./

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](#_Toc55470819)

[Miniproject -1 [Team/Individual] 4](#_Toc55470820)

[Module/s 4](#_Toc55470821)

[Topic and Subtopics 4](#_Toc55470822)

[Objectives & Requirements 4](#_Toc55470823)

[Design 4](#_Toc55470824)

[Test Plan 4](#_Toc55470825)

[Implementation Summary 4](#_Toc55470826)

[Video Summary 4](#_Toc55470827)

[Git Link 4](#_Toc55470828)

[Git Dashboard 4](#_Toc55470829)

[Summary 4](#_Toc55470830)

[Individual Contribution & Highlights 5](#_Toc55470831)

[Summary 5](#_Toc55470832)

[Challenges faced and how were they overcome 5](#_Toc55470833)

[Future Scope (If applicable) 5](#_Toc55470834)

[Miniproject -2 [Team/Individual] 6](#_Toc55470835)

[Module/s 6](#_Toc55470836)

[Topic and Subtopics 6](#_Toc55470837)

[Objectives & Requirements 6](#_Toc55470838)

[Design 6](#_Toc55470839)

[Test Plan 6](#_Toc55470840)

[Implementation Summary 6](#_Toc55470841)

[Git Link 6](#_Toc55470842)

[Git Dashboard 6](#_Toc55470843)

[Summary 6](#_Toc55470844)

[Individual Contribution & Highlights 6](#_Toc55470845)

[Summary 6](#_Toc55470846)

[Challenges faced and how were they overcome 6](#_Toc55470847)

# Miniproject -1 [Team]

## **Module:**

“SDLC (System Development Life Cycle)”

### Topic and Subtopics:

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

All these topics had been implemented using V-model.

## **Objective:**

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

## **Requirements:**

***High Level Requirement:***

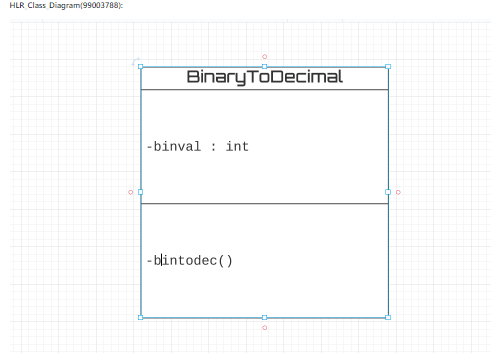
1. Maximum input digits a user can enter is up to 12 digits.
2. Arithmetic Operations
3. Exponential and Logarithmic Operations.
4. Square roots and powers.
5. Permutation and Combinations.
6. Finding area of different geometrical shapes like circle, square, rectangle, triangle.
7. Finding volume of different geometrical shapes like cube, cylinder, cone, sphere.
8. Trigonometric and Inverse Trigonometric Functions.
9. Measurements conversion (km, cm, inch, etc.).
10. weight conversion (kilogram, liter, etc.).
11. Mixed to improper fractions.
12. Cube and cube root.
13. Temperature conversion.
14. Arithmetic operations of fractions. (+,-,\*,/) .
15. Calculation of Reminder.
16. Calculation of simple interest.
17. Binary to decimal conversion

***Low Level Requirement:***

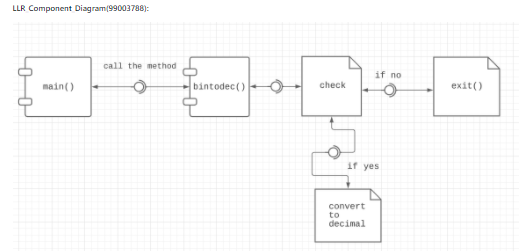
1. Exponential operation (result=b^x where b=base and x=exponent. Input type: integer and float) Logarithmic operation (result=log(x) where the input type of x is integer or float) .
2. Area of different geometrical shapes (The input data type used in this scenario are integer, float and long).
3. Volume of different geometrical shapes (The input data type used in the scenario are integer and float.)
4. Permutation and Combination (result=nPr; input type: integer and result=nCr; input type: integer).
5. Using float for Decimal values.
6. Using int for Integer values.
7. Using Double.
8. Calculation of reminder using modulus operator.
9. Using arithmetic operators for calculation of simple interest.

