./

Learning Report – Applied System Development Life Cycle and Software Testing

* Sampreeth Rayadurga

99002500(PS no)



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

**Document History**

Table of Contents

[Table of Figures](#_Toc53213903) 4

[ACTIVITY 1: BEVERAGE VENDING MACHIne(V MODEL)](#_Toc53213905) 4

[**INTRODUCTION**](#_Toc53213906) 4

[Formal Definition:](#_Toc53213907) 4

[My product “Beverage Vending Machine”](#_Toc53213908) 4

[SWOT Analysis of the product:](#_Toc53213909) 5

[Requirements and Research](#_Toc53213910) 6

[Ageing of the product](#_Toc53213911) 6

[Cost of the product](#_Toc53213912) 6

[High level requirements](#_Toc53213913) 6

[Low level requirements](#_Toc53213914) 7

[**DESIGN OF THE SYSTEM**](#_Toc53213915) 8

[**TEST PLAN**](#_Toc53213916) 12

[Requirement based test plan:](#_Toc53213917) 12

[Scenario based test plan 1](#_Toc53213918)3

[Boundary based test plan 1](#_Toc53213919)3

[REFERENCES 1](#_Toc53213920)4

**GROUP ACTIVITIES**

*Banking System Recalls.............................................................................................................................................15*

*Difference between UML and SysML .............................................................................................................................................16*

*References........................................................................................................................................17*

**ACTIVITY 2: BEVERAGE VENDING MACHINE(AGILE MODEL)..........................................................**18

Theme.............................................................................................................................................................18

Epic................................................................................................................................................................19

User Stories....................................................................................................................................................19

Scrum..............................................................................................................................................................19

**GROUP ACTIVITY (Agile)**..................................................................................................................

*Manifesto........................................................................................................................................................20*

*Principles...........................................................................................................................................20*

*Roles.................................................................................................................................................................22*

*Ceremonies.......................................................................................................................................................23*

Artifacts............................................................................................................................................................25

Tools................................................................................................................................................................26

References......................................................................................................................................................26

**ACTIVITY 3: Applying Vmodel on Project ................................................................................27**

**INTRODUCTION**

Introduction*...............................................................................................................................................27*

*High level requirements.................................................................................................................28*

*Low level requirements..................................................................................................................28*

[**DESIGN OF THE SYSTEM**](#_Toc53213915) 29

[**TEST PLAN**](#_Toc53213916) 31

[Requirement based test plan:](#_Toc53213917) 32

[Scenario based test plan](#_Toc53213918) 33

[Boundary based test plan](#_Toc53213919) 33

*REFERENCES 34*

**ACTIVITY (TEAM) 4: Applying V Model for Simple Calculator.............................................35**

[Formal Definition:](#_Toc53213907) 35

[My product “Beverage Vending Machine”](#_Toc53213908) 36

[SWOT Analysis of the product:](#_Toc53213909) 36

[Requirements and Research](#_Toc53213910) 37

[Ageing of the product](#_Toc53213911) 38

[Cost of the product](#_Toc53213912) 38

[High level requirements](#_Toc53213913) 39

[Low level requirements](#_Toc53213914) 40

[**DESIGN OF THE SYSTEM**](#_Toc53213915) 41

[**TEST PLAN**](#_Toc53213916) 42

[Requirement based test plan:](#_Toc53213917) 43

[Scenario based test plan](#_Toc53213918) 44

[Boundary based test plan](#_Toc53213919) 44

[REFERENCES](#_Toc53213920) 44

**ACTIVITY (TEAM) 5: Applying Agile for Simple Calculator.............................................45**

Theme.............................................................................................................................................................45

Epic................................................................................................................................................................46

User Stories....................................................................................................................................................46

Scrum..............................................................................................................................................................47

**ACTIVITY (TEAM) 6: Tools Used.............................................48**

# ACTIVITY 1: BEVERAGE VENDING MACHINE (V MODEL)

## **INTRODUCTION**

Formal Definition: Beverage Vending machine is a vending machine which dispenses hot coffee,milk,hot water and other coffee beverages. Machine was invented in United States by Rudd-Melikan Company in 1947 debuting as the “Kwik Kafe”.

My product “Beverage Vending Machine**”:** Beverage Vending Machine dispenses the required beverage to the user on the click of button. User can choose from the available beverages displayed in the machine and then click on the button to dispense required beverage.

Beverage Vending Machines are convenient allowing users to perform quick self service and get beverage instantly just with a button click. There is also steam option in beverage vending machine so that beverage can be heated to required temperature and it can be used to froth milk which is must for creating latte for espresso beverage surface.

Beverage Vending Machine is very popular piece of furniture in offices, factories or public buildings. It not only serves as dispenser of beverages but also a spot where staff and visitors gather for a quick chat.

Beverage Vending Machine comes in 3 popular sizes namely classic, medium and compact size. Different types of beverage vending machines are single option vending machine, double option vending machine, four option vending machine, six option vending machine, table top beverage vending machine. Some of the manufactures of coffee vending machine are Nescafe, Coffee Day, Lipton and Barista.

SWOT Analysis of the product:

|  |  |  |  |
| --- | --- | --- | --- |
| **Strength** | **Weakness** | **Opportunities** | **Threats** |
| Quick and easy service | Weak brand awareness | Huge market of offices, factories and hospitals. | Consumers may cut back on coffee consumption due to health related risks. |
| Occupies less space | Requires Regular Maintainence | Can make use of IoT to make machine smart. | Strong competition |
| 24/7 Service | Usage of high volt of electricity | Can be used in restaurants to popularize the brand. | Resistance from consumers as it may not replace home brewed coffee or tea. |

Table 1 SWOT Analysis of the product

**Requirements and Research**:

|  |  |
| --- | --- |
| AGEING OF THE PRODUCT | COST OF THE PRODUCT |
| Beverage Vending Machine installation took place in 1947 in United States by Rudd-Mekian Company to dispense coffee in 5 seconds. Machines used liquid coffee concentrate that needs mixing with boiling water. | $ 3000 in the 1970s in US initially when the machine was invented |
| In 1988, bean grinders were added to coffee vending machines which were able to provide choices like espresso and capuccino. | Rs 400000 in the early 2000s |
| In 2009, multifunctional beverage machines were introduced which had touch screen capabilities and multiple options of beverages to choose and some functionalities like steam to produce froth on milk which is required for latte. | Rs 10000-200000 is the current price of the Beverage Vending Machine |

Table 2 Ageing v/s costing of the product

High level requirements:

|  |  |
| --- | --- |
| **ID** | **DESCRIPTION** |
| HL\_01 | Shows quantity of each beverage |
| HL\_02 | Dispense beverage only when cup is placed below filter |
| HL\_03 | Shows quantity of ingredients for beverages |

Table 3 High level requirements of Beverage Vending Machine

Low level requirements:

|  |  |
| --- | --- |
| **ID** | **DESCRIPTION** |
| LL\_01\_HL01 | Checking mixing quantity ratio of milk and coffee powder |
| LL\_02\_HL02 | Power Supply |
| LL\_03\_HL03 | Digital Display |

