./



|  |  |  |
| --- | --- | --- |
| **Name** | **PS Number** | **Email ID** |
| Asha N | 99002646 | asha.n@ltts.com |

**Document History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **To be approved By** | **Remarks/Revision Details** |
| 1 | 15/10/2020 | Asha N | Akash Shetty and Chappidi Lokanath Reddy | Dr Prithvi Sekhar |  |
| 2 | 16/10/2020 | Asha N | Akash Shetty and Chappidi Lokanath Reddy | Dr Prithvi Sekhar |  |
| 3 |  | Asha N |  |  |  |

**Table of Contents**

**INDIVIDUAL** [**ACTIVITY - 1 : BEVERAGE DISPENSER**](#_heading=h.1fob9te)

**1.** [**INTRODUCTION**](#_heading=h.3znysh7)……………………………………………………………………………………………………..

**2.** [**RESEARCH**](#_heading=h.44sinio)…………………………………………………………………………………………………………….**11**

2.1 [AGING](#_heading=h.44sinio) …………………………………………………………………………………………………………..

2.2 [COSTING](#_heading=h.44sinio)…………………………………………………………………………………………………………….

**3.** [**DEFINITION OF THE PRODUCT**](#_heading=h.2jxsxqh)…………………………………………………………………………………..

**4. SWOT ANALYSIS**…………………………………………………………………………………………………...

**5. REQUIREMENT ANALYSIS**……………………………………………………………………………..

5.1 [HIGH LEVEL REQUIREMENTS](#_heading=h.z337ya)……………………………………………………………………...

5.2 LOW LEVEL REQUIREMENTS…………………………………………………………………………..

**6. DESIGN ANALYSIS**……………………………………………………………...

6.1 [BEHAVIORAL DIAGRAM](#_heading=h.2s8eyo1)S……………………………………………………………………………………..

6.2 [STRUCTURAL DIAGRAM](#_heading=h.1y810tw)S……………………………………………………………………………………….

**7.** [**TEST PLAN**](#_heading=h.4i7ojhp)………………………………………………………………………………………………………

7.1 [UNIT TESTING](#_heading=h.3rdcrjn)………………………………………………………………………………………………….

7.2 [INTEGRATION TESTING](#_heading=h.26in1rg)………………………………………………………………………………….

**INDIVIDUAL** [**ACTIVITY - 2 :**](#_heading=h.1fob9te) **V-MODEL ON C++ PROJECT**

**1.** [**INTRODUCTION**](#_heading=h.3znysh7)……………………………………………………………………………………………………...

1.1 DEFINITION ……………………………………………………………………………………………………

1.2 FEATURES…………………………………………………………………………………………….

**2. REQUIREMENT ANALYSIS**…………………………………………………………………………………..

2.1 [HIGH LEVEL REQUIREMENTS](#_heading=h.z337ya)………………………………………………………………………….

2.2 LOW [LEVEL REQUIREMENTS](#_heading=h.z337ya)……………………………………………………………………….

**3. DESIGN ANALYSIS**……………………………………………………………………………………...

3.1 [BEHAVIORAL DIAGRAM](#_heading=h.2s8eyo1)S………………………………………………………………………………..

3.2 [STRUCTURAL DIAGRAM](#_heading=h.1y810tw)S………………………………………………………………………………..

**4. GIT ASPECTS…………………………………………………………………………...**

**5.** [**TEST PLAN**](#_heading=h.4i7ojhp)……………………………………………………………………………………………………...

5.1 [UNIT TESTING](#_heading=h.3rdcrjn)……………………………………………………………………………………………….

5.2 [INTEGRATION TESTING](#_heading=h.26in1rg)………………………………………………………………………………………

**6. CONCLUSION……………………………………………………...**

**INDIVIDUAL** [**ACTIVITY - 3 :**](#_heading=h.1fob9te) **AGILE MODEL ON** [**BEVERAGE DISPENSER**](#_heading=h.1fob9te)

1. THEME……………………………………………………………………………………………………..

2. EPIC………………………………………………………………………………………………..

3. USER STORY AND SPRINTS…………………………………………………………………..

**TEAM ACTIVITY - 1 : IMPACT OF DEFECTIVE PRODUCT(AEROSPACE)**

1. INCIDENTS…………………………………………………………………………...
2. CAUSES…………………………………………………………………………………….
3. IMPACTS……………………………………………………………………………………….

**TEAM ACTIVITY - 2 : POWER OF VISUAL REPRESENTATION**

1. SysML vs UML**…………………………………………………………………………..**

**TEAM ACTIVITY - 3 : AGILE METHODOLOGY**

1. WHAT IS AGILE MANIFESTO?.......................................................................................................
2. WHAT IS AGILE PRINCIPLE?.........................................................................................................................
3. WHAT IS AGILE ROLES?......................................................................................................................................
4. WHAT IS AGILE CEREMONIES?............................................................................................................
5. SPRINT PLANNING……………………………………………………………………………………..
6. SPRINT REVIEW……………………………………………………………………………………………
7. SPRINT RETROSPECTIVE………………………………………………………………………………
8. DAILY SCRUM……………………………………………………………………………………………….
9. INCREMENT……………………………………………………………………………………………
10. PRODUCT BACKLOG……………………………………………………………………………………………..
11. PRINT BACKLOG……………………………………………………………………………………………..
12. WHAT IS AGILE TOOLS?...........................................................................................................................

**TEAM ACTIVITY - 4 : SIMPLE CALCULATOR**

**1. INTRODUCTION……………………………………….**

**2. FEATURE…………………………………**

**3. REQUIREMENT ANALYSIS**…………………………………………………………………………………..

3.1 [HIGH LEVEL REQUIREMENTS](#_heading=h.z337ya)………………………………………………………………………….

3.2 LOW [LEVEL REQUIREMENTS](#_heading=h.z337ya)……………………………………………………………………….

**4. DESIGN ANALYSIS**……………………………………………………………………………………...

**5.** [**TEST PLAN**](#_heading=h.4i7ojhp)……………………………………………………………………………………………………...

5.1 Requirement based testing

5.2 Boundary condition

5.3 Scenario based

**6. GIT ASPECTS……………………………………………………………………………**

**7.** SDLC for Simple Calculator Agile…………………………………..

**TEAM ACTIVITY - 5 : TOOLS**

[**REFERENCE**](#_heading=h.35nkun2)**S :**

**INDIVIDUAL** [**ACTIVITY - 1 : BEVERAGE DISPENSER**](#_heading=h.1fob9te)

**1.** [**INTRODUCTION**](#_heading=h.3znysh7)

Beverage Dispenser is a vending machine that gives beverages like coffee, milk, tea, hot water, cream, sugar to users after cash is inserted into the machine. Some of the machines, notably older models, utilize powdered instant coffee  mixed with hot water, and a few of those provide condiments like cream and sugar. Some newer models fresh-brew the coffee using hot water and ground coffee beans, and some also grind the coffee to order using coffee grinders installed in the machines, as well as providing various condiments. Some modern machines also provide other hot drinks such as tea, espresso, lattes, cappuccinos, mochas and hot chocolate. Some of the machines dispense canned coffee, and some dispense both hot coffee and iced coffee.

