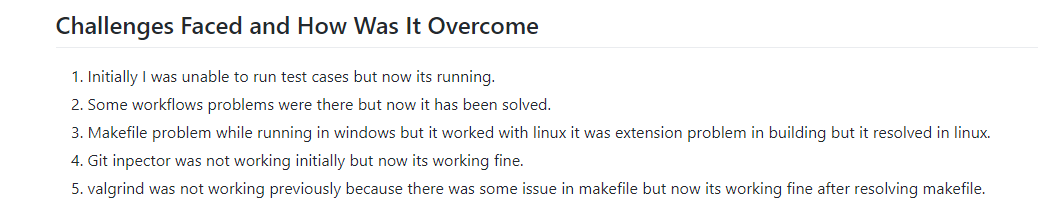
Technical skills developed: -

* Advanced C programming
* Make File
* Unit Testing through Unity
* Version Control using GitHub

Soft-Skills Developed: -

* Team-work
* Team-management
* Time Management
* Assertiveness

### 1.8.3 Challenges faced and how were they overcome



### 1.8.4 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.

# Miniproject -2 [Individual] – Python

## **2.1 Module**

Modules used in this project are Core and Advanced Python.

## **2.2 Project title:** Retrieve data from multiple Excel sheet

## **2.3 Topic and Subtopics**

* Openpyxl
  + Read excel file
  + Write excel file
* Barchart
* functions

## **2.4 Objectives:**

## Creating one Excel file with five Sheets, one sheet is the master sheet. Here in 4 Excel sheets some data should be same. By searching on the particular data we can get the total data in the mastersheet.

## **2.5 Requirements**:

### 2.5.1 High Level requirement analysis

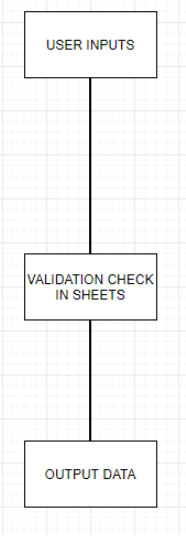
|  |  |  |  |
| --- | --- | --- | --- |
| S. No | Requirements | Description | Status |
| HL1 | Creating | In 5 sheets data of a person is present, one is the master sheet | Implemented |
| HL2 | Combining | All the datasheets are combined in one data sheet | Implemented |
| HL3 | Searching | By using the Ps number of that person, we can get all the information of that person in master sheet | Implemented |

### 2.5.2 Low Level requirement analysis

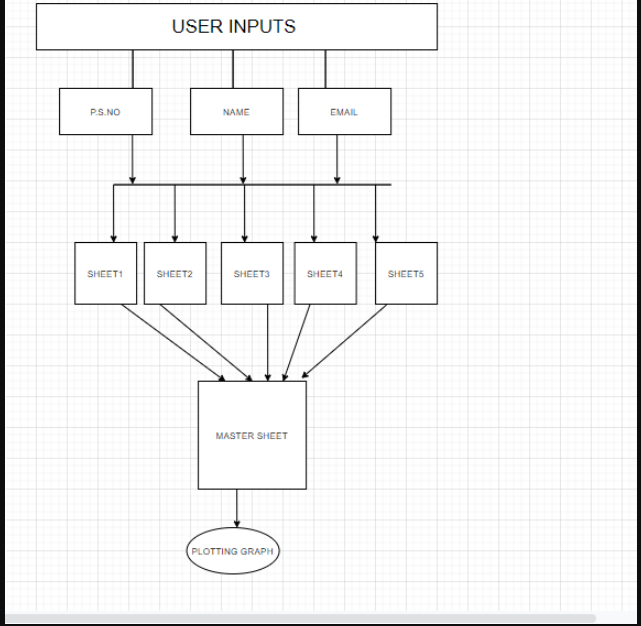
|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Requirements** | **Description** | **Status** |
| LL1 | Creating rows and columns | In each sheet, we have to create 10 columns and 40 rows | Implemented |
| LL2 | Searching | Searching the data, we need in all sheets | Implemented |
| LL3 | Output | As per the requirement we have to display the output and store it | Implemented |