Table 4 Low level requirements of Beverage Vending Machine

## **DESIGN OF THE SYSTEM**

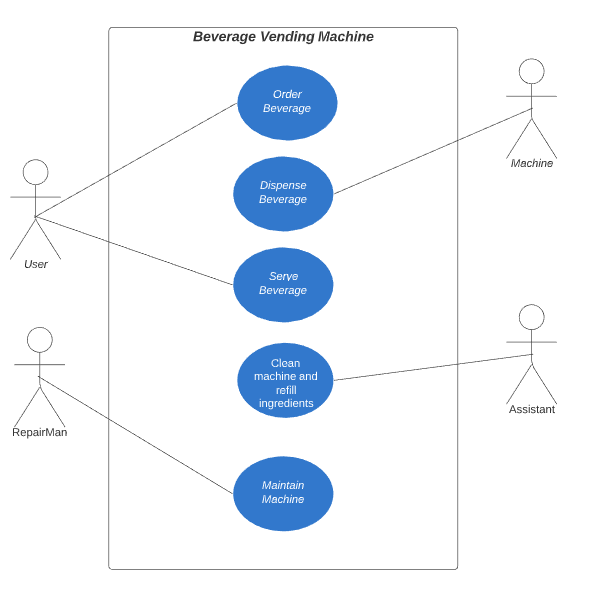


Figure 1 Use case diagram of a Beverage Vending machine

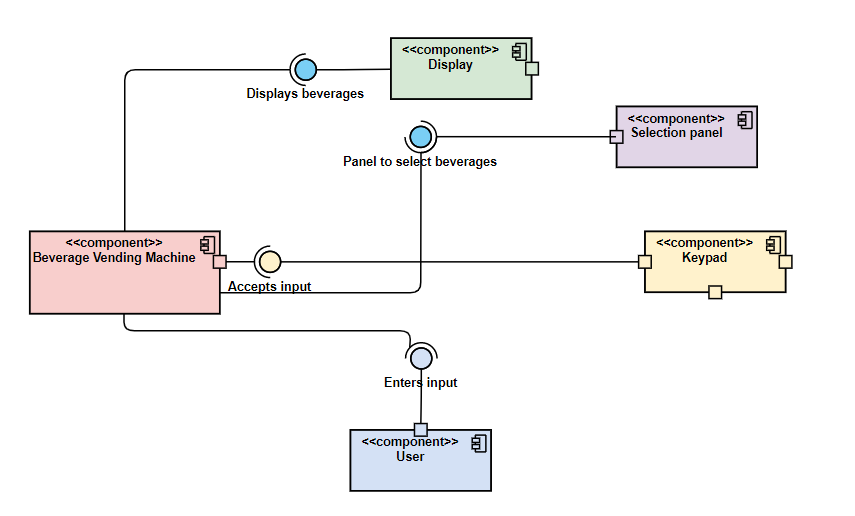


Figure 2 Component diagram describing dispensing of beverage from the Beverage vending Machine

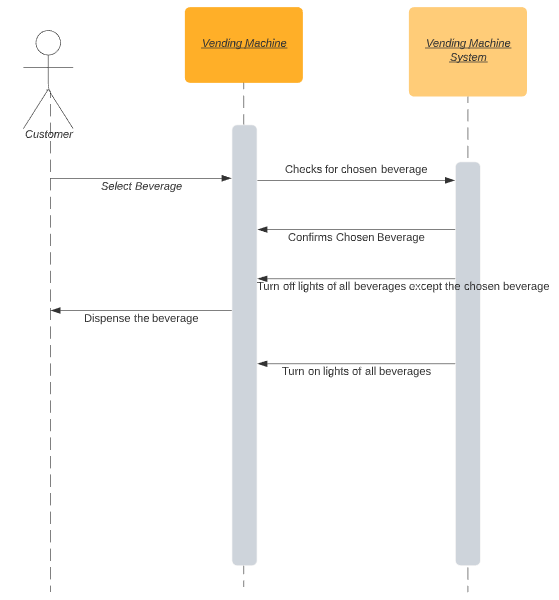


Figure 3 Sequence diagram of Beverage Vending machine

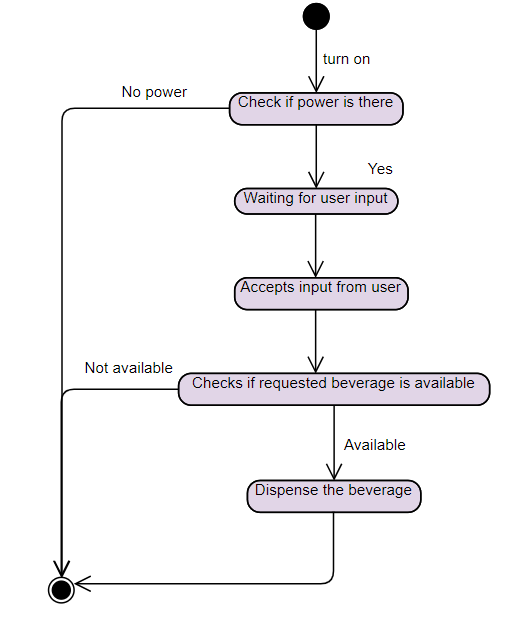


Figure 4 State diagram of a Beverage Vending Machine

## **TEST PLAN**

### Requirement based test plan:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | DESCRIPTION | PRE-CONDITION | EXPECTED INPUT | EXPECTED  OUTPUT | ACTUAL OUTPUT |
| HL\_01 | Show quantity of beverage | Beverage should be present | None | Displays the quantity of beverage | Quantity of each beverage is displayed on screen |
| HL\_02 | Dispense beverage when cup is placed below filter | Cup should be present | Press button after placing cup | Beverage dispensed after cup is placed and button is pressed. | Beverage dispensed when cup is placed below filter after button click. |
| HL\_03 | Displays quantity of ingredients | Ingredients should be present | None | Displays quantity of each ingredient. | Displays quantity of each ingredient. |
| LL\_01\_HL\_01 | Checking mixing quantity of milk and coffee powder | Milk and coffee beans should be present | User clicks on button | Checks mixing quantity and dispenses coffee | Dispenses coffee after checking mixing quantity, |
| LL\_02\_HL\_02 | Power supply | Machine should be connected to power socket | Click on Power on button | Device switches on if power supply is there or it remains switched off | Machine switches on when power supply is there or remains off. |
| LL\_03\_HL\_03 | Digital Display | Display should be working on machine | None | Displays details like time, date, quantity of beverages and ingredient. | Displays correct details. |

Table 5 Test plan of the ATM

**Scenario based test plan**:

1) When a user clicks multiple buttons within 1 second.

2) When a user clicks on button even though beverage is empty.

**Boundary based test plan**:

1) When the user tries to get beverage without glass below filter

2) When a user tries to get beverage more than available quantity

**References**

1. https://en.wikipedia.org/wiki/Coffee\_vending\_machine

2. https://www.slideshare.net/minie747/marketing-ppt-x

3. https://wearedolcegusto.wordpress.com/2012/09/13/swotanalysis/

**GROUP ACTIVITIES**

***BANKING SYSTEM FAILURES AND RECALLS***:

It is believed that banks are safest place to protect our finances, it is not the case always. Some errors in banking systems can have tremendous impact on customers as well as bank which can lead to huge losses and cause inconvenience to customers. Here are some of the banking process failures which have caused doubt in reliability of respected bank organizations:

***1. Technical Faults*** The Uk's Royal Bank of Scotland had updated their software batch CA-7 scheduling process which caused inability to process payments for customers. Customers were charged for late payments and customer in mexican hospital was denied medical suport.It costed bank whooping 175 million euros

***2. TimeZone Differences:*** On 26th June 1974, Hersetatt German bank was seized due to glitch in their software which caused inability to receive money between countries due to timezone differences.

***3. Ethical Failure:*** The Cooperative bank which is a commercial bank described as "a hurricane of negative publicity" in 2013 following the news that there was an alarming shortfall between the bank’s load balance sheet and its actual sale value if ever forced to sell assets.

***4. Global Financial Crisis:*** The financial crash of 2007/2008 is largely considered as the worst banking failure since the Great Depression of the 1930s. The crisis was largely caused as a result of insufficient process aims. Two banks that underwent some of the greatest losses as a result of and within the 2008 global banking crisis included Washington Mutual (WaMu) and IndyMac Bancorp.

***5. Debit card Recalls*** More than 32 lakh debit cards of customers have been blocked or recalled by banks to prevent them from falling prey to any financial fraud after a major security breach at a payment services provider that manages ATM network of a private sector bank. This happened in india in 2016.