**2.** [**RESEARCH**](#_heading=h.44sinio)

**2.1** [**AGING**](#_heading=h.44sinio)

The first vending machine in the U.S. was built in 1888 by the Thomas Adams Gum Company, The first modern coin-operated vending machines were introduced in London, England in the early 1880s, dispensing postcards.

Vending machines exist in many countries and, in more recent times, specialized vending machines that provide less common products compared to traditional vending machine items have been created.

**2.2** [**COSTING**](#_heading=h.44sinio)

Coin Operated Vending Machine (Floor standing vending machine)

Type: Food and Drinks

Environmental Condition: Semi-outdoor

Payment: Coin, Bill, Cashless Payment

Charge System: Coin and Note

Function: Insulation, Heating

Touch Screen: Touch Screen

Specification: 1920x720x620mm

Power-off Protection: With Power-off Protection

Powder Canister: 3 KG/ Canister

No. Of Canister: 1 Bean Canister+4 Powder Canisters

Temperature Control System: Hot drinks 105 ºC max

Water Supplying: Pump

Price: US $2,500-3,000

2. Tabletop Coffee-Tea Vendor

LCD Display: 14 inch touch screen

Power-off Protection: With Power-off Protection

Maximum Power: 2700W

Standard cup size: 10oz(80mm diameter cup size)

Grinder: Ditting

Coffee brewer: Jetinno patent

Daily maintenance: Automatically clearance

Espresso drink speed: 45s

Instant Drinks Speed: 25S (120ml)

Cup capacity: 120cups per day

Price: US $880.00-$890.00

**3.** [**DEFINITION OF THE PRODUCT**](#_heading=h.2jxsxqh)

* Adding sensors to to measure beverages.
* Adding New features like Cold drinks and also Accepting exact amount of beverage.
* Implementing feature to select the amount distribution.
* Upgrading output method for different drinks

**4. SWOT ANALYSIS**

|  |  |
| --- | --- |
| **Strength** | **Weaknesses** |
| * Basic need of consumer * Different types of flavors * Low cost * Non-Alcoholic | * Age of life cycle * Time taken to register * Depend on power supply |
| **Opportunities** | **Threat** |
| * Replace alcoholic drinks * Growing possibilities if cold drinks * Upgrading output method for different drinks | * Competition with barista , Mochas , Gloria, Jean , Costa Coffee. * Presence of other ‘Hangout’ locations * Competition with Starbucks , Lavazza, Caribou coffee, Dinkin Donuts etc.. |

Table 2: SWOT Analysis

**5. REQUIREMENT ANALYSIS**

**5.1** [**HIGH LEVEL REQUIREMENTS**](#_heading=h.z337ya)

|  |  |
| --- | --- |
| **ID** | **Description** |
| HL\_01 | Different types of cold drinks. |
| HL\_02 | Different Output method for different drinks. |
| HL\_03 | Display the amount to pay. |
| HL\_04 | Display the selection panel to select amount distribution. |

Table 3: High Level Requirements

**5.2 LOW LEVEL REQUIREMENTS**

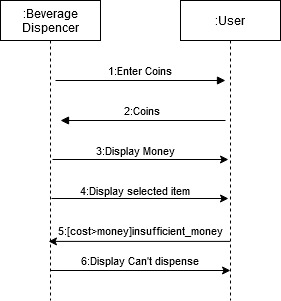
|  |  |
| --- | --- |
| **ID** | **Description** |
| LL\_01 | Sensor to measure beverages. |
| LL\_02 | Software maintenance. |
| LL\_03 | Depending on power. |

Table 4: Low Level Requirements

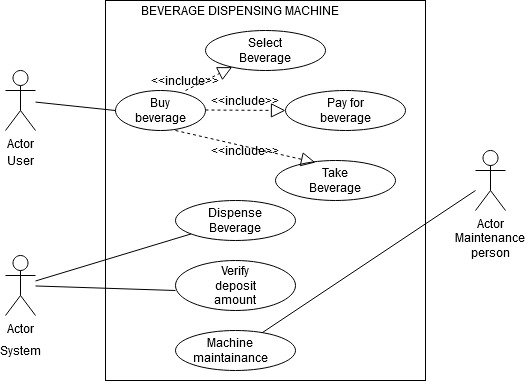
**6. DESIGN ANALYSIS**

**6.1** [**BEHAVIORAL DIAGRAM**](#_heading=h.2s8eyo1)**S**

**1. Sequence diagram**

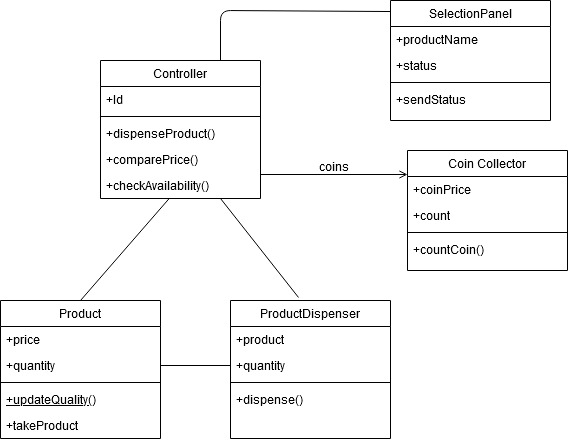


**2. Use case Diagram**

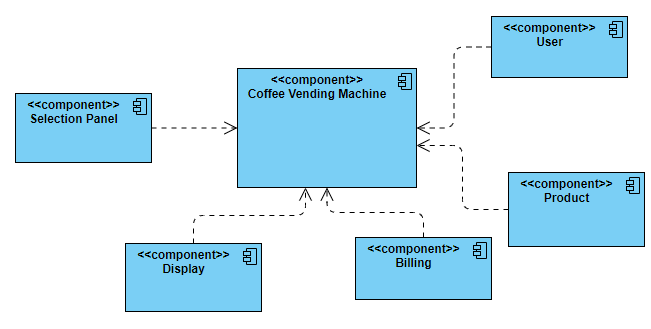


**6.2** [**STRUCTURAL DIAGRAM**](#_heading=h.1y810tw)**S**

1. **Class diagram**



1. **Component diagram**

****

**7. TEST PLAN**

**7.1** [**UNIT TESTING**](#_heading=h.3rdcrjn)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test id** | **Description** | **Expected input** | **Expected output** | **Actual output** |
| HH\_01 | On selection of cool drinks | different types of cool drinks | Display of different types of cool drinks | Display of different types of cool drinks |
| HH\_02 | Different output for cool drinks | Choosing between different types of cool drinks | Cool drinks should come from different path | Cool drinks should come from different path |
| HH\_03 | Display the amount to pay | Selection of beverage | Displays the correct amount | Displays the correct amount |
| HH\_04 | On display of money selection | Clicking on Menu | Display of different types of money distribution | Display of different types of money distribution |

Table 5: Unit Testing

**7.2** [**INTEGRATION TESTING**](#_heading=h.26in1rg)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test id** | **Description** | **Expected input** | **Expected output** | **Actual output** |
| LL\_01 | Sensor Implementation | To add Cup first | True | True |
| LL\_02 | On Software Maintenance | Maintenance person login | Display of quantity of cool drinks and coffee | Display of quantity of cool drinks and coffee |
| LL\_03 | Depends on Power | If power cut Immediately | Restart the machine | Restart the machine |

Table 6: Integration testing

**INDIVIDUAL** [**ACTIVITY - 2 :**](#_heading=h.1fob9te) **V-MODEL ON C++ PROJECT**

**1.** [**INTRODUCTION**](#_heading=h.3znysh7)

The Hospital Management System is focused on the principle of making appointments for patients. Here, the user can update doctors after logging in as an administrator. Other features include visualising the hospital's complete doctor data and making / attending appointments.