**2.6 Design**

HIGH LEVEL DIAGRAM



LOW LEVEL DIAGRAM



## **2.7 Implementation Summary**

The aim of the project is to extract the data present in different spreadsheets in one excel file as required by the user. The excel sheet consists of 5 spreadsheets. The user defines the data that needs to be searched on the basis of Name, PS Number and Email ID. The python program then reads the data corresponding to the particular 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 plot the bar graph from the data present in the master sheet.

## **2.9 Git Link**

<https://github.com/99003758/mini-project1.git>

## **2.10** **Summary**

### 2.10.1 Outcomes:

Technical:

* Improved implementation of Python concepts.
* Practical implementation of SDLC lifecycle.
* Source code management. (GitHub)

Soft skills:

* Project management
* Conflict management.

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

* System issues(crashing and Interfacing).
* Differentiation of high level and low level.
* Committing to GitHub, pull and push in GitHub.
* Converting pictures & tables into readme

# 3. Miniproject -3 Embedded C => [Team]

## **3.1 Modules Used:**

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

### 3.2 Topic and Subtopics

1)Driver API Development

a) GPIO

b) ADC

c) SPI, UART, I2C

d) External interrupt

e) Debugging using STM Board

2) Driver Development (Hardware Abstraction Level- HAL)

* 1. GPIO
  2. ADC
  3. External Interrupt
  4. Debugging using STM Board

## **3.3 Objectives & Requirements**

BCM module was implemented using STM32f407VG microcontroller featuring 32 bit ARM-cortex M4 with FPU core.

This BCM module have following features:

1. Wiper Control module
2. Interior light control using PIR sensor
3. Adaptive light Control System using LDR
4. Car Reverse Gear Buzzer

**Components Used:**

1. STM32f407VG Microcontroller
2. Breadboard
3. LED
4. LDR Sensor
5. Humidity Sensor (FC-28)
6. PIR Motion detection sensor
7. Ring Buzzer Sensor
8. Jumper Wires

**HIGH LEVEL REQUIREMENTS:**

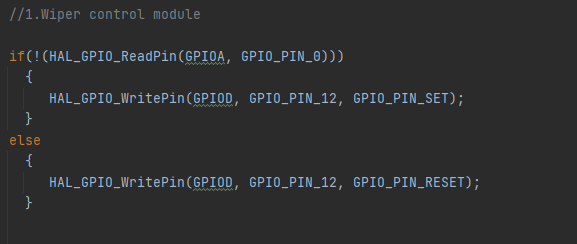
|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requirements** | **Description** | **Status** |
| **1** | Wiper Control module | When moisture sensor senses the water, it sends signal to wiper shield to turn on. | Implemented |
| **2** | Interior light control using PIR sensor | Adjustment of interior lighting using PIR Motion Detection Sensor (when door is opened i.e. when motion is detected, the interior lighting will work.) | Implemented |
| **3** | Adaptive light Control System using LDR | LDR sensor sense the presence and absence of light is necessary and turns on and off the lights | Implemented |
| **4** | Car Reverse Gear Buzzer | Car Reverse Radar Sensor detect the objects during reversing the vehicle and sends signal to buzzer | Implemented |