***Difference between SysML and UML***

|  |  |
| --- | --- |
| UML | SysML |
| UML is a standardized language for specifying software systems | SysML uses a subset of the diagrams defined by UML and has extensions. It is a UML-profile. |
| UML is software-centric | SysML is more engineering systems-oriented. Used in system level design on SoC |
| Composite structures, which are seldom used in UML. | Composite structures take a central role in SysML as “Blocks”. |
| UML is used to represent software semantics(interpretations of notations). | SysML expresses systems engineering semantics (interpretations of notations) better than UML. |
| Bigger than SysML and difficult to learn | SysML is smaller and easier to learn than UML. |
| UML projects have set of elements, diagrams, and profiles. | SysML has a set of elements,diagrams and profiles along with additional capabilities for requirements management. |

**REFERENCES**

*1. https://www.processexcellencenetwork.com/organizational-change/articles/ top-5-biggest-banking-process-failures-in-modern-h*

*2. https://www.tribuneindia.com/news/archive/business/banks-recall-over-32- lakh-debit-cards-due-to-security-breach-312331*

# ACTIVITY 2- BEVERAGE VENDING MACHINE (AGILE MODEL)

**Theme:** To build Beverage Vending Machine that is efficient and easy to use and operational for 24/7 with less maintenance so that user can have beverages at any point of time just by button click which dispenses beverages instantly and there is steamer which can be used to produce froth on milk or increase temperature of beverage.

**Epic:**

1. To build function to check if beverage is available when requested by user and show error message if it is not available

2. To build function which alerts the operator if quantity of ingredients goes below threshold.

3. To build function which dispenses beverage of choice when user clicks on the button

4. To build function which alerts maintenance team if there is fault in some parts of machine.

**User Stories:** 1. As a user I want the machine to dispense the beverage of my choice on button click *Acceptance Criteria-*  → Beverage should be available when user clicks on button to dispense beverage. → Machine should show error message if beverage is not available 2. As a user I want machine to mix proper proportions of coffee powder and milk *Acceptance Criteria-*  → Seperate pipelines for coffee liquid and milk → Dispenser should calculate appropriate amount of coffee liquid and milk before dispensing it. 3. As a user I want machine to dispense beverage at appropriate temperature. *Acceptance Criteria-*  → Machine should calculate appropriate temperature for beverage and heat the ingredients at particular temperature before dispensing it.

**Scrum:** The requirements gathering process planned for the 1st sprint where in all user requirements are gathered for further processing. The next sprint is planned to create a check beverage functionality wherein all requirements are considered and appropriate functionality is built at end of the sprint. In the further sprints the other functionalities are planned where each of the sprint and correspondingly integrating the functionality with main project.

**GROUP ACTIVITY- AGILE METHODOLOGY**

1. Manifesto: The Agile Manifesto is a brief document built on 4 values and 12 principles for [agile](https://www.productplan.com/agile-product-management/) software development. The Agile Manifesto was published in February 2001 and is the work of 17 software development practitioners who met to discuss on lightweight development methods. The Agile Manifesto outlines a set of 4 values and 12 principles for agile software development. The agile mentality has 4 overarching values differentiating it from traditional software development processes. They are as follows:

a. Individuals and interactions over Processes and tools

b. Working software over Comprehensive documentation

c. Customer collaboration over Contract negotiation

d. Responding to change over following a plan

## Twelve Principles of Agile Manifesto:

*Customer Satisfaction*− Highest priority is given to satisfy the requirements of customers through early and continuous delivery of valuable software.

*Welcome Change*− Changes are inevitable during software development. Ever-changing requirements should be welcome, even late in the development phase. Agile processes should work to increase customers' competitive advantage.

*Deliver a Working Software*− Deliver a working software frequently, ranging from a few weeks to a few months, considering shorter time-scale

*Collaboration−* Business people and developers must work together during the entire life of a project.

*Motivation*− Projects should be built around motivated individuals. Provide an environment to support individual team members and trust them so as to make them feel responsible to get the job done.

*Face-to-face Conversation*− Face-to-face conversation is the most efficient and effective method of conveying information to and within a development team.

*Measure the Progress as per the Working Software*− Working software is the key and it should be the primary measure of progress.

*Maintain Constant Pace*− Agile processes aim towards sustainable development. The business, the developers, and the users should be able to maintain a constant pace with the project.

*Monitoring−* Pay regular attention to technical excellence and good design to enhance agility.

*Simplicity*− Keep things simple and use simple terms to measure the work that is not completed.

*Self-organized Teams*− An agile team should be self-organized and should not depend heavily on other teams because the best architectures, requirements, and designs emerge from self-organized teams.

*Review the Work Regularly*− Review the work done at regular intervals so that the team can reflect on how to become more effective and adjust its behavior accordingly.

**Roles:**  The roles in Scrum are quite different. Clearly defined roles and expectations help individuals perform their tasks efficiently. In Scrum, there are three roles. Together these are known as the Scrum Team.

***1. Product owner***  The product owner represents the stakeholders of the project. The role is primarily responsible for setting the direction for product development or project progress. The Product Owner understands the requirements of the project from a stakeholder perspective and has the necessary soft skills to communicate the requirements to the product development team. Owner need not have technical skills. Key responsibilities of Product Owner include: → Scrum backlog management → Release management → Stakeholder management

***2. Team Lead/ Scrum Master*** The Team Lead or Scrum Master ensures team coordination and supports the progress of the project between individual team members. The Scrum Master takes the instructions from the Product Owner and ensure that the tasks are performed accordingly. The role may involve in facilitating the daily Scrum and Sprint initiatives, communicate between team members regarding the evolving requirements and planning.

Responsibilities may include the following: → Implementing changes → Coordinating between stakeholders to find necessary resources. → Helping product owners to optimize the backlog planning for optimum performance.

***3. Development Team Members*** The team members within the Development Team are comprised of individuals with responsibilities including but not limited to product development. The team takes cross-functional responsibilities necessary to transform an idea or a requirement into a tangible product for the end-users. The key responsibilities of the Development Team is to perform work sprints as per the requirements provided by the Product Owner and coordinated by the Scrum Master. A regular standup meeting called the Daily Scrum is followed to communicate project progress with the peers and the Scrum Master.

***4. Stakeholders*** The Stakeholder position may not be directly involved in the product development process but is used to represent a range of key roles that impact the decisions and work of the Scrum team. The stakeholder may be the end-user of the product, business executives, production support staff, investors, external auditors or Scrum team members from other associated projects and teams. Input from the Stakeholders is key to direct the progress of the project in different directions to align product development with business goals, end-user expectations as well as addressing challenges facing the Scrum Development Team.

**Ceremonies:**

Scrum ceremonies are important elements of the agile software delivery process. They are not just meetings for the sake of having meetings. Rather, these scrum ceremonies provide the framework for teams to get work done in a structured manner, help to set expectations, empower the team to collaborate effectively, and ultimately drive results. Four scrum ceremonies are sprint planning, daily scrum, sprint review and sprint retrospective.

→ *Sprint Planning:* Sprint Planning is the scrum ceremony designed to make sure the team is prepared to get the right things done every sprint. This scrum meeting happens at the beginning of a new sprint and is designed for the Product Owner and Development Team to meet and review the prioritized Product Backlog. Through a series of discussions and negotiations, the team should ultimately create a sprint backlog that contains all items they are committing to complete at the end of the sprint.

*→ Daily scrum:* The Daily Scrum is the team’s chance to get together, define a plan for the day’s work, and identify any blockers. This scrum ceremony provides a frequent opportunity for the team to get together and communicate individual progress toward the sprint goal. It's not a status update. Instead, it should illuminate any impediments the team is having. The Scrum Master is responsible for clearing these roadblocks for the Development Team so they can focus on delivering the work identified in Sprint Planning. Attended by Scrum master and development team and it lasts no longer than 15 minutes.