**1.1**  **DEFINITION**

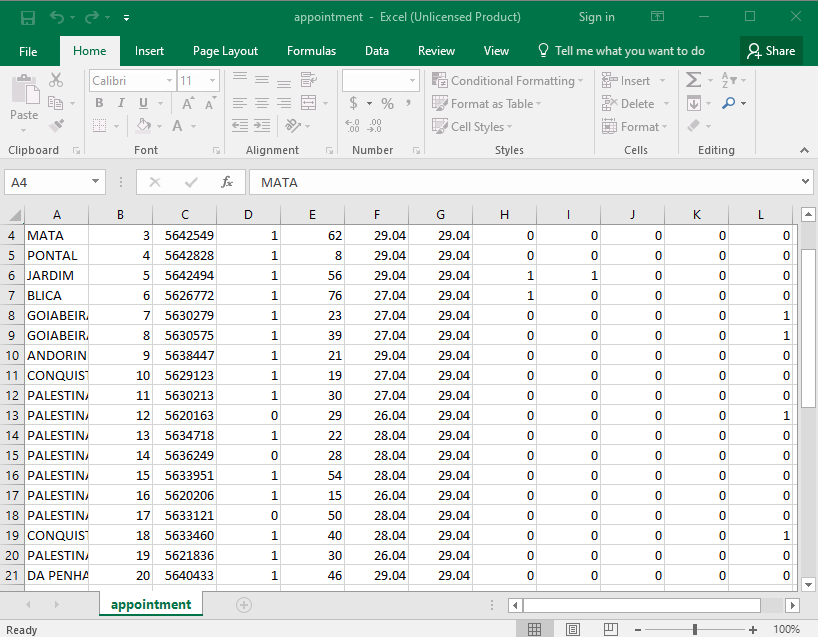
A user can view the complete data of the doctor, including ID number , name and appointment time. Having an appointment and attending it is the key aspect of this project. Before they move on it. The user must provide his / her name, choose a gender and provide a number. He/she can make an appointment quickly, but he/she needs to pick a doctor and enter the doctor's ID for appointments. Similarly, he / she must enter the number of the doctor when attending an appointment.

**1.2 FEATURES**

1. Login System
2. Make/Attend Appointment
3. Update Doctor
4. View all doctor’s information

**2. REQUIREMENTS ANALYSIS**

Input to the Doctor appointment is taken from a csv file where all the details stored in a specified format.



**2.1 High level requirements:**

|  |  |
| --- | --- |
| **ID** | **Description** |
| HL\_01 | Analysis of patient details. |
| HL\_02 | Comparison of different diseases |
| HL\_03 | Highest and lowest aged patients. |
| HL\_04 | Adding new patient details |

**2.2 Low level requirements:**

Low level requirements:

|  |  |
| --- | --- |
| ID | Description |
| LL\_01 | Reading data from csv file. |
| LL\_02 | Saving all data on list using STL concepts |
| LL\_03 | Implementation of CI/CD. |