**LOW LEVEL REQUIREMENTS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Requirements** | **Description** | **Status** |
| 1 | When moisture sensor detects the water content wiper shields starts working | GPIO pin PD12 is set and the LED on the discovery board glows | Implemented |
| 2 | When no water is detected from moisture sensor wiper shields will not work | GPIO pin PD12 is reset and the LED on the discovery board turned off | Implemented |
| 3 | When the door is opened motion is detected, interior lighting will work properly | GPIO pin PD12 is set and the LED on the discovery board glows | Implemented |
| 4 | When the door is closed motion is not detected interior lighting will not work | GPIO pin PD12 is reset and the LED on the discovery board turned off | Implemented |
| 5 | When the darkness is detected around the exterior lights will work properly. | GPIO pin PD12 is set and the LED on the discovery board glows | Implemented |
| 6 | When the high intensity lights are detected around exterior lights will not work | GPIO pin PD12 is reset and the LED on the discovery board turned off | Implemented |
| 7 | When the person in a vehicle uses reverse gear the buzzer starts beeping | GPIO pin PD12 is set and the LED on the discovery board glows | Implemented |
| 8 | When the reverse gear is not in use the buzzer will not work | GPIO pin PD12 is reset and the LED on the discovery board turned off | Implemented |

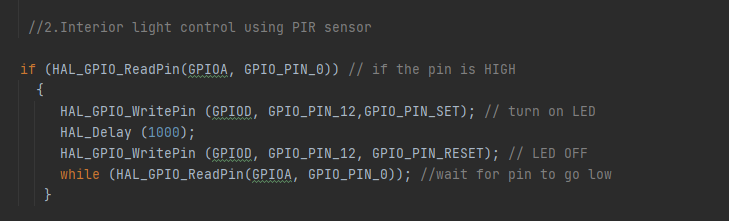
## **Design:**

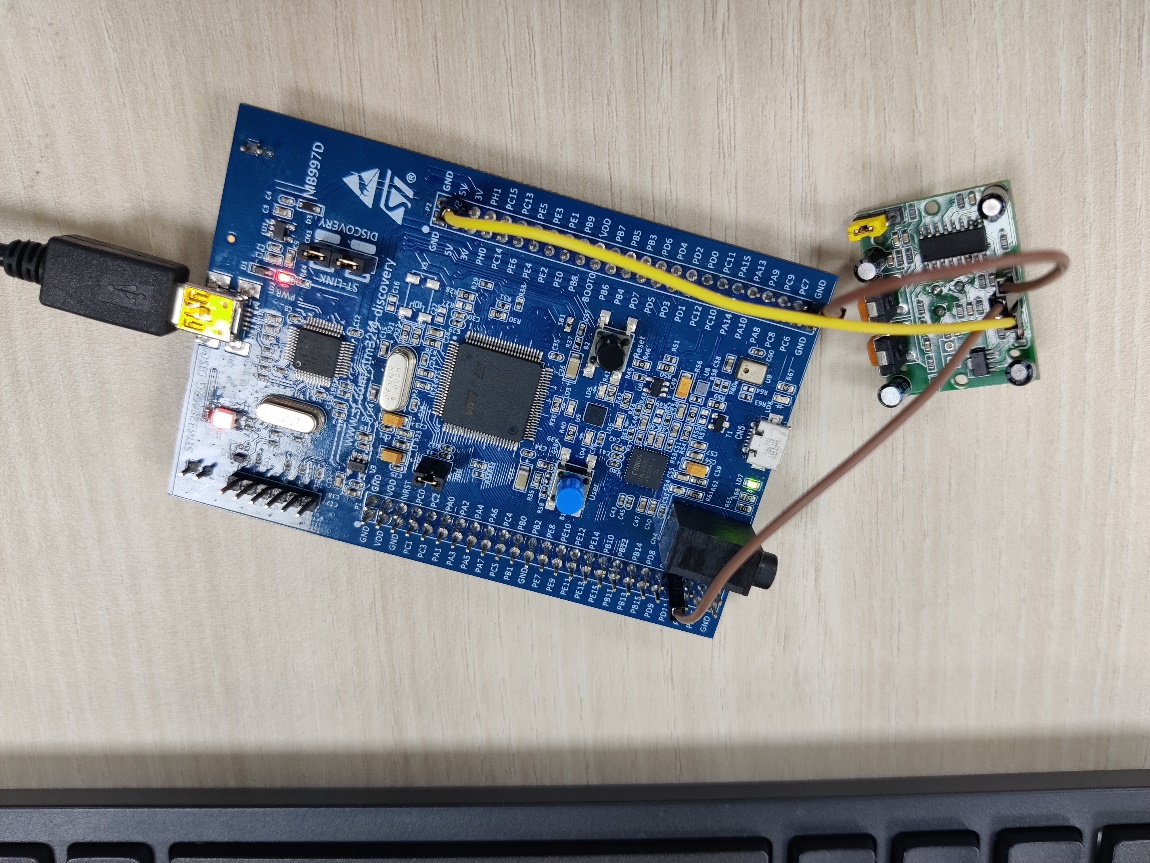
**Wiper Control Module:**

****



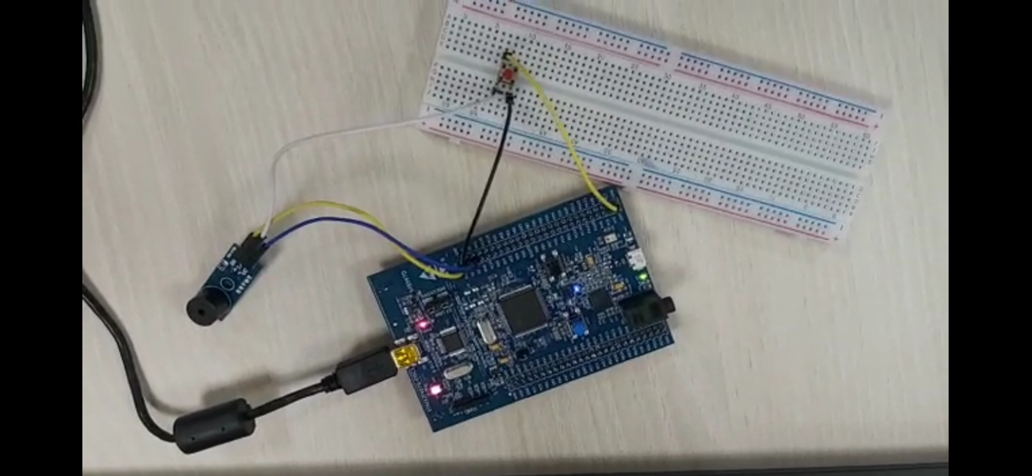
**Interior light control using PIR sensor:**