→ *Sprint Review:* The Sprint Review is the scrum ceremony where all work completed during the sprint can be showcased the stakeholders. At the conclusion of each sprint, the Sprint Review provides a platform for the Development Team to showcase all of the work that has been completed. This allows stakeholders to see things sooner than later and inspect or adapt the product as it emerges. Attended by the scrum team – product owner, development team & scrum master – and typically a mixture of management, outside stakeholders, customers, and even developers from other projects. Lasts for One hour per week of the sprint.

→ *Sprint Retrospective:* The [Sprint Retrospective](https://thedigitalprojectmanager.com/how-run-sprint-retrospective/) is the final scrum ceremony in the sequence that allows the team to look back on the work that was just completed and identify items that could be improved. After a Sprint Review has been conducted, the scrum team needs to have the opportunity to reflect on the work that was just showcased and discuss ways in which to improve, Some common questions asked are:

* What went well over the last sprint?
* What didn’t go so well?
* What could we do differently to improve?

Attended by The Scrum Master and the Development Team. The Product Owner is an optional attendee. Typically, retrospectives should last no more than 1.5 hours for a two-week sprint.

**Artifacts**

An agile scrum has three tangible deliverable, called artifacts.

***1. Product Backlog:***

The Product Backlog is an ordered list of everything that is known to be needed in a product.

***2. Sprint Backlog:***

* The Sprint Backlog is a list of everything that the team commits to achieve in a given Sprint. Once created, no one can add to the Sprint Backlog except the Development Team.
* If the Development Team needs to drop an item from the Sprint Backlog, they must negotiate it with the Product Owner. During this negotiation, the Scrum Master should work with the Development Team and Product Owner to try to find ways to create some smaller increment of an item rather than drop it altogether.

***3. Potentially Releasable Product Increment***

At the end of every Sprint, the team must complete a product increment that is potentially releasable, meaning that meets their agreed-upon definition of done. (An example might be fully tested and fully approved.)

**Tools**

* ***Zephyr:*** Zephyr is able to take care both automated cases( through java selenium) and manual case by integrate with Jira, initially its little bit difficult to get how the system works and the doc is not 100% up to date and sync, however the customer service is very helpful and prompt response and they can set up live meeting to timely solve customer issue!
* ***Backlog:*** Backlog's simple yet powerful interface can be quickly adopted by anyone. Work with developers, clients, designers, and other teams on one connected platform With Backlog, you can keep bug and issue tracking under one roof. Developers can easily collaborate on and release code, tracking each step via pull-requests right in issues. Git and Subversion repositories keep teams connected through it all.
* ***JIRA:*** The basic use of this tool is to track issue and bugs related to your software and Mobile apps. It is also used for project management. This software is used for bug tracking, issue tracking, and project management.
* ***Soap UI:***SoapUI is an agile testing tool for service-oriented architectures (SOA) and REST. Its functionality includes web service inspection, invoking, development, functional testing, and load testing.
* ***Jmeter***

**References**

***Agile principles were defined in the source websites like:***

https://thedigitalprojectmanager.com/scrum-ceremonies-made-simple/

<https://www.tutorialspoint.com/agile/index.htm>

**The documents provided in the Yammer**

**[**[**https://www.yammer.com/lnttsgroup.onmicrosoft.com/threads/902387543801856**](https://www.yammer.com/lnttsgroup.onmicrosoft.com/threads/902387543801856)**]**

# ACTIVITY 3: Applying V MODEL on Project

**Introduction:**

A calculator is a mobile app that performs arithmetic operations on numbers. The simplest calculators can do only addition, subtraction, multiplication, and division. More sophisticated calculators can handle exponential operations, roots, trigonometric functions, and hyperbolic functions. Internally, some calculators actually perform all of these functions by repeated processes of addition. Portable, battery-powered calculators are popular with engineers and engineering students. The calculator we have designed will have,

 Simple Calculations like addition, subtraction, multiplication, division and modulo division.

 Scientific Operations like nth power of a number, square root of a given number, factorial of a number and multiplicative inverse of a number.

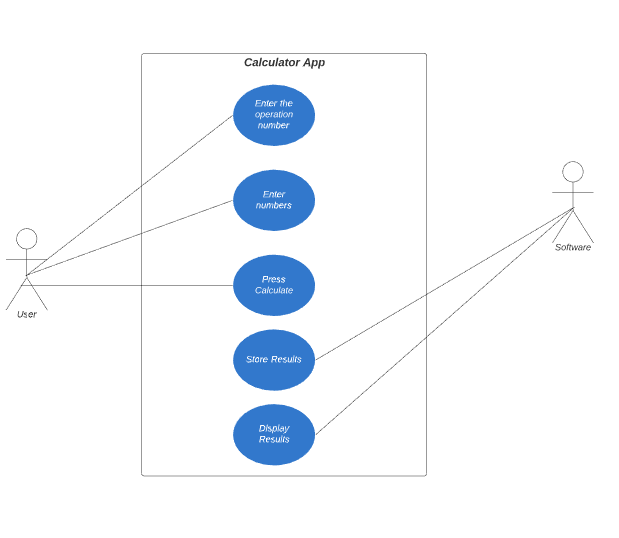
High Level Requirements:

|  |  |
| --- | --- |
| ID | Description |
| HL\_01 | Calculator should perform basic calculations and perform some scientific calculations. |
| HL\_02 | Developed using c programming.Should run on machines supporting gcc compiler |
| HL\_03 | Should display menu like 1. Add 2. Sub 3. Multiply 4. Divide 5. Factorial 6. Square Root 7. Exit 8. Start |
| HL\_04 | Should support both potrait and landscape |
| HL\_05 | Should support all kinds of devices ranging from tablets, large screen phones to small screen phones. |

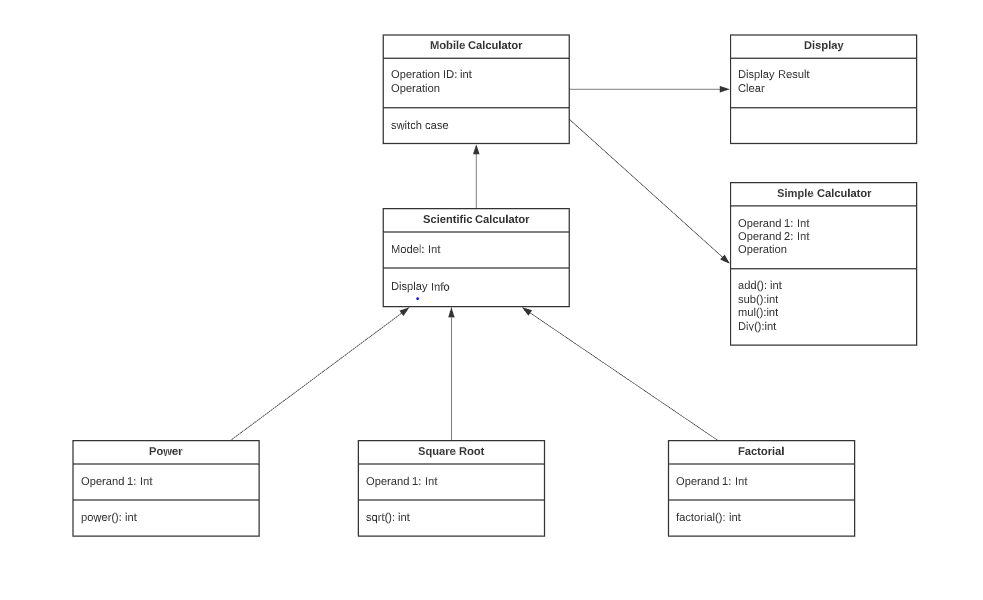
Low Level Requirements:

|  |  |
| --- | --- |
| ID | Description |
| LL\_01 | Should exit when 7 is entered |
| LL\_02 | Not allow Divide by Zero Errors |
| LL\_03 | Not allow user to select same operator consecutively. |
| LL\_04 | Should show error when user types of negative number while choosing operation |