Table 4: Low Level Requirements

**3. DESIGN ANALYSIS**

**3.1** [**BEHAVIORAL DIAGRAM**](#_heading=h.2s8eyo1)**S**

**Class Diagram**

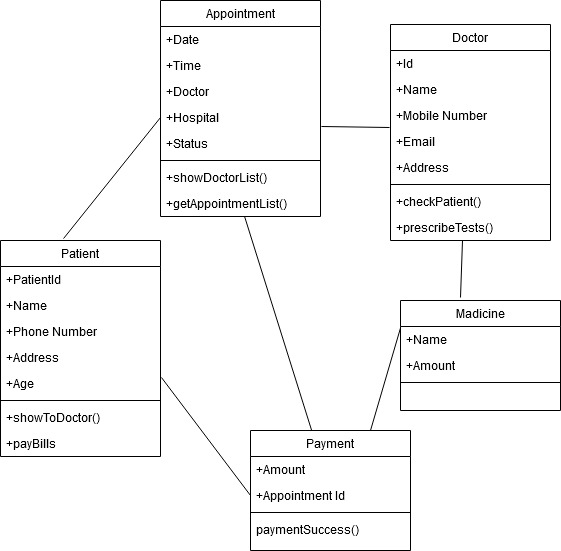
****

Figure : Class Diagram

**Use Case Diagram**

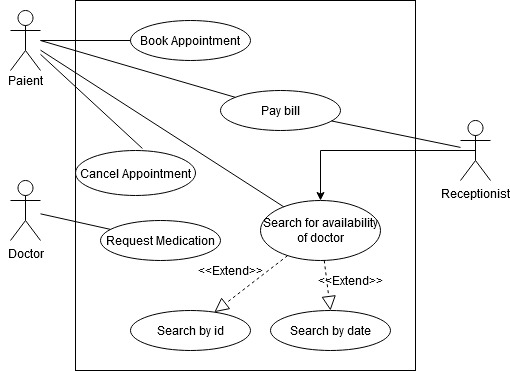
****

Figure :Use case diagram

**3.2** [**STRUCTURAL DIAGRAM**](#_heading=h.1y810tw)**S**

**Activity Diagram**

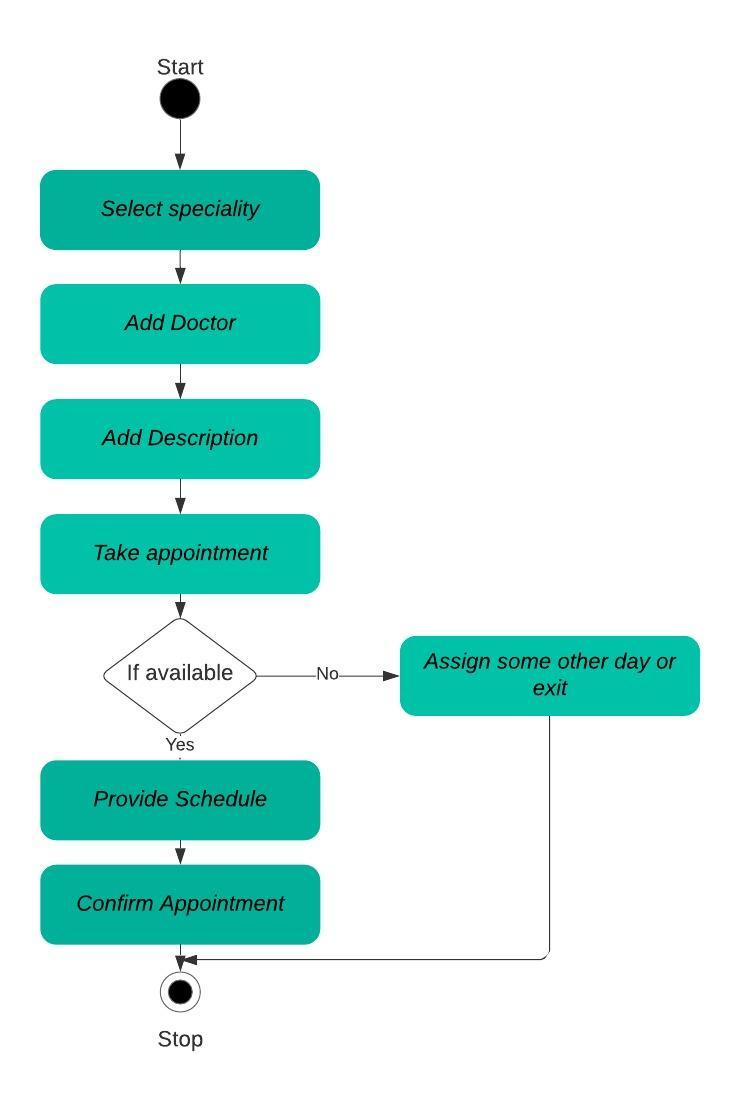


Figure : Activity diagram

**Component diagram**

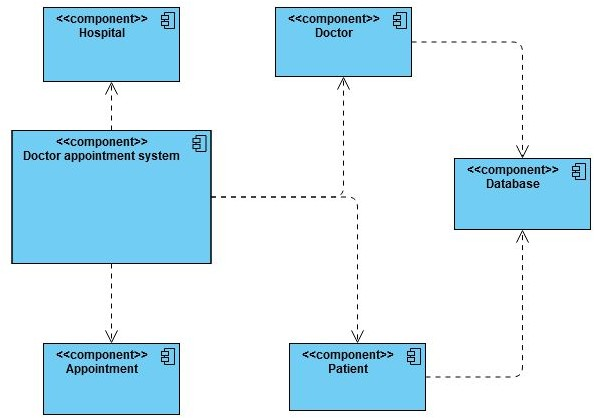
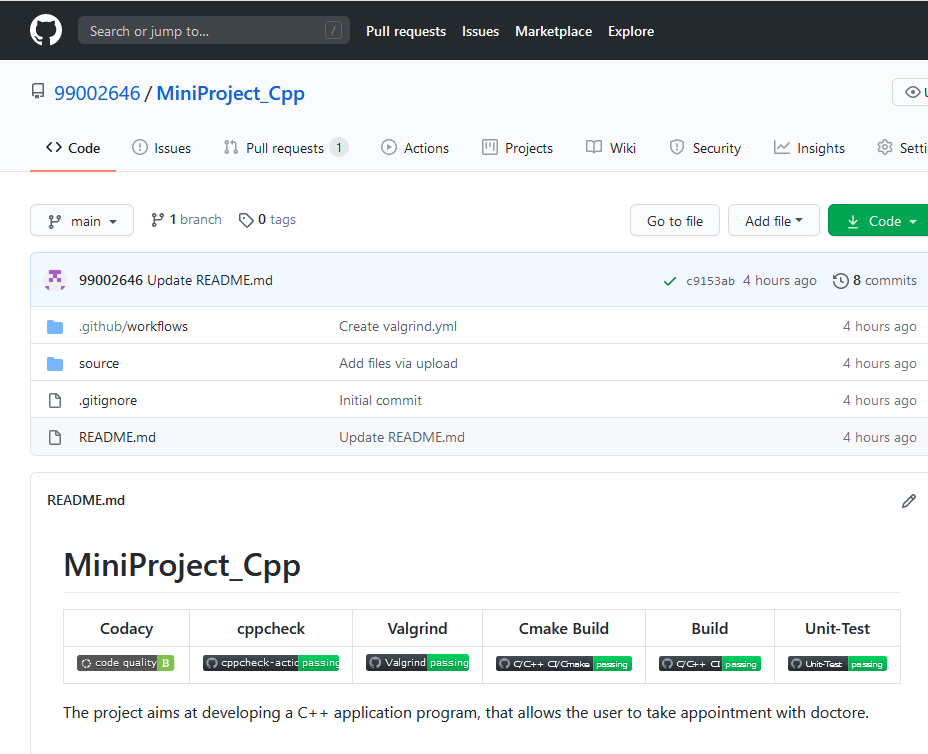