****

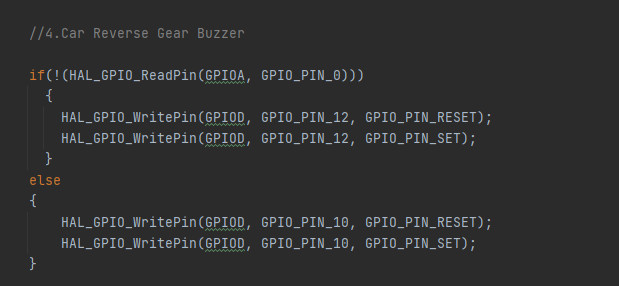


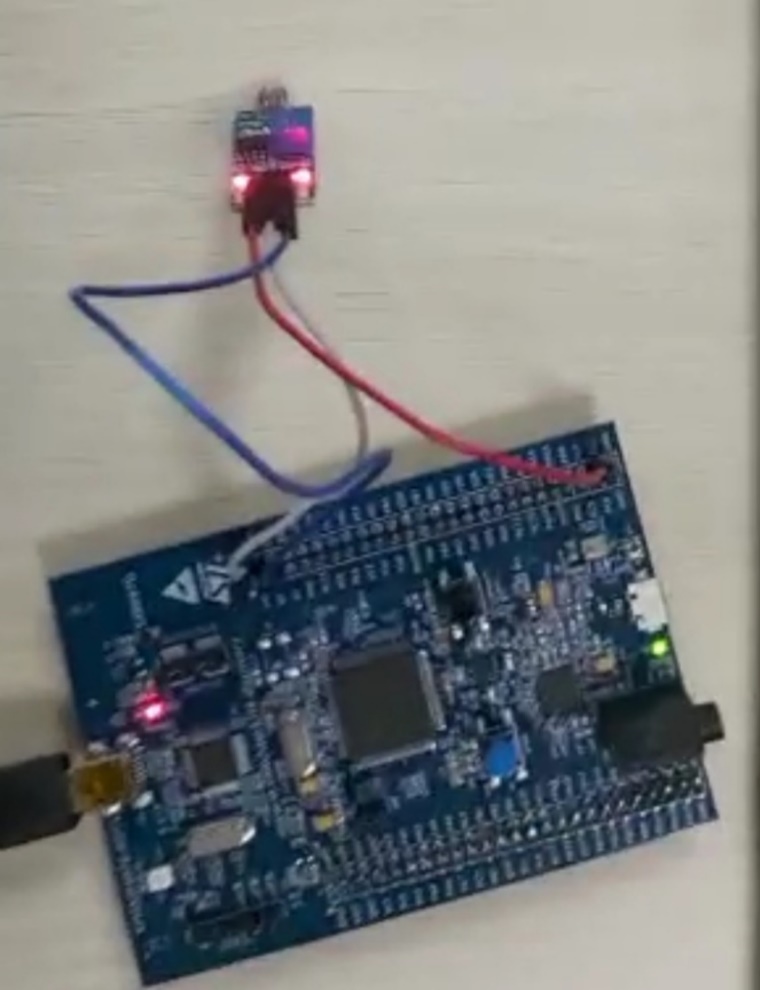
**Adaptive light control using LDR:**





**Car Reverse Gear Buzzer:**





**TEST PLAN:**

**HIGH LEVEL TEST PLAN:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL No** | **TEST\_ID** | **TESTING FUNCTION** | **EXPECTED INPUT** | **EXPECTED OUTPUT** |
| 1 | HLR\_1 | Wiper Control Module | When moisture sensor senses the water, it sends signal to wiper shield to turn on | Wiper shield will work properly |
| 2 | HLR\_2 | Interior Light Control Module | Any kind of motion detected in the PIR Motion Sensor (for example when the door is open) | The interior lighting will work properly |
| 3 | HLR\_3 | Adaptive light Control Module | LDR sensor sense the presence and absence of light is necessary | Turn on/off street lights when needed |
| 4 | HLR\_4 | Car Reverse Gear Buzzer | Car Reverse Radar Sensor detect the objects during reversing the vehicle and sends signal to buzzer | Buzzer starts beeping when the person in vehicle uses reverse gear |

**Low Level Test Plans:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SL No** | **TEST\_ID** | **TESTING FUNCTION** | **EXPECTED INPUT** | **EXPECTED OUTPUT** |
| 1 | HLR\_1-LLR1 | Wiper Control Module | When the input is 1 i.e. When the moisture sensor senses the water | The green LED will glow i.e. the GPIO pin with PORT D 12 will be set |
| 2 | HLR\_1-LLR2 | Wiper Control Module | When the input is 0 i.e. When no water is detected from moisture sensor | The green LED will stop i.e. the GPIO pin with PORT D 12 will be reset. |
| 3 | HLR\_2-LLR1 | Interior Light Control Module | When the input Is 1 i.e. when the motion is detected | The green LED will glow i.e. the GPIO pin with PORT D 12 will be set |
| 4 | HLR\_2-LLR2 | Interior Light Control Module | When the input Is 0 i.e. when no motion is detected | The green LED will stop i.e. the GPIO pin with PORT D 12 will be reset. |
| 5 | HLR\_3-LLR1 | Adaptive light Control Module | When the input is 1 i.e. When the darkness is detected around | The green LED will glow i.e. the GPIO pin with PORT D 12 will be set |
| 6 | HLR\_3-LLR2 | Adaptive light Control Module | When the input is 0 i.e. When the lights are detected around | The green LED will stop i.e. the GPIO pin with PORT D 12 will be reset. |
| 7 | HLR\_4-LLR1 | Car Reverse Gear Buzzer | When the input is 1 i.e. When the person in a vehicle uses reverse gear | The green LED will glow i.e. the GPIO pin with PORT D 12 will be set |
| 8 | HLR\_4-LLR2 | Car Reverse Gear Buzzer | When the input is 0 i.e. When the reverse gear is not in use | The green LED will stop i.e. the GPIO pin with PORT D 12 will be reset. |