**System Design**



**Figure 5: Use Case Diagram for Project**

**Figure 6 : Class diagram for Project**

**Requirement based Test Plan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Description | Pre-Condition | Expected input | Expected output | Actual output |
| T\_01 | Add 2 numbers | Numbers must be integers | 5+98 | 103 |  |
| T\_02 | Subtract 2 numbers | Numbers must be integers | 34-23 | 11 |  |
| T\_03 | Add 2 Numbers | Numbers must be integers | 5.34+2.34 | Error: Output is in double |  |
| T\_04 | Subtract 2 numbers | Numbers must be integers | 45-98 | Error: Output is negative |  |
| T\_05 | Multiply 2 numbers | Numbers must be integers | 5\*7 | 35 |  |
| T\_06 | Divide 2 numbers | Numbers must be integers | 50/10 | 5 |  |
| T\_07 | Multiply 2 numbers | Numbers must be integers | 2.45\*6.45 | Error: Output is in double |  |
| T\_08 | Divide 2 numbers | Number must be integers | 3/0 | Error: Divide by zero error |  |

**Boundary condition testing:**

1) When the number is too large to perform an operation

2) When a negative number is given as an input to find the square root

3) When there are more than 20 decimals after number

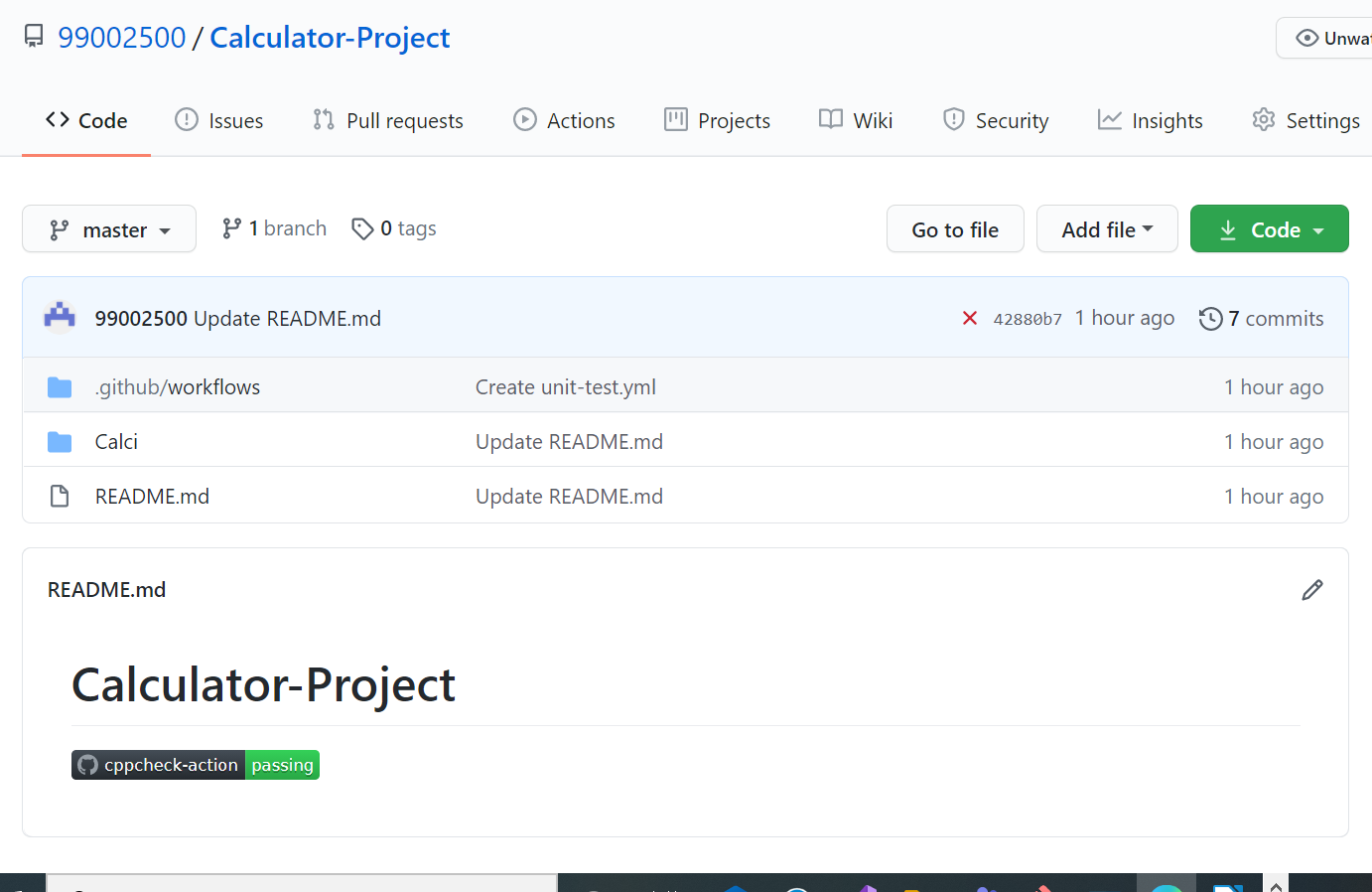
**Scenario based testing:**

1) When the user enters a character value instead of a number

2) When the user enters an operand which is undefined

3) When user tries to do arithmetic operations on characters

**References(CI/CD)**

**Link:** **https://github.com/99002500/Calculator-Project**

# ACTIVITY 4 (TEAM): SIMPLE CALCULATOR (V MODEL)

## **INTRODUCTION**

Formal Definition: Simple calculator is a device which performs only basic calculations based on the operands entered and operator. Simple calculator can do operations only on numbers. Operations which can be performed are: Addition, subtraction, multiplication and division.

**SWOT Analysis of product**

|  |  |  |  |
| --- | --- | --- | --- |
| **Strengths** | **Weakness** | **Opportunities** | **Threats** |
| Portable and user friendly | Heavy competition | Can target small shopkeepers and schools to sell simple calculators | Lot of competitors |
| Long lasting | Used by less number of people | Advertise the brand through social media to get more customers. | Lack of demand |
| Chargeable batteries which charges using solar cells | Less sales as calculators come built in smartphones. | Improve design, enhance battery capacity and durability. | Rapid technology growth |

**Requirements and Research:**

**Aging and Cost Gradation**

-> In 1623, Willhelm Shickard invented a calculating clock which is composed of a multiplying device, mechanism for recording intermediate results and 6 digit decimal adding device.

-> In 1773, Philips Matthas Hahn developed first functional calculator based on machine developed by Gottfried Wilhelm Leibniz in 1672.This calculator featured set of twelve drums in a circular arrangement that could be activated by crank located in the axis of the drums.

-> In 1820, Charles Xavier Thomas DE Colmar developed an arithmometer which was able to perform 4 basic operations namely addition, subtraction, multiplication and division.

It featured a second result display for subtraction as well as multiplication. It was based on stepped drum mechanism of Lebiniz

-> In 1954, IBM developed IBM 608 which was the First All Transistor calculator. It used solid- state transistors instead of vacuum tubes. Machine main memory could store 40 nine digit numbers and perform 4500 additions per second. It cost 83210 US dollars.

-> In 1961, Bell Punch developed a device named ANITA  which was vacuum tube based calculator. It featured 170 cold cathode vacuum tubes, a Dekatron decade counter tube and Numicator display/indicator tubes.

-> In 1967, Texas Instruments developed first hand held calculator named CALTECH. It weighed 45 ounces and had keyboard with 18 keys and small output screen to display visual output upto 12 decimal digits

-> In 1971, BUSICOM developed First truly pocket sized electronic calculator and used LED display. It used Calculator on chip integrated circuit. Cost was $395.