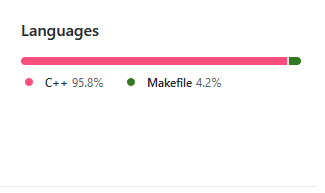
Figure : Component diagram

**4. GIT ASPECTS**

GitHub repo link : <https://github.com/99002646/MiniProject_Cpp>



Doctor appointment system github dashboard with badges



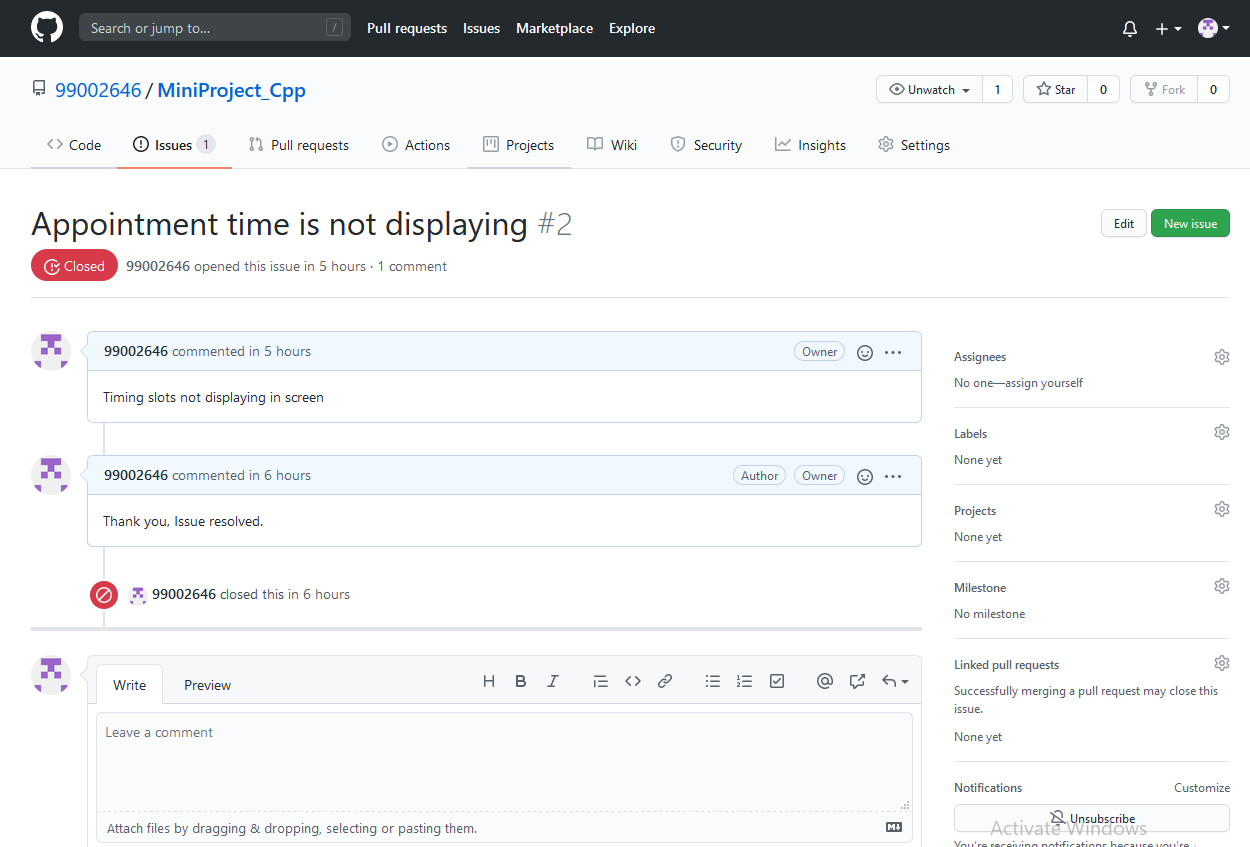


Figure : Issue snapshot

**5. TEST PLAN**

**5.1 UNIT TESTING**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test id | Description | Expected input | Expected output | Actual output |
| HH\_01 | Knowing of patient details. | Adding the data to list | Display of list where patient is added | Patient added |
| HH\_02 | Analysis of different diseases | Checking of different diseases | Printing of different diseases | True |
| HH\_03 | Highest aged patients | Giving patient name | Giving the highest aged patient name. | Year patient name |
| HH\_04 | Adding of new patient | Adding of new patient | Display of list where new patient is added | True |

**5.2 INTEGRATION TESTING**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test id | Description | Expected input | Expected output | Actual output |
| LL\_01 | Reading of csv file | Csv file | Adding of all data present in csv to list | Data added to list |
| LL\_02 | Adding data to list using STL concepts | Adding data to list | Data added to list | Display of list |
| LL\_03 | CI/CD | GitHub Actions | Cppcheck, valgrind, unit testing , codacy | Passing all CI/CD |

**6. CONCLUSION**

The Doctor Appointment System is moderned and improved system, so anyone from any place can book the appointment and meet the doctor. It is very useful.

**INDIVIDUAL** [**ACTIVITY - 3 :**](#_heading=h.1fob9te) **AGILE MODEL ON** [**BEVERAGE DISPENSER**](#_heading=h.1fob9te)

**1.Theme**

The beverage dispenser uses the grinder to grind the coffee beans so the coffee can be prepared fresh. With that the beverage dispenser dispenses the different types of cool drinks and also uses the different path to dispense different beverages.

**2.Epic**

* This begins when the customer wants to purchase drinks.
* The customer selects the drink.
* Then the dispenser shows the quantity of the drink.
* The customer selects the quantity and the next task will proceed.
* Dispenser checks for the availability of the drinks and shows the error if the condition is not satisfied.
* If the condition is satisfied then the dispenser displays the amount to pay.
* The customer should pay the correct amount.
* If the paid amount is lesser than the bill amount, then the error message will be displayed.
* If the customer pay the correct amount, then the drink which is selected by the customer is produced.
* Then the completion message will be displayed on the display.
* Then the machine completes the use case.

**3.User Story and sprints**

**Sprint-1 Amount not enough**

* If the paid amount is lesser than the bill amount, then the error message will be displayed.
* After the dispenser cancels the transaction.

**Sprint-2 Drinks not in stock**

* The customer selects the drinks he wants.
* The dispenser checks for the availability of drinks.
* The dispenser shows the error message.

**Sprint-3 Power cut**

* When the power cuts, the dispenser will shut.
* After the power connection comes back then the dispenser will start from the beginning.
* Then the path will be cleaned.

**TEAM ACTIVITY - 1 : IMPACT OF DEFECTIVE PRODUCT(AEROSPACE)**

**1. Incidents**

Boeing Recall – Dreamliner (2016): Failure of the engine.

Boeing Recall - Max 737 (2019): Faulty sensor gives erroneous data about positions.

Airbus Recall - A320neo, 2018: The unavailability of parts for repairs.

**2. Causes**

Erroneous data.

Glitches in the design.

Failure of maintenance.

**3. Impacts**

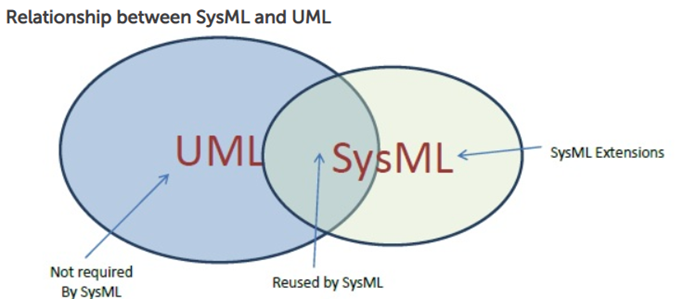
Recalls of more than 200.

357 casualties combined.

Market valuation decreases.

**TEAM ACTIVITY - 2 : POWER OF VISUAL REPRESENTATION**

The Systems Modelling Language (SysML) is a general purpose modelling language for engineering systems. SysML supports the analysis, design and verification of complex systems including hardware, software, information, personnel, procedures, and facilities in a graphical notation.SysML is defined as an extension of a subset of the Unified Modelling Language (UML) using UML’s profile mechanism.



* SysML is a comparatively small language that is easier to learn and apply. Since SysML removes many of UML's software-centric constructs, the overall language is smaller both in diagram types and total constructs.
* SysML allocation tables support common kinds of allocations. Whereas UML provides only limited support for tabular notations, SysML furnishes flexible allocation tables that support requirements allocation, functional allocation, and structural allocation. This capability facilitates automated verification and validation (V&V) and gap analysis.
* SysML model management constructs support models, views, and viewpoints. These constructs extend UML's capabilities and are architecturally aligned with IEEE-Std-1471-2000 (IEEE Recommended Practice for Architectural Description of Software Intensive Systems)
* SysML reuses seven of UML 2's fourteen diagrams, and adds two diagrams (requirement and parametric diagrams) for a total of nine diagram types. SysML also supports allocation tables, a tabular format that can be dynamically derived from SysML allocation relationships.

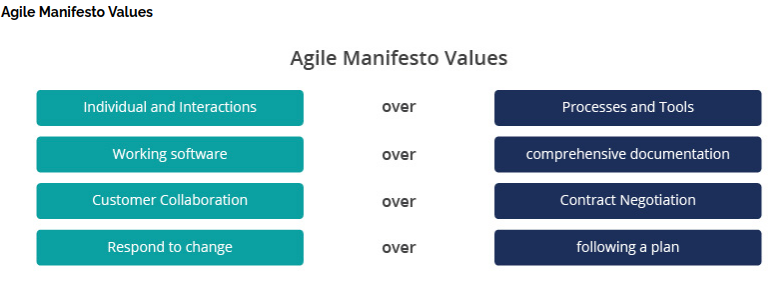
**TEAM ACTIVITY - 3 : AGILE METHODOLOGY**

Firstly, Agile software development, also known as Agile, is an outlook to software development, one that unfolds requirements and solutions through the collaborated effort of self-organising, cross-functional teams and their clients or end users. It recommends planning using adaptive methods along with evolutionary development, empirical knowledge, and continual progress.