## **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.

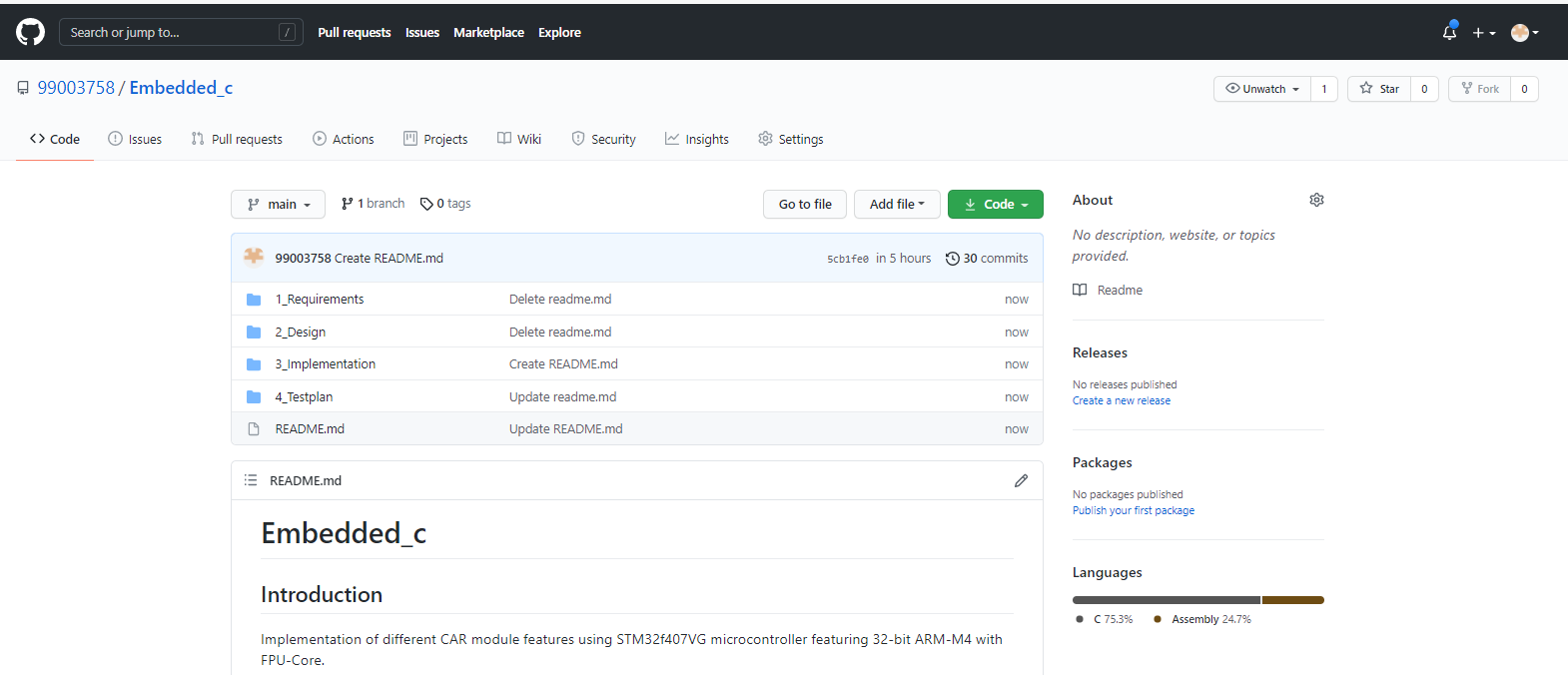
Features that I have implemented and pin configurations are as follows: -