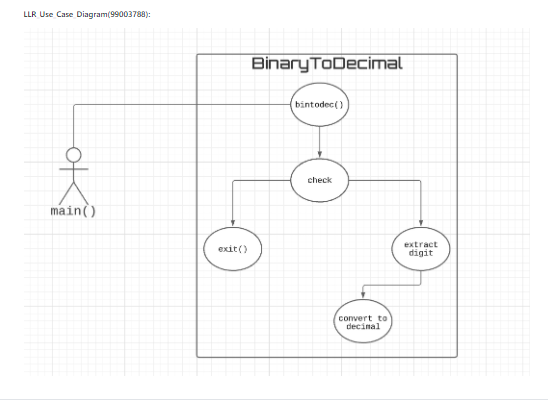
## **Design**

* High Level UML Diagram:

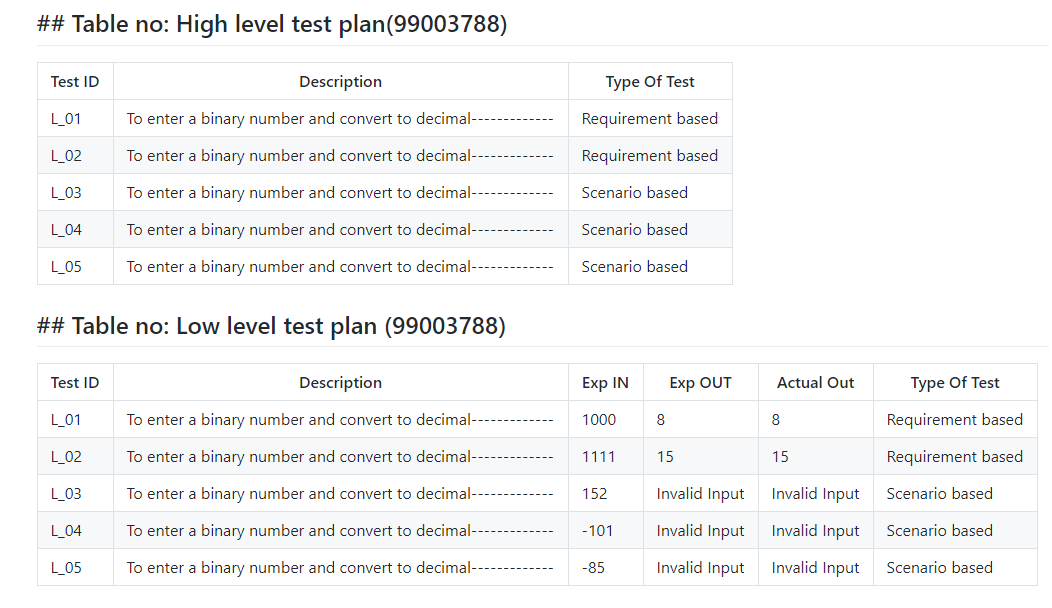


* Low Level UML Diagram:





## Test Plan



## Implementation 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.

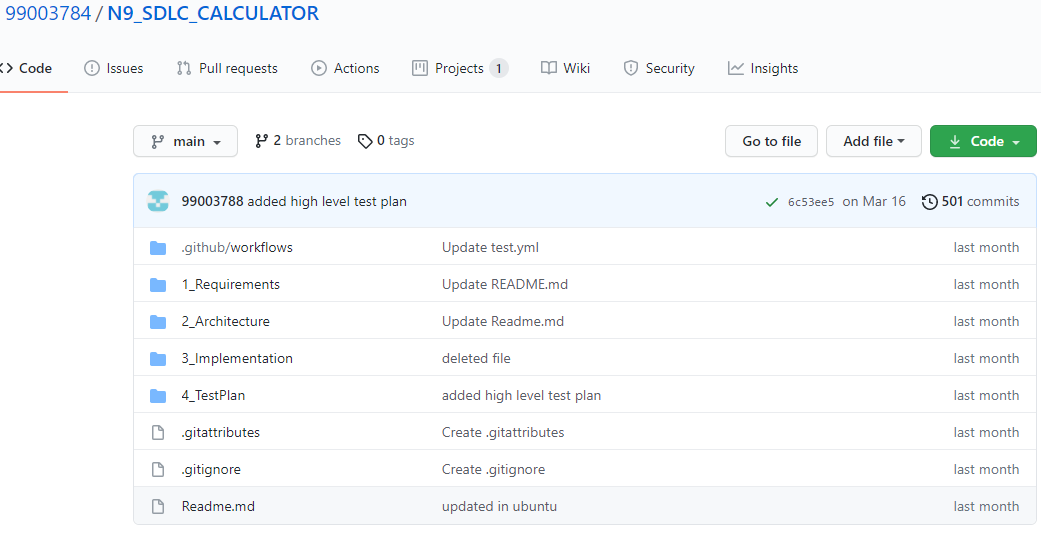
### Video Summary

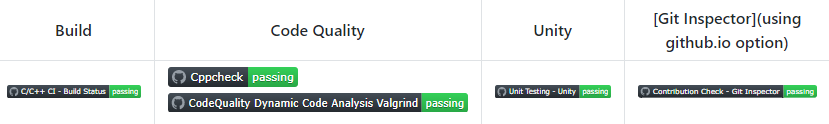
“Please upload a short video on the repo for the walkthrough of the project (Team/Individual) less than 7min and less than 30MB File Size. Start is the Standard opening slide with title of miniproject + Team members followed by the walkthrough”