1. **What is Agile Manifesto?**

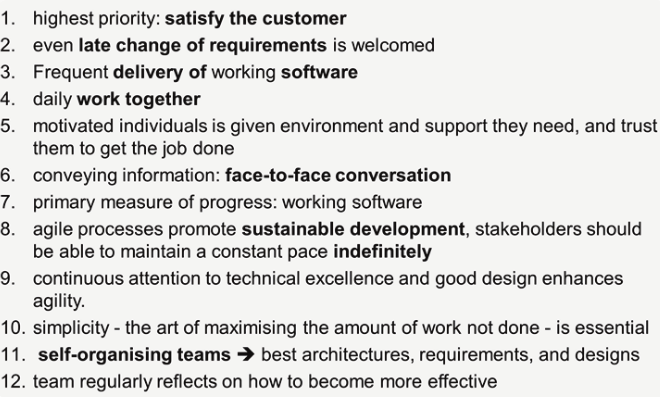
The Manifesto for Agile Software Development, commonly referred to as Agile Manifesto, Is a legal official order that includes twelve principles and four values to show the way for an iterative and people-centric approach to software development. It focuses primarily on testing while keeping the code simple, delivering the functioning bits of the application as soon as they are ready. It promotes an easy, clear and simple approach to developing software in short sprints so that each functioning bit of the software could be analysed and tested based on the client’s or the end user’s requirements, and may be changed if required to meet their needs.Although this set of values and principles were formed primarily for software development, the same can be applied to different forms of business.

This makes Agile a very effective and flexible method for all forms of business.

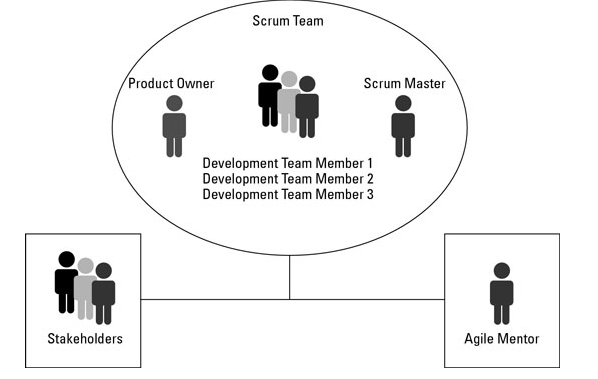


1. **What is Agile Principle?**

The following 12 Principles are based on the Agile Mainfesto.



1. **What is Agile Roles?**



**Scrum Master:**

* The Scrum Master is considered to be the top-dog in every organization because companies usually hire them and don’t treat them as permanent employ that is why they are with no authority.
* It is their duty to remove all the hindrance or obstruction in the way of achieving any goal.
* It is also their role to enforce scrum ceremonies and processes.
* They are the ones who commit to goals and deadlines on behalf of the team.

**Product Owner:**

* The product owner is responsible for conveying the vision of the stakeholders to the team.
* They have the authority to alter the scope.
* The Product Owners are responsible for the return on investment (ROI) that is why they occupy an authoritative position in the firm.
* Because they convey the vision of the stakeholders that is why they are the voice of the stakeholders.
* Not only with the team, but they also communicate with the stakeholders about progress and problems.

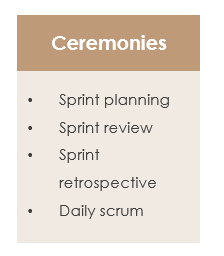
**Scrum Team:**

* The Scrum Team is responsible for all the activities that lead them towards their sprint goals.
* They have to work with the Scrum Master to prioritize the items from the product backlog in the sprint planning.
* Once committed, it is their responsibility to fulfill the commitment and deliver the agreed results on time with great quality.
* The Scrum Master is not responsible for keeping his team organized that is they it is the duty of the Scrum Team to get self-organized.
* They have to be agile in the office and have to attend every stand-up and other ceremonies.
* They have to participate in all the meetings despite their nature and have to ensure that all the findings of the meetings are getting practically addressed in the project.

**Stakeholders:**

* The Stakeholder has to keep a healthy relationship with the Product Owner in order to share every detail regarding his project.
* The Stakeholder is responsible for conveying his wishes and concerns to the product owner or else the product owner would not be responsible for his project quality and time duration.
* The Stakeholder has to provide regular input to queries from the Product Owner.
* Prioritizing the work effectively with the Product Owner is another job that the Stakeholder has to do to ensure his project development.
* Keep taking updates or keep giving updates regarding any change in the plans.

1. **What are Agile Ceremonies?**



1. **Sprint Planning**

Sprint Planning is used to determine what the team will accomplish in the upcoming Sprint. The event itself has two parts. The first half of Sprint Planning is used to determine 'What' the team will be working on, by pulling items from their Backlog into their Sprint Backlog. The second half of Sprint Planning is when the Development Team determines 'How' they will accomplish the work that's been pulled into the Sprint Backlog.

1. **Sprint Review**

The Sprint Review is when the team presents their work from the Sprint to the project's stakeholders. It should cover not only the work they accomplished, but also open discussions around the work they were not able to complete. The attendees of this event should include anyone with a vested interest in the project. Particularly stakeholders, clients, and end-users.

1. **Sprint Retrospective**

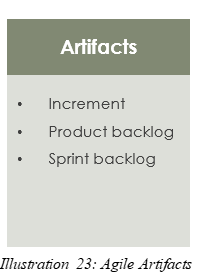
The Sprint Retrospective is the primary event in which the Scrum Team can inspect and adapt their approaches based on their experiences from the previous sprints. Retrospectives can be held using a large variety of games, questions, and exercises; but at it's core, the Sprint Retrospective helps the team to determine: What worked well in the last sprint? What did not work well? And what can be implemented into the next Sprint to improve how the Scrum Team does it's work? Retrospectives allow the team to consistently improve from one Sprint to the next.

1. **Daily Scrum**

The Daily Scrum, sometimes referred to as the Daily Stand-up, has a time-box for 15 minutes or less, and is specifically for the benefit of the development team. The goal of this event is for the team to get in sync on a daily basis, allowing for better collaboration and transparency. The Daily Scrum should be held at the same time each day and should not include anyone outside of the Scrum Team. Traditionally, the Daily Scrum involves each team member answering three questions:

* What did I achieve yesterday to help us meet our Sprint Goal?
* What do I hope to achieve today to help us meet our Sprint Goal?
* Do I see any impediments that prevent me or my team from achieving our Sprint Goal?

**What are Agile Artifacts?**



1. **Increment**

The Increment is the sum of all the Product Backlog items completed during a Sprint and all previous Sprints.

At the end of a Sprint, the new Increment must be “Done,” which means: It must meet the Scrum Team’s Definition of “Done.”

1. **Product Backlog**

A product backlog is a list of all the things that are required in the product and it is a dynamic and best understood requirement for any changes to be made to the product. Product backlog owned by the Product Owner (PO) which consists of a list of all features, functions, requirements, enhancements, and fixes that constitute the changes to be made to the product in the future releases.

1. **Sprint Backlog**

The Sprint Backlog is the set of Product Backlog items selected for the Sprint plus a plan for delivering the product Increment and realizing the Sprint Goal. The Sprint Backlog is a forecast by the Development Team about what functionality will be in the next Increment and the work needed to deliver that functionality. The Sprint Backlog defines the work the Development Team will perform to turn Product Backlog items into a “Done” Increment. The Sprint Backlog makes visible all of the work that the Development Team identifies as to meet the Sprint Goal.