->In 1974, Hewlett Packard(HP) developed first handheld programmable calculator named HP-65 which allowed users to write programs upto 100 lines and record them on blank cards. It had 35 keys controlling around 80 operations. Cost was $795

-> In 2003, Sharp developed the first graphing calculator with touch functionality and some advanced features.

-> Currently most of the smartphones,PC and laptops have calculator applications built in with GUI which can perform operations ranging from simple calculations till advanced calculations. Example is windows calculator.

**Defining system**

-> Allow users to perform only one task at a time.

-> Display result of the operation when ‘=’ button is clicked.

-> Switch on the calculator when pressed on the ‘ON’ button.

-> Support basic operations:Addition, subtraction, multiplication and division

-> Switch off calculator when pressed on the ‘OFF’ button.

-> Supports tasks which have multiple operators and operands.

-> Should allow the user to enter the operator only after an operand.

-> Should not allow users to enter 2 operators next to each other.

**Requirements**

**High Level Requirements**

|  |  |
| --- | --- |
| ID | Description |
| HL\_01 | Simple calculator should perform 4 basic operations: Addition, subtraction, multiplication and division |
| HL\_02 | Display results quickly |
| HL\_03 | Allow upto 2 operands |
| HL\_04 | Sufficient memory to store the result of operation |

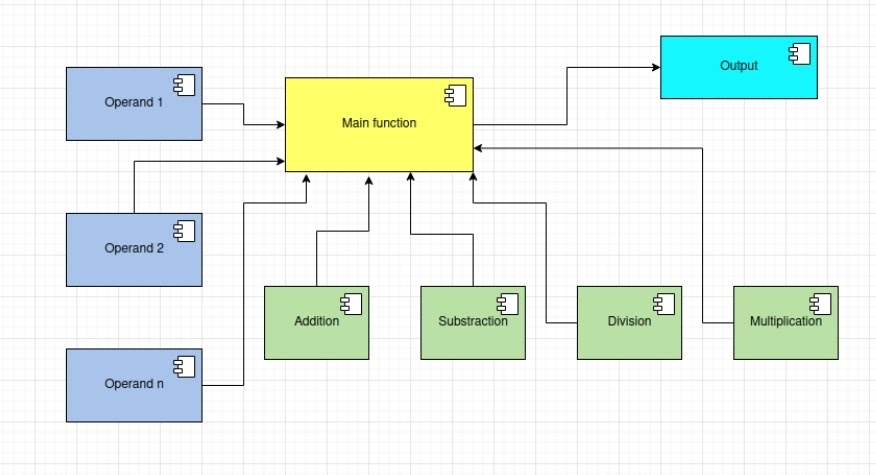
**Low Level Requirements**

|  |  |
| --- | --- |
| ID | Description |
| LL\_01 | Display output only after taking inputs |
| LL\_02 | Divide by zero should show error |
| LL\_03 | Perform addition operation on the operands. |
| LL\_04 | Perform subtraction operation on the operands. |
| LL\_05 | Perform multiplication operation on the operands. |
| LL\_06 | Perform division operation on the operands. |
| LL\_07 | Switch on when pressed  ‘ON’ button and switch off when pressed  ‘OFF’ button. |