### Git Link

<https://github.com/99003784/N9_SDLC_CALCULATOR.git>

### Git Dashboard

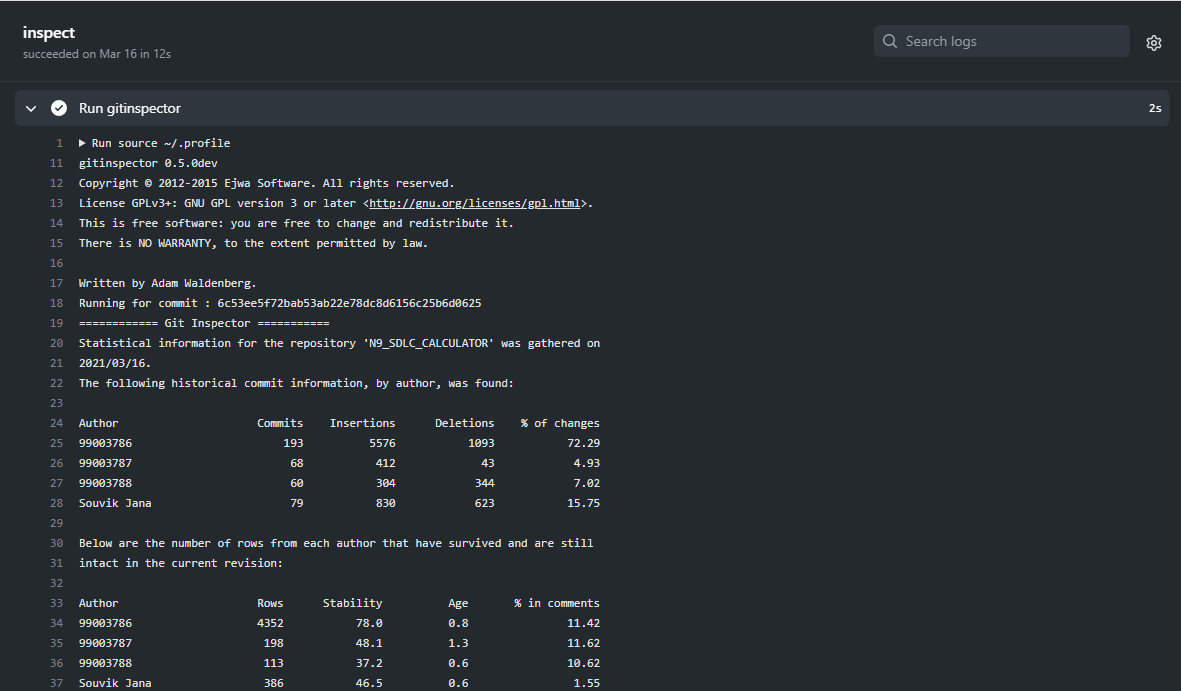
* ***GitHub Repo***
* ***Badges***

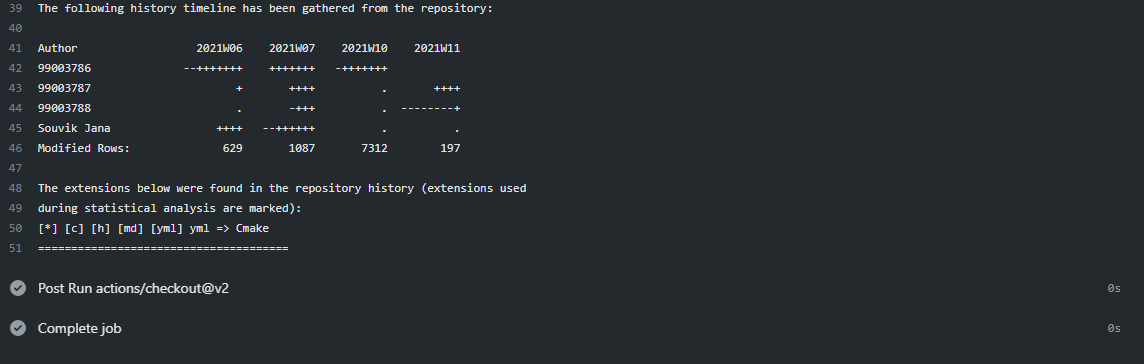
******

### Summary

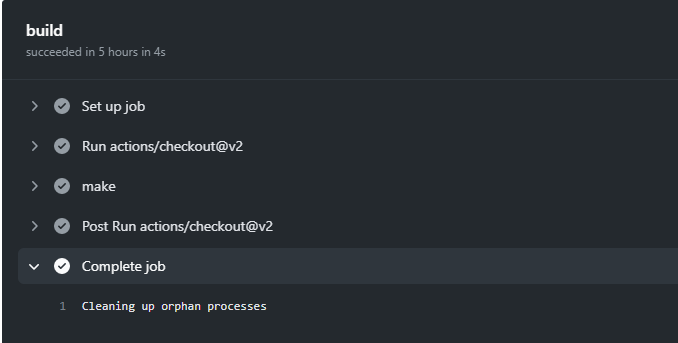
It is a simple electronic hardware/software device that can perform the simple calculations such as addition, subtraction, multiplication, division, calculating power of number, exponential function, logarithmic function, permutation and combination, trigonometry, inverse-trigonometric functions, factorial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

#### Git inspector summary

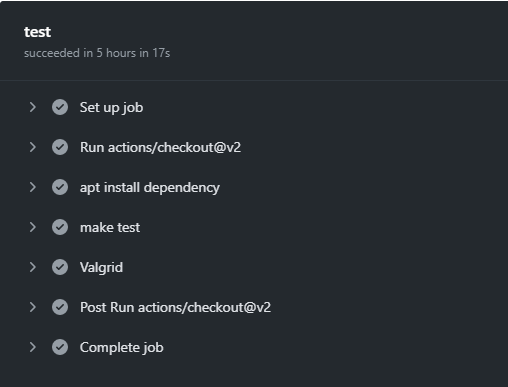


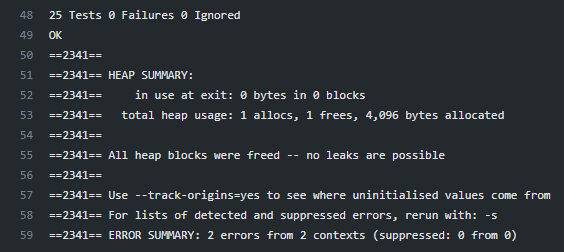


#### Build



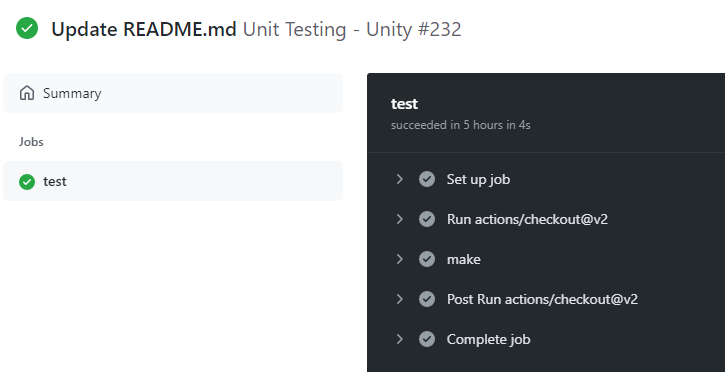
#### Code quality and Issues or Bug Tracking