1. **What is Agile Tools?**

The list below shows some of the best tools on offer. For a complete list, [see this post](http://blog.capterra.com/agile-project-management-software/).

* Active Collab

An affordable tool for small businesses, Active Collab is easy to use. This software development aid requires little training and provides excellent support.

* Agilo for Scrum

Stakeholders get updated automatically on the project’s progress with Agile for Scrum. Features sprint reports and burn down charts for better data mining.

* [Atlassian Jira + Agile](https://www.atlassian.com/software/jira?a=capterra)

This powerful project management tool facilitates development by incorporating Scrum, Kanban, and customization workflows.

* [Pivotal Tracker](https://www.pivotaltracker.com/)

This methodology tool is geared specifically for mobile projects. A little jargon-heavy, it’s user-friendly after a brief orientation period.

* [Prefix](https://stackify.com/prefix/)

This free tool from Stackify provides an instant feedback loop to catch and fix bugs before they can deploy.

* [Retrace](https://stackify.com/retrace/)

For a more robust solution complete with monitoring, errors, logs, and more, Stackify’s Retrace provides app performance insights from integration to QA to production, at the code level.

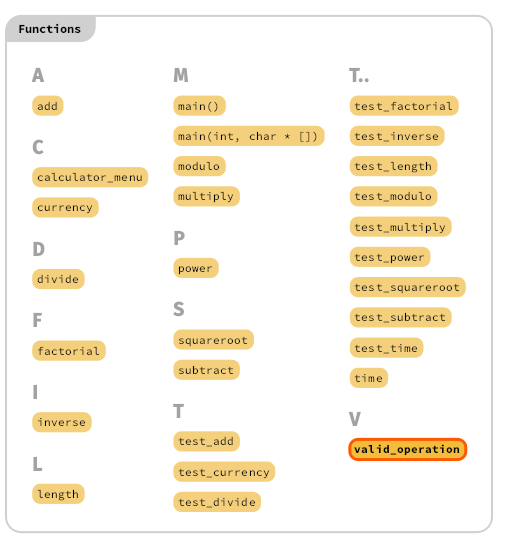
**TEAM ACTIVITY - 4 : SIMPLE CALCULATOR**

**1. Introduction**

A calculator app is one of the most basic yet important apps on the phone. It deals with calculations every day. It has a different set of functionality and features based on the requirement.

**2. Features**

The calculator is based on the following kind of operations which is added in its function module.

**

*Illustration 1: Feature extraction for simple calculator*

**3.Requirements Analysis**

**3.1 High Level Requirements**

|  |  |
| --- | --- |
| **ID** | **Description** |
| HRA-1 | A calculator application that performs calculations, scientific calculations and conversions. |
| HRA-2 | The calculator is developed using standard C language. |
| HRA-3 | * Display – to display operation in console * Add * Subtract * Multiply * Divide * Modulus * Power * Square root * Factorial * Inverse * Currency * Length * Time * Exit -to close the calculator |

## 

## **3.2 Low Level Requirements:**

|  |  |
| --- | --- |
| **ID** | **Description** |
| LRA-1 | Should exit. |
| LRA-2 | Infinite division and zero error specified. |
| LRA-3 | Working with class of two arithmetic operands |
| LRA-4 | Invalid option for menu driven option when user choose invalid options |

# 

# 

# 

# 

# 

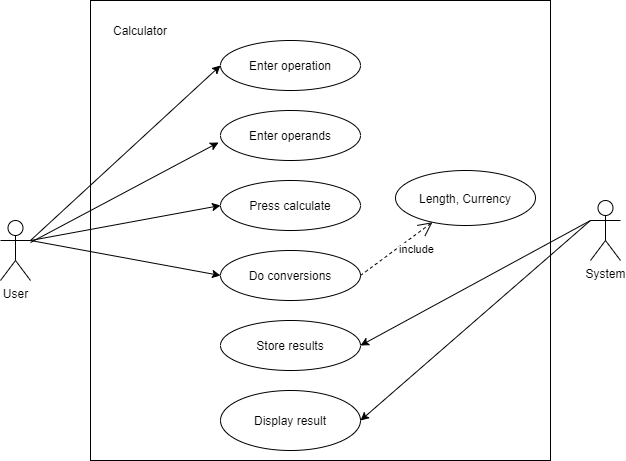
# 

# 

# 

# 4. DESIGN ANALYSIS

## **4.1 Use case diagram for simple calculator**



## 

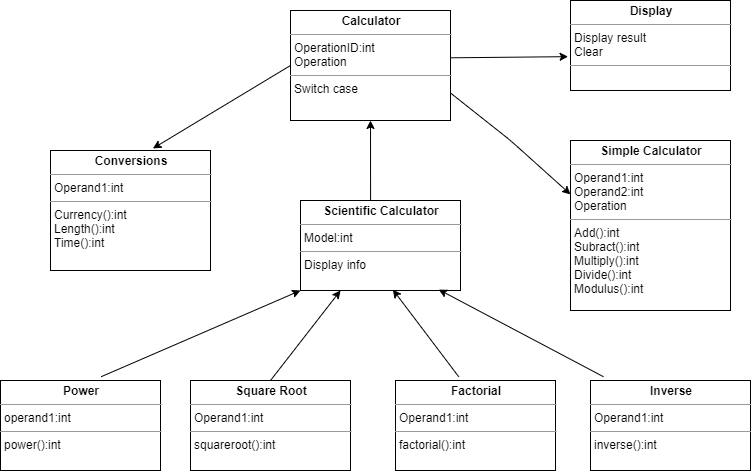
## 

## 

## 

## 

## **4.2 Class diagram for simple calculator**



## **4.3 Activity diagram for simple calculator**

## 

## 

## **4.4 Component diagram for simple calculator**

# 

# 

# 

# 

# 

# 

## **4.5 Deployment diagram for simple calculator**

# 

# 

# 

# 

# 

## **4.6 Sequence diagram for simple calculator**

# 

# 

# 

## **4.7 Package diagram for simple calculator**

# 

# 

# 

# 5. Test Plan:

## **5.1 Requirement based testing**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Description** | **Pre-Condition** | **Expected input** | **Expected output** | **Actual output** |
| T1 | Addition | Integer Input | 4+16 | 20 |  |
| T2 | Subtraction | Integer Input | 61-4 | 57 |  |
| T3 | Multiply | Integer Input | 9\*3 | 27 |  |
| T4 | Division | Integer Input | 8/2 | 4 |  |
| T5 | Square | Integer Input | 4^2 | 16 |  |
| T6 | Square root | Integer Input | 81 | 9 |  |