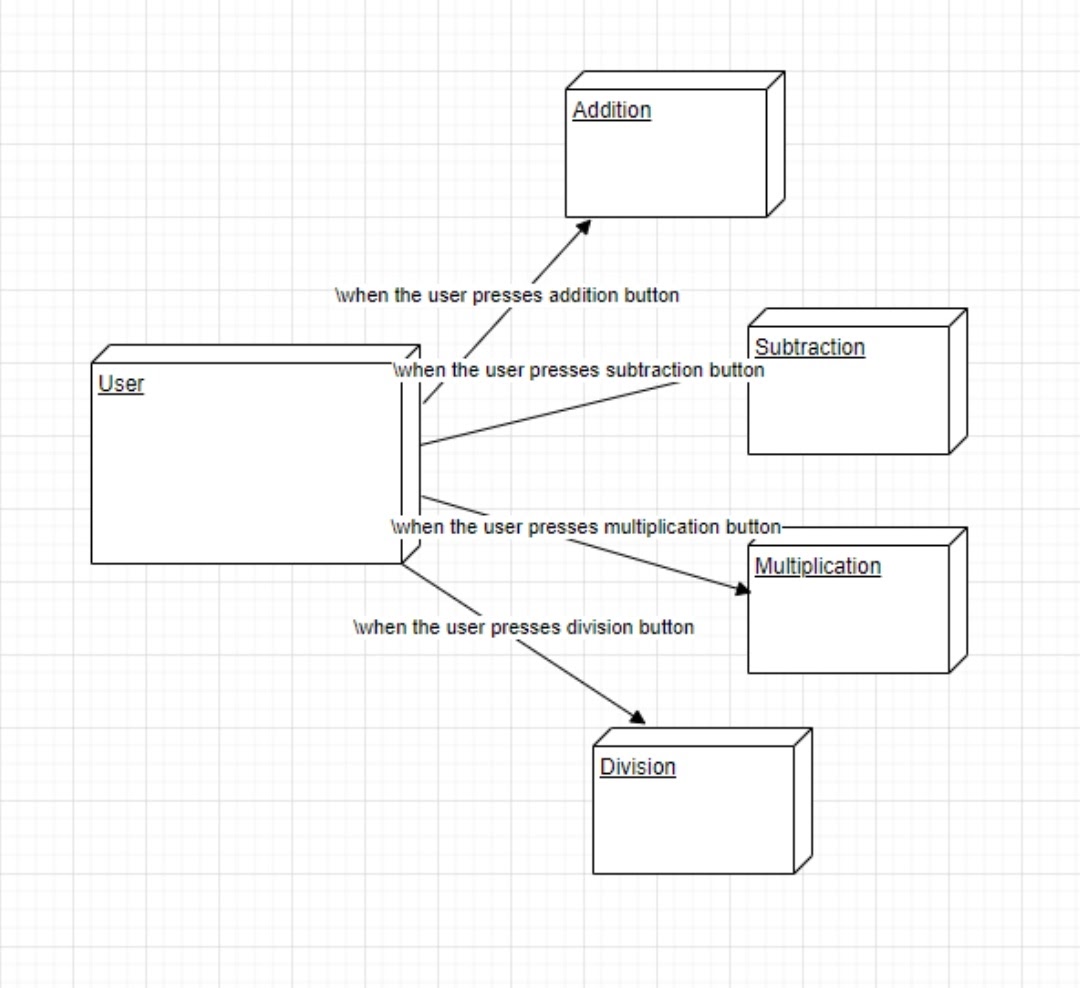
**Design Plan**

**Structural Diagrams**

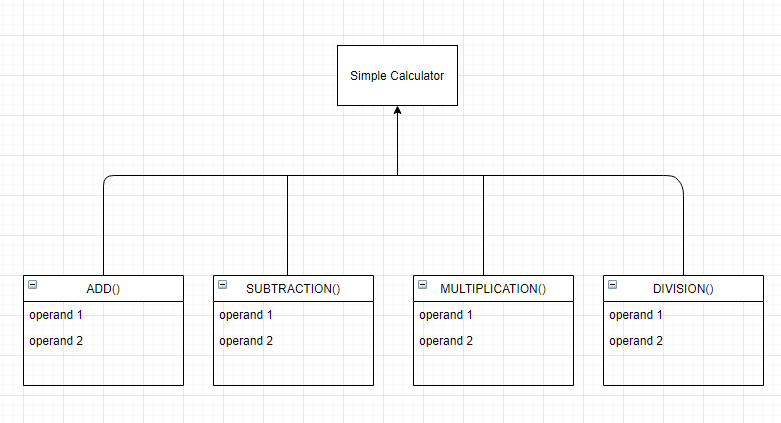
1. Component Diagram



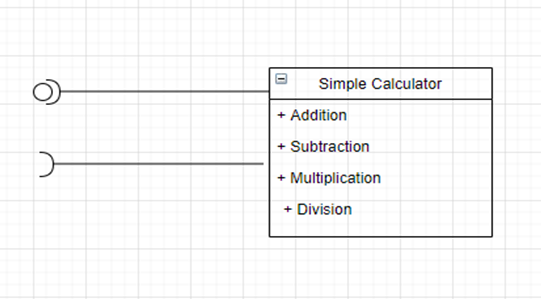
2. Deployment Diagram



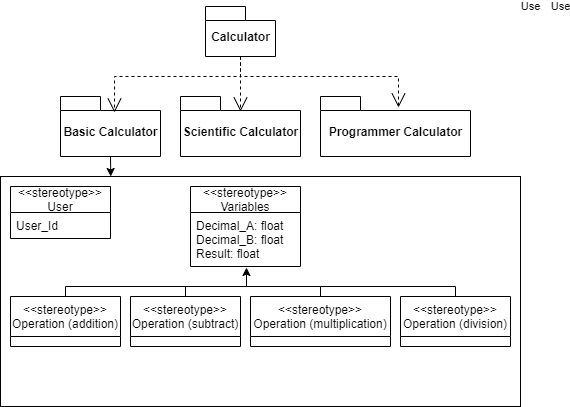
3. Class Diagrams



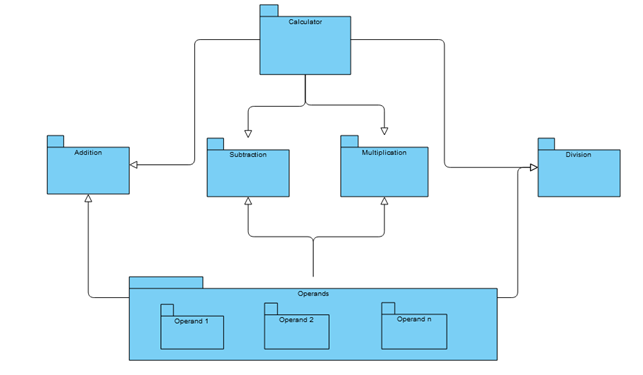
4. Composite Diagram



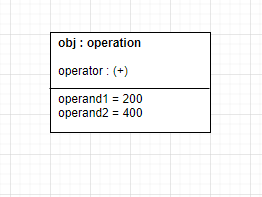
5. Profile Diagram



6. Package Diagram

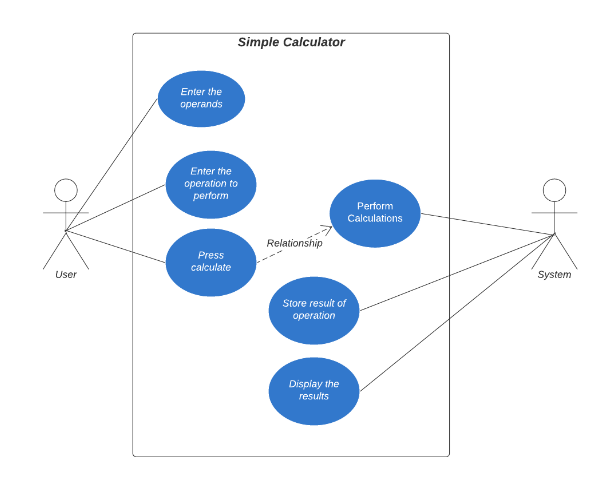


7. Object Diagrams

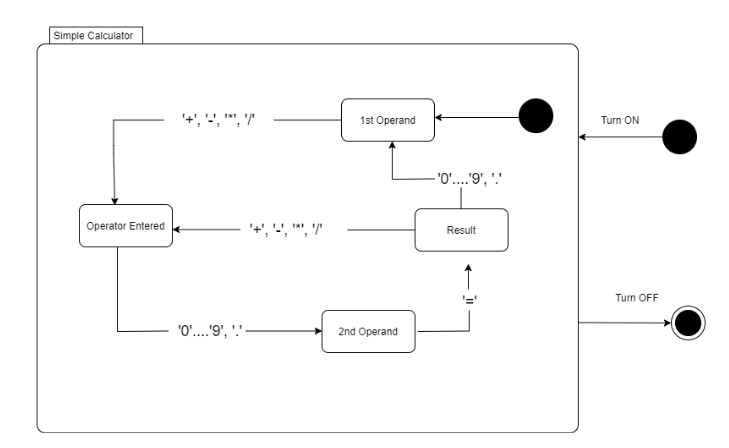


**Behavioral Diagrams**

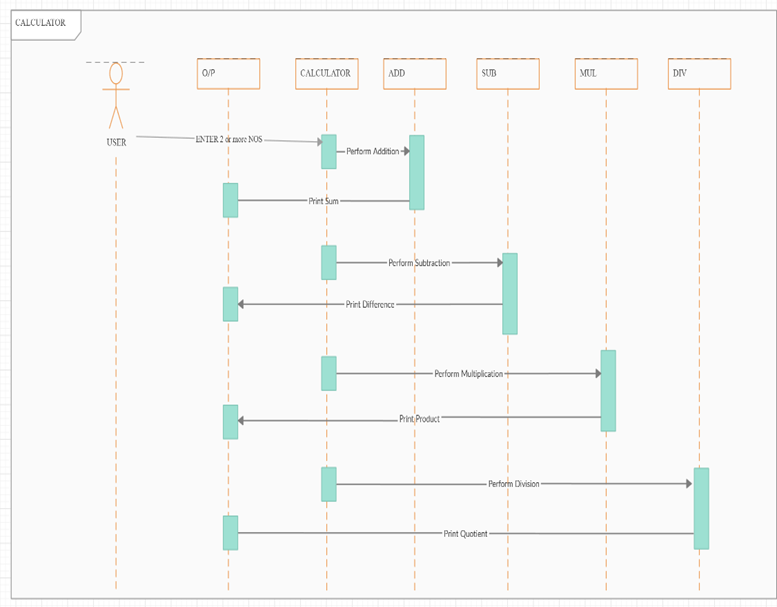
1. Use Case Diagram



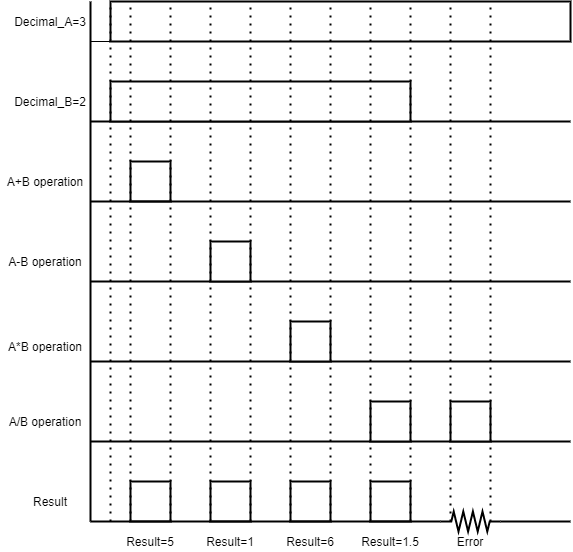
2. State Diagram



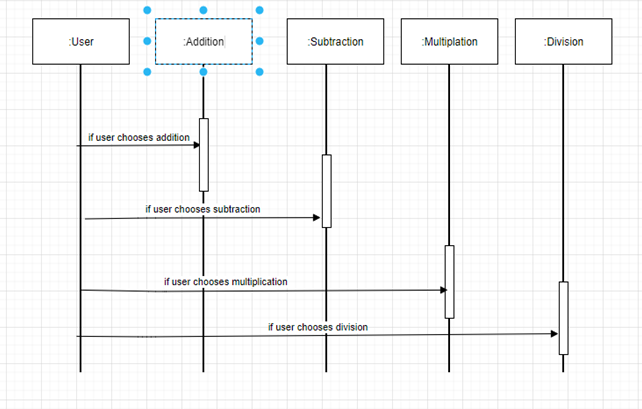
3. Sequence Diagram



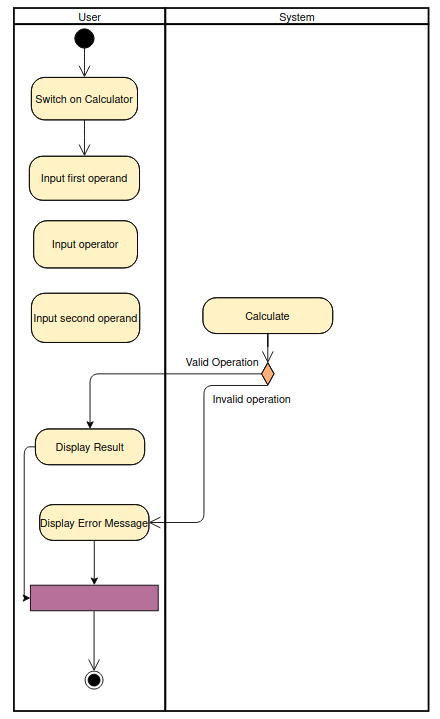
4. Timing Diagrams



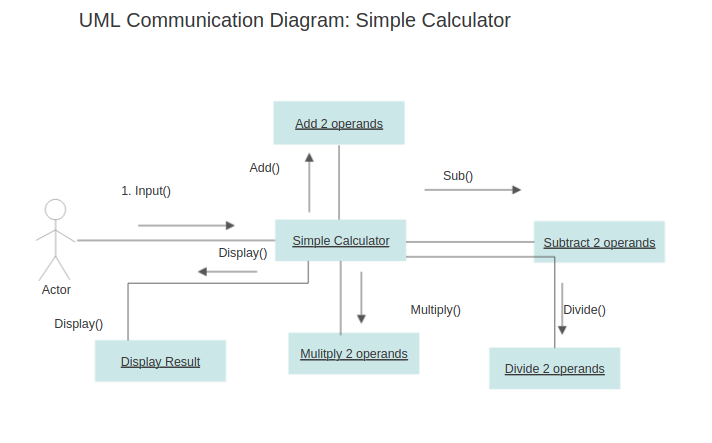
5. Interaction Overview Diagrams



6. Activity Diagrams



7. Communication Diagrams



**Test Plan**

**Unit Testing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Description | Expected Input | Expected Output | Actual Output |
| TC\_LL\_1\_1 | Output should be displayed after inputs are entered | Provide two operands and the type of operation.  ex - 2+5 | 7 |  |
| TC\_LL\_2\_2 | Error should be displayed when division is done by zero. | Give denominator as zero | ERROR |  |
| TC\_LL\_3\_3 | To verify the output of addition operation | 5 + 10 | 15 |  |
| TC\_LL\_4\_4 | To verify the output of subtraction operation. | 10-5  5-10 | 5  -5 |  |
| TC\_LL\_5\_5 | To verify the output of multiplication. | 5\*10 | 50 |  |
| TC\_LL\_6\_6 | To verify the output of division | 5/10 | 0.5 |  |
| TC\_LL\_7\_7 | The screen should display 0 when calculator is turned on | Press On button | 0 |  |
| TC\_LL\_7\_8 | The screen should be blank when calculator is turned off | Press off button | Blank |  |

* **Boundary Condition Testing**

Overflow

1) When addition is performed on two large numbers.

2)When multiplication is performed on two large numbers.

* **Scenario Based Testing**

1) When the user enters a character value instead of a number

2) When the user enters an operand, which is undefined

# **ACTIVITY 4(Team): SIMPLE CALCULATOR (AGILE MODEL)**

**Theme**

To Construct Precise Calculator which includes simple arithmetic operations. User is expected to give at least 2 to 4 input values to perform operations like addition, subtraction, division, and multiplication.

**Epic1**: To Build a function to perform addition operation.

User Stories-1

1. Given Input are 10 and 15.
2. Expected output is 25.

**Epic2:** To Build a function to perform Subtraction operation.

User Stories-2: -

1. Given inputs are 15 and 5.
2. Expected Output is 10.

**Epic3:**To Build a function to perform Multiplication operation.

User Stories-3: -

1. Given inputs are 15 and 5.
2. Expected Output is 75.

**Epic4: -**To Build a function to perform Division operation.

User Stories-4: -

1. Given inputs are 15 and 5.
2. Expected Output is 3.
3. It should be able to handle the exceptions when the number is divided by 0.