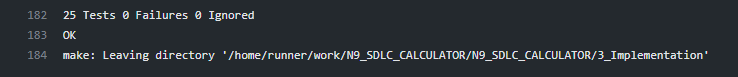
******



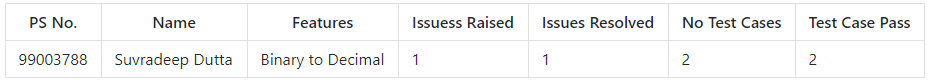
#### Unit Testing

“Unit Testing setup alignment with test plans and summary of outcome”





## Individual Contribution & Highlights

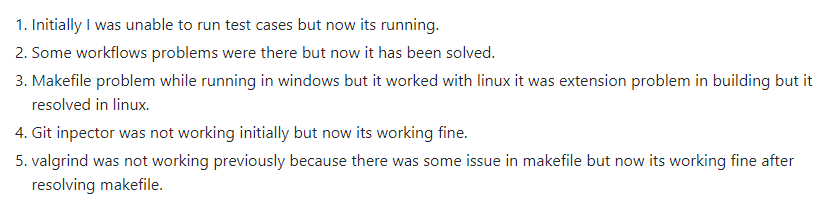


The function that I have implemented is conversion form binary to decimal. Here, first we need to accept a number and check whether it is greater than equal to 0 or not. If not then print “Invalid input”. If yes then the next step is to check whether the number is binary or not. If not the print “invalid Input”. If the number is binary, covert the binary number to its decimal equivalent. Finally print the decimal value.

### 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.

### Challenges faced and how were they overcome



### Future Scope (If applicable)

# Miniproject -2 [Team]

## Module

Embedded C

### Topic and Subtopics

* Development process pf embedded software.
* Introduction to target board and GPIO programming
* Interrupts
* ADC, USART, SPI and I2C protocols
* Coding Guidelines and optimization

## Objectives

## & Requirements

## Design

## Test Plan

## Implementation Summary

### Git Link

### Git Dashboard

### Summary

#### Git inspector summary

#### Build

#### Code quality

#### Unit Testing

#### Issues

## Individual Contribution & Highlights

### Summary

### Challenges faced and how were they overcome

# Miniproject -3 [Individual]

## Module

Python Programming

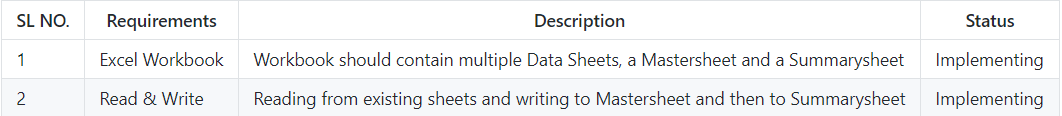
### Topic and Subtopics

## Objectives

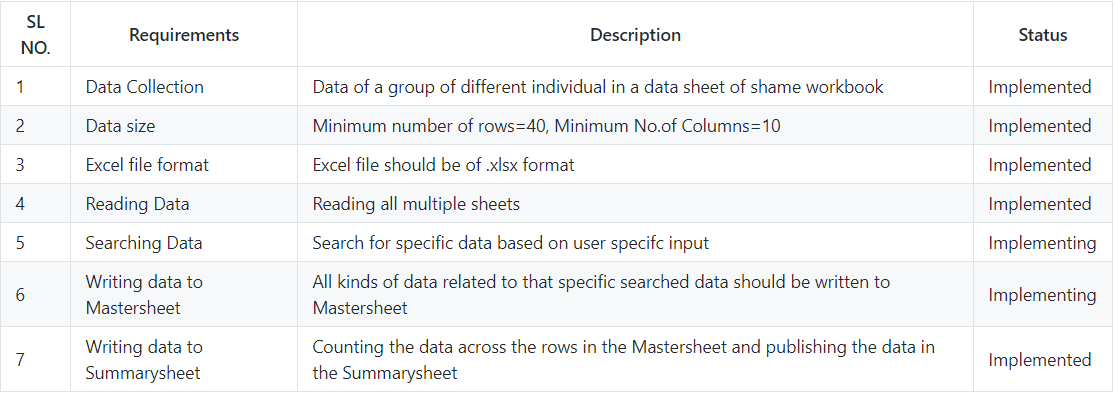
Analyzing data from 5 different excel spreadsheets and write specific data from all the excel sheets on a mastersheet and then counting number of entries in the mastersheet and write it in a summarysheet using Python Programming.

