./

Course Code: <CODE>



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **Approved By** | **Remarks/Revision Details** |
| 1 | 17-09-2020 | V Anuj Bharadwaj |  |  | Made changes to the requirements table |
| 2 | 18-09-2020 | V Anuj Bharadwaj |  |  | Added mapping to requirements table |
| 3 | 19-09-2020 | V Anuj Bharadwaj |  |  | Added Learning report for new application. |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Document History**

# 

**REQUIREMENT SHEET**

**POTHOLE DETECTION AND WARNING SYSTEM FOR VEHICLES**

**LIST OF TABLES**

Table 1: Ageing and Cost Gradation…………………………………………………………................... 7

Table 2: SWOT Analysis of the Project………………………………………………………………………. 8

Table 3: Requirements Mapping………………………………………………………………………………… 8

Table 4: Test Plans………………………………………………………………………………………………………9

**LIST OF FIGURES**

Fig 1: Component Diagram for High Level Requirements………………………………………………………. 7

Fig 2: Component Diagram for Low Level Requirements………………………………………………………. 7

Fig 3: Deployment Diagram for High Level Requirements……………………………………………………….8

Fig 4: Deployment Diagram for Low Level Requirements……………………………………………………… 8

Fig 5: Activity Diagram for High Level Requirements……………………………………….……………………. 9

Fig 6: Activity Diagram for Low Level Requirements………………………………………………………………. 10

Fig 7: Sequence Diagram for High Level Requirements………………………………….…………… ………… 11

Fig 8: Sequence Diagram for High Level Requirements……………………………………. …………………… 11

**Purpose of the proposed System**: To detect a pothole or a hump on the road ahead and warn the driver of the vehicle about the same.

**High Level Requirements:**

* A high speed microcontroller capable of processing analog signals from a sensor and convert the same into required form for further process.
* Knowledge on using the Arduino IDE and programming in C/ C++ to write and flash the code into the microcontroller.
* A LiDAR sensor which is the core requirement for the detection process.
* 5V DC Power supply to power the microcontroller and all other peripherals which includes sensors, warning system etc.

**Low Level Requirements:**

* Electronic devices to alert the driver via visual and audio warning which include LED’s, electronic buzzer, vibration motor, 7 segment LCD Display.
* A mount to attach the sensor to the vehicle for proper operation.

**Ageing and Cost Gradation:**

The technologies already present to address the objective at the time of design of the current system and the cost gradation of each system is as follows.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Detection and Inter Vehicular  Communication. (2015) | A bot vehicle is sent to detect the potholes and the geographical location data is shared on a local server which is accessible to the all the vehicles present in the close proximity to the server. | The entire prototype system costed very low including building of a bot vehicle and gps sensor along with a local server. |
|  | Jaguar Land Rover Pothole Detection System. (2015) | Numerous sensors were used to detect and map virtual images of road surfaces and changes in heights of wheel suspension and vehicle movements. | Due to the number of sensors present for the action, the cost of the entire detection system to could sky rocket. |
| 3. | Detection using 2-D LiDAR and camera. (2017) | In this system, a camera was used to capture images of the road and compare with the reference images by using Image Processing and sensor data as a redundant source to back the results obtained from images. The major downside is the vehicle has to pass through the pothole. | The cost here could be a downside because of the hardware present in the system. The speed of processing the data could be affected due to the immense amount and size and the major downside is the risk of vehicle passing through the pothole. |
| 4. | **Pothole Detection and Warning system using LiDAR sensor. (Current Project). (2020)** | **Only a LiDAR sensor was used to detect the potholes and the microcontroller uses the sensor data to warn the driver using the mentioned warning system. The major upside is the pothole can be avoided by detecting it way ahead and within safe distance.** | **The cost of the entire system including sensor, controller, warning system components could be high for a prototype. But the speed of processing, warning to take necessary action could totally prevent the vehicle from entering the pothole.** |
| 5. | **Future Scope of the Project** | **Actuators can be added to the entire system to enable it to automatically take action and safely control the vehicle.** | **Unknown yet.** |

**Table 1: Ageing and Cost Gradation**

**SWOT ANALYSIS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Strengths** | **Weaknesses** | **Opportunities** | **Threats** |
| 1.The vehicle doesn’t have to pass through the pothole to register it.  2.Speed of processing is high due to the minimum amount of data present.  3.A reliable warning system to make sure, the driver is alerted. | 1.Presence of water on the surface, dust and few other environmental constraints could restrict the ability of the sensor to accurately detect the distance. | 1.Actuators can be added in future to the functioning of the entire system. The microcontroller can be used to control the actuators necessarily and automate the process of safely manoeuvring the vehicle without driver intervention. | 1.Vibrations experienced by the vehicle could change the position of the sensor in front of the vehicle, thereby changing the angle of attack of the sensor which could lead to returning false distances to the controller. |

**Table 2: SWOT Analysis of the Project**

**Requirements Mapping:**

|  |  |
| --- | --- |
| Name | Description |
| H\_01 | ATMega 2560 is used as a microcontroller to operate the entire system. |
| H\_02 | A TFMini 2D LiDAR Sensor with UART Com is used for sensing |
| H\_03 | A development environment used to program the microcontroller |
| H\_04 | A 5V DC Power supply from a portable power source |
| H\_01\_L\_01 | LED triggered by the controller to warn the driver. |
| H\_01\_L\_02 | Electronic Buzzer triggered by the controller to warn the driver. |
| H\_01\_L\_03 | 7 segment LCD Display triggered by the controller to warn the driver. |
| H\_01\_L\_04 | Vibration motor triggered by the controller to warn the driver. |
| H\_01\_L\_05 | Used to safely hold the LiDAR sensor to the vehicle. |

**Table 3: Requirements Mapping**