## **5.2 Boundary condition**

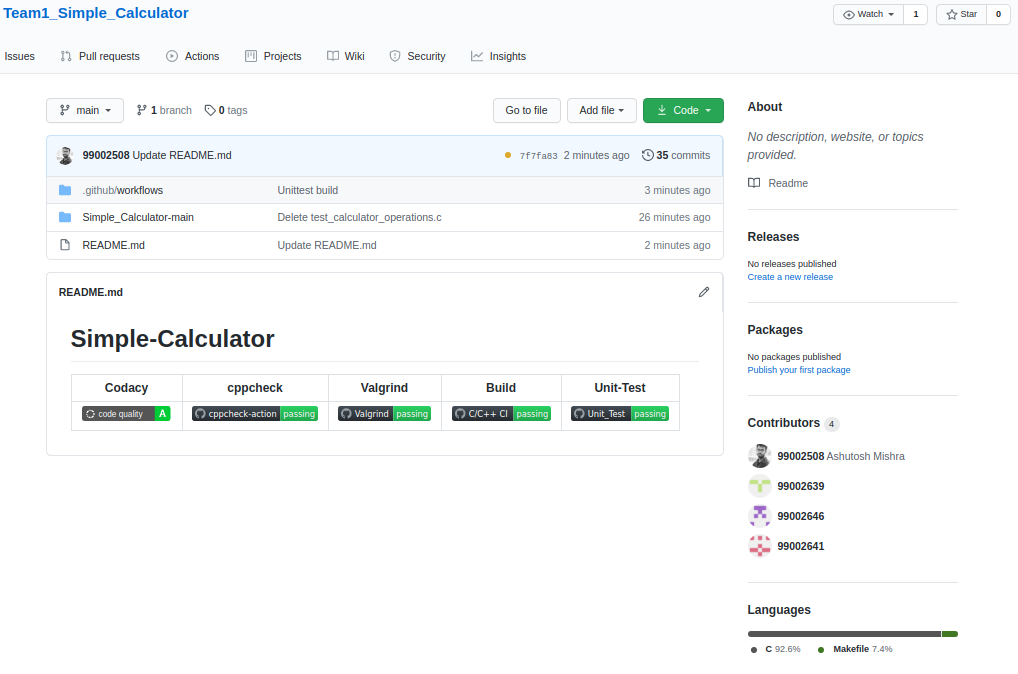
* Long long integer operation
* For imaginary number and complex number as input

## **5.3 Scenario based**

* Alphabet as an input or any special character.
* Undefined attributes added.

## **6. Git Dash Board with CI/CD Framework**

Repolink:<https://github.com/99002508/Team1_Simple_Calculator>



# Team Activity-4 SDLC for Simple Calculator Agile

## **1. Product backlog and theme.**

A calculator app is one of the most basic yet important apps on the phone. It deals with calculations every day. It have different set of functionality and features based on the requirement.

## **2. Epic**

* Simple arithmetic like addition, subtraction, multiplication, division.
* All arithmetic operation using two operand.
* Scientific operations like nth power of a number, square root of a number
* Divide by Zero
* Complex number as operand invalid operation.
* Robust query handling
* Take discount query and handling.

## **3. User Story and sprints**

## **Sprint-1 Divide by zero**

* Invalid operation.
* Console clear and time exit.
* Back to the initial display.
  1. **Sprint-2 Complex number as operand**
* Terminate the operands
* Console clear and time exit
* Back to initial display.
  1. **Sprint-3 Square root of imaginary number**
* Terminate the operands
* Console clear and time exit
* Back to initial display.
  1. **Sprint-4 Long long integer as an operand**
* Shift the operands
* Console clear and time exit
* Back to initial display.
* Clear screen to zero.

**TEAM ACTIVITY - 5 : TOOLS**

**1) UML DIAGRAMS:**

|  |  |  |
| --- | --- | --- |
| SL .NO | TOOL NAME | DESCRIPTION |
| 1 | ENTERPRISE  ARCHITECT | * Enterprise Architect covers the core aspects of the application development life-cycle, from requirments managment through to design, construction, testing and maintenance phases, with support for traceability,project managment and [change control](https://en.wikipedia.org/wiki/Enterprise_Architect_(software)#Change_management) of these processes * Facilities for model driven development of application code using an [internal integrated-development platform](https://en.wikipedia.org/wiki/Enterprise_Architect_(software)#System_development). |
| 2 | VISUAL  PARADIGM | * Visual Paradigm (VP-UML) is a UML CASE Tool supporting UML 2, SysML and Business Process Modeling Notation (BPMN) from the Object Management Group (OMG). In addition to modeling support, it provides report generation and code engineering capabilities including code generation. * Visual Paradigm supports requirements management including user stories, use cases, SysML requirement diagrams and textual analysis. |
| 3 | MS VISIO | * In Visio we can start with a blank UML template or (in some cases) modify a UML starter diagram. Microsoft made Visio 2013 for Windows available in two editions: Standard and Professional. * Visio 2010 added support for the VDX file format, which is a well-documented XML Schema-based ("DatadiagramML"). * Visio also supports saving files in SVG files, other diagramming files and images. However, images cannot be opened. |
| 4 | MAGIC DRAW | * MagicDraw is a visual UML,SYSML,BPMN, and UPDM modeling tool with team collaboration support. Designed for business analysts, software analysts, programmers, and QA engineers, this dynamic and versatile development tool facilitates analysis and design of Object oriented(OO) systems and databases. * It provides the code engineering mechanism (with full round-trip support for J2EE, C#, C++, CORBA IDL programming languages, .NET, XML Schema, WSDL), as well as database schema modeling, DDL generation and reverse engineering facilities. |
| 5 | IBM DOORS | * IBM Engineering Requirements Management DOORS Next (formerly RDNG) is a requirements management tool that provides a smarter way to define, trace, analyze, and manage requirements. Use Engineering Requirements Management DOORS Next to optimize communication and collaboration, allowing your teams to increase quality and work more |
| 6 | CREATELY | * **Creately** is diagramming and design software operated by Cinergix, Pty Ltd.It is acloud-based diagram tool built on Adobe's flex/flash technologies and provides a visual communication platform for virtual teams * It can be used to create info-graphics, flowcharts, Gantt charts, organisational charts, website wireframes, UML designs, mind maps, circuit board designs, doodle art and many other diagram types. |

**2)TEST PLAN:**

|  |  |  |
| --- | --- | --- |
| SL .NO | TOOL NAME | DESCRIPTION |
| 1 | LDRA | * The LDRA tool suite helps you build quality into your software development life-cycle. * Many users of the LDRA tool suite are required to certify their software. * The LDRA tool suite’s open and extensible platform is unique in its integration of software life-cycle traceability, static and dynamic analysis, unit test and system-level testing on virtually any host or target platform. |
| 2 | PARASOFT | * It is an [independent software](https://en.wikipedia.org/wiki/Independent_software_vendor) vendor specializing in [automated softwa](https://en.wikipedia.org/wiki/Automated_software_testing)re testing and [application security](https://en.wikipedia.org/wiki/Application_security) * Parasoft also develops Memory Detection technology that finds run-time errors in C and C++ programs. * For service visualization, Parasoft technologies are used to automatically capture and emulate dependent system behavior of mainframes, third-party components, or any system component that is unavailable or difficult to access for development and testing purposes |

**REFERENCES**

[1]. The below link is used for drawing the UML diagrams

<https://app.diagrams.net>

[2] <https://lucid.app>.

[3] <https://www.atlassian.com/agile>

[4] <https://www.youtube.com/watch?v=WjwEh15M5Rw>

[5] <https://www.youtube.com/watch?v=oTZd2vo3FQU&t=337s>