**Scrum:** The requirements gathering process planned for the 1st sprint where in all user requirements are gathered for further processing. The next sprint is planned to check the functionality of a calculator wherein all requirements are considered and appropriate functionality is built at end of the sprint. In the further sprints the other functionalities are planned where each of the sprint and correspondingly integrating the functionality with main project.

# **ACTIVITY 5(Team): TOOLS CLASSIFICATION**

Tools for Agile

|  |  |  |
| --- | --- | --- |
| Tools | Description | Link |
| Zephyr | Providing end-to-end solutions for agile teams of all sizes. Get the flexibility, visibility, and insights you need to release better software faster | [https://marketplace.atlassian.com/apps/1014681/zephyr-for-jira-test-management](https://bit.ly/30Nryg8) |
| Backlog | All-in-one project management tool built for developers. Agile Teams use Backlog to work with other teams for enhanced team collaboration and high-quality project delivery. | <https://backlog.com/> |
| JIRA | It is a defect tracking tool which is used for Agile testing as well as project management. This tool is not only used for recording, reporting but also integrated with code development environment. | [https://www.atlassian.com/software/jira/free](https://www.atlassian.com/software/jira/try) |
| Soap UI | Is an agile testing tool for service-oriented architectures (SOA) and REST. Its functionality includes web service inspection, invoking, development, functional testing, and load testing. | <https://www.soapui.org/downloads/download-readyapi-trial/> |
| Jmeter | Application is an open source agile performance testing tool. It is used to load functional test behavior and measure performance of the website. | <http://jmeter.apache.org/download_jmeter.cgi> |

Tools used in V-model:

|  |  |  |
| --- | --- | --- |
| Tools | Description | Link |
| Junit | Used inJava programming language.This tool tests the data first and then inserts in the code. | <https://www.guru99.com/junit-tutorial.html> |
| JMockit | Code coverage tool also has line and path metrics. | <http://jmockit.github.io/index.html> |
| EMMA | Support all coverage types including method, line, basic block. | <http://emma.sourceforge.net/> |
| Squish (Froglogic) | Commercial cross-platform GUI and regression testing tool which is used to test applications based on a variety of GUI technologies | <https://en.wikipedia.org/wiki/Squish_(Froglogic)> |
| Rational Integration tester | It gives a scripting free environment for developing the business process of integration projects. | <https://www.ibm.com/support/knowledgecenter/SSBLQQ_9.2.0/com.ibm.rational.rit.gs.doc/topics/c_ritov_test_methodology.html> |
| Squish | used as the single GUI testing solution for all the applications under test (AUT) | <https://www.froglogic.com/squish/> |
| Usersnap | It’s an easy to use UAT solution that helps QA teams verify if the certain solution works for the user or not. | <https://usersnap.com/blog/types-user-acceptance-tests-frameworks/> |

**References**

***Agile principles were defined in the source websites like:***

https://thedigitalprojectmanager.com/scrum-ceremonies-made-simple/

<https://www.tutorialspoint.com/agile/index.htm>

**The documents provided in the Yammer**

**[**[**https://www.yammer.com/lnttsgroup.onmicrosoft.com/threads/902387543801856**](https://www.yammer.com/lnttsgroup.onmicrosoft.com/threads/902387543801856)**]**