* Wiper Control Module:
* PD12: Denotes the wiper control module status (SET (1): On, RESET (0): Off)
* Interior light control module using PIR sensor:
* PD12: Denotes the interior light control status (SET (1): On, RESET (0): Off)
* Adaptive light control using LDR:
* PD12: Denotes the exterior light control status (SET (1): On, RESET (0): Off)
* Car reverse gear buzzer:
* PD12: Denotes the reverse gear buzzer status (SET (1): On, RESET (0): Off)

### Git Link

<https://github.com/99003758/Embedded_c.git>

### Git Dashboard



### 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 door lock/unlock 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.

### 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.

## **Individual Contribution & Highlights**

1. My contribution to this is project is to design features like Adaptive Light Control System using LDR, Car Reverse Gear Buzzer

For Adaptive Light Control, I have used STM32 microcontroller which is connected to LDR sensor

* When the LDR sensor detects the dark, the GPIO pin PD12 is set, the LED will be turned on indicating that it light needs to be switched on
* When the LDR sensor detects the dark, the GPIO pin PD10 is reset, the LED will be turned on indicating that it light needs to be switched off

For Car Reverse Gear Buzzer, I have used STM32 microcontroller

* when the gear comes to reverse gear mode (reverse movement), the GPIO pin PD12 is set, and the buzzer will be turned on
* when the gear comes normal gear mode (forward movement), the GPIO pin PD12 is reset, and the buzzer will be turned on

### Summary

Challenges faced and how were they overcome

* In some systems STM32 microcontroller board is working and in some systems, it’s not so we have tested multiple microcontrollers
* Even few sensors didn’t work properly so we have tested with multiple sensors
* There were even few issues with our code at the beginning, after proper debugging we are able to eliminate the errors.

# Miniproject -4 [Individual] – Kernel Programming and Device Drivers

## **4.1 Module/s:**

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

## **4.2 Topic and Subtopics:**

* Basic Linux commands.
* Qemu Based Emulation.
* Creation of SD card.
* Building custom Kernel.
* Cross Compilation.
* Static and dynamic libraries.
* System calls.
* Adding system calls in kernel space.
* Invoking system calls from user space.
* Kernel modules.
  + In-Tree modules: Dynamic.
  + In-tree modules: static.
* Basics of Kernel Device Drivers.
* Registering Char Driver.
* Kernel Data Structure.
  + Kfifo API.
  + List API.
* IPC Kernel
  + Concurrency.
    - Kernel Threads.
  + Locking and Synchronization.
    - Mutex.
    - Semaphore.
    - Spinlocks.
    - Wait queues.
* IOCTL.
* Driver model.

## **4.3 Objectives & Requirements:**

The main objective of this module is to apply the concepts of Linux kernel, kernel device drivers to develop:

* Custom kernel.
* Create char drivers.
* Developing cross compiled code for target qemu.
* Creating own system calls.

### 4.3.1 Requirements:

* Basic Linux commands.
* Programming in Linux Environment.
* Custom kernel.
  + zImage
  + vexpress-v2p-ca9.dtb
  + rootfs.img
* Operating system Basics.
* IPC concepts.
* Concurrency.
* File handling using system calls.
* Virtual Memory concept.