## Requirements

* High Level Requirements:

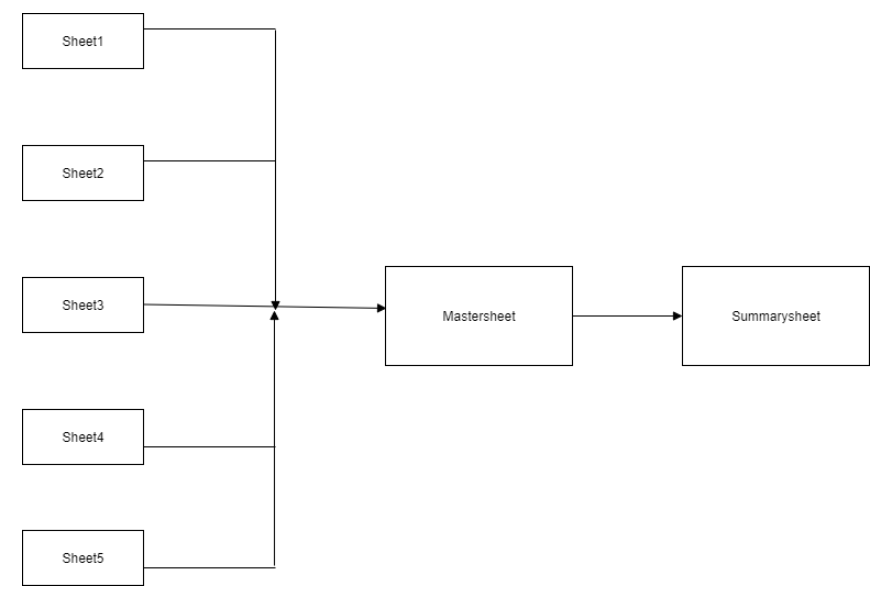


* Low Level Requirements:

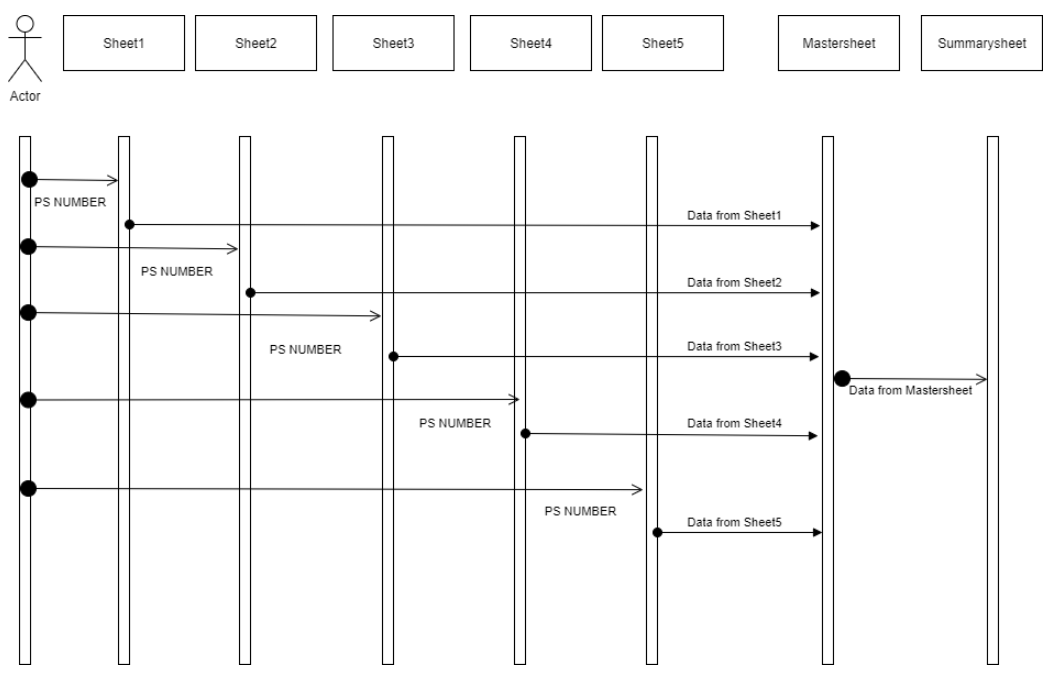


## Design

* High Level UML Diagram



* Low Level UML Diagram



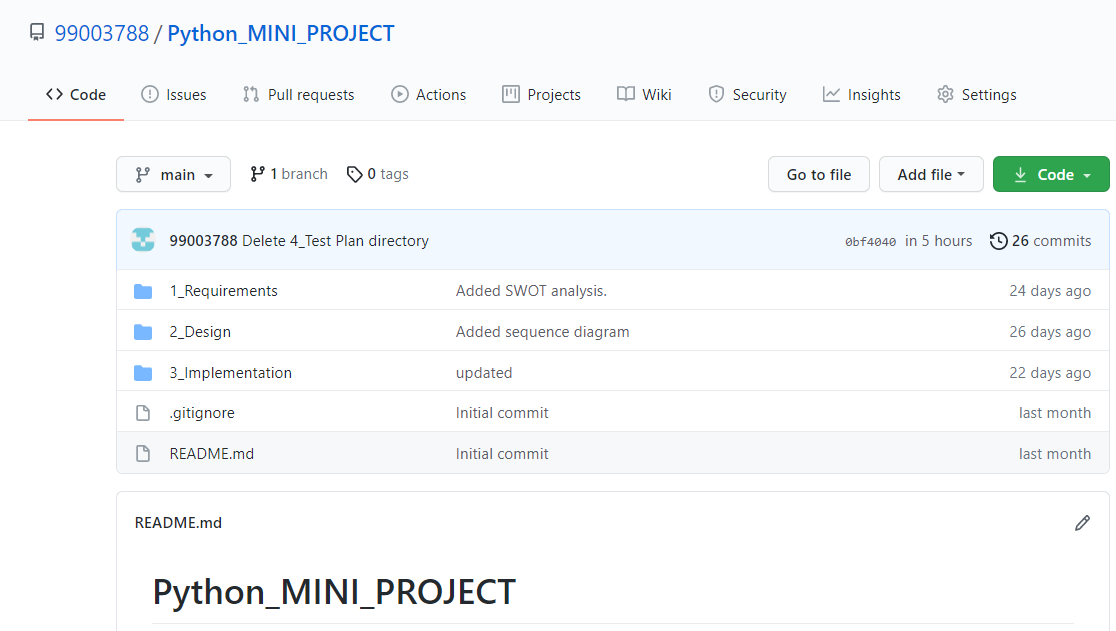
## Implementation Summary

Implementation folder has the python file with extension “.py” containing the code. It also contains the excel file which consists of 5 sheets.

### Git Link

<https://github.com/99003788/Python_MINI_PROJECT.git>

### Git Dashboard



### Summary

As per the objective, firstly I created an excel file that has five sheets. Then I started populating the sheets with data like NAME, PS NUMBER, Email, D.O.B, etc. The PS NUMBER column is common throughout all the five sheets.

Now coming to the code, here I have used openpyxl library. I have also followed Object Oriented Programming approach and used two classes. In the first class named ***accept\_value*** I have created a method ***take\_vlaue ()*** that gives the user a choice to select number of input and then asking him/her to enter the input(s). In the next class named ***main*** that inherits the ***accept\_value*** class***,*** am creating a method ***my\_func ().*** In this method, firstly am loading the excel file. Then making use of the PS NUMBER(S) entered by the user am searching all the data related to that PS NUMBER(S) and the dumping the entire date collected from the 5 sheets into a existing sheet named ***mastersheet***. After that I am counting the number of rows and columns in the mastersheet and then reflecting the result into a sheet named summarysheet.

#### Code quality

#### Unit Testing

#### Issues

## Individual Contribution & Highlights

### Summary

### Challenges faced and how were they overcome

# Miniproject -4 [Individual]

## Module/s

### Topic and Subtopics

## Objectives & Requirements

## Design

## Test Plan

## Implementation Summary

### Git Link

### Git Dashboard

### Summary

#### Git inspector summary

#### Build

#### Code quality

#### Unit Testing

#### Issues

## Individual Contribution & Highlights

### Summary

### Challenges faced and how were they overcome