## **4.4 Implementation Summary:**

### 4.4.1 Hands-on Activity that are implemented are as follow:

* Register char driver
* Register file operations
* Device Create, Class Create
* Read, write operations using global buffer
* Read, write operations using kfifo.
* ioctl operations, returning length/remaining space, reset operation
* ioctl operations - filling length/remaining space in structure
* synchronization in char driver - using wait queue

### 4.4.2 User space code:

* simple read, write
* multiple read, multiple write
* User space code for IOCTL operations

### 4.4.3 kthread examples:

* simple two threads
* Race condition scenarios
* Mutual exclusion using semaphore, mutex, spinlock
* Synchronization using semaphores, wait queues
* Device Tree based platform driver code -- dummy UART
* Activity that are implemented are as follow:
* System calls -- echo back the given string.
* System calls—traverse process list print pid and ppid.
* System calls—length of string.
* System calls—taking simple parameter.
* IOCTL operation traverse the list.

## **4.5 Git Link:**

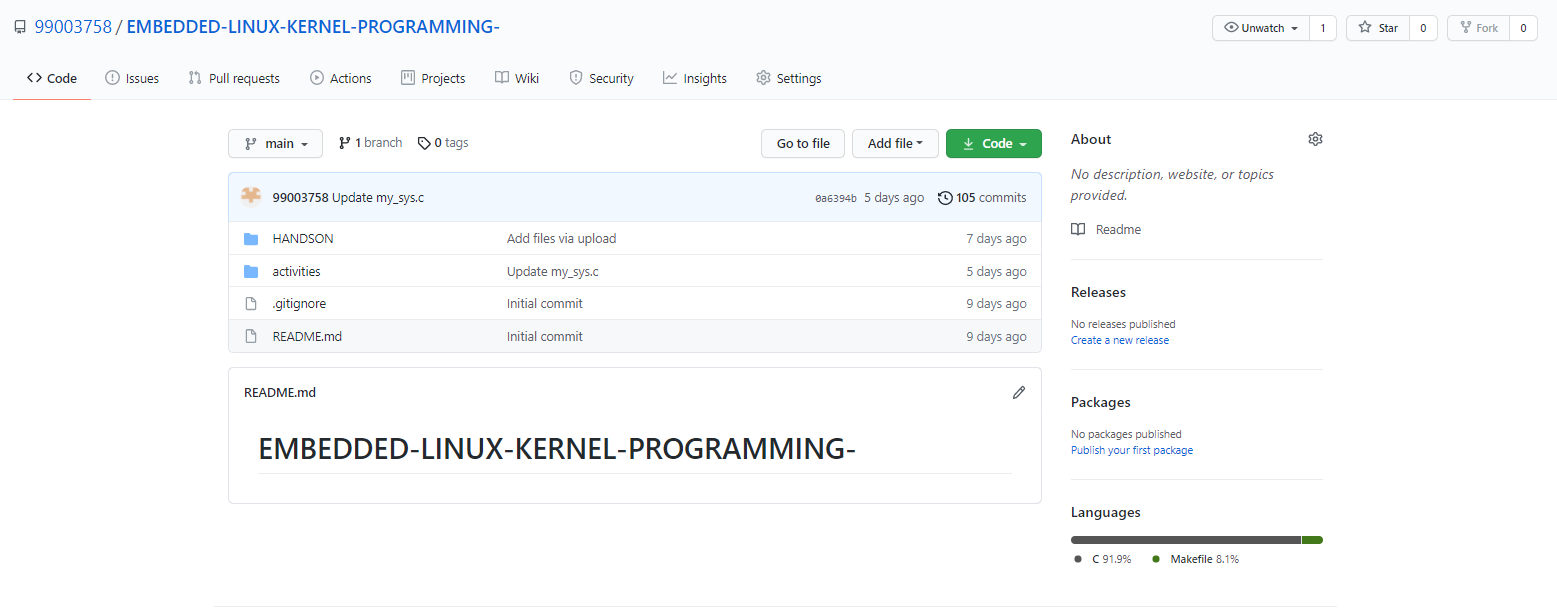
<https://github.com/99003758/EMBEDDED-LINUX-KERNEL-PROGRAMMING-.git>

## **4.6 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.

**4.7** **Git Dashboard:**

****

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

* Unable to directly access string in kernel space from userspace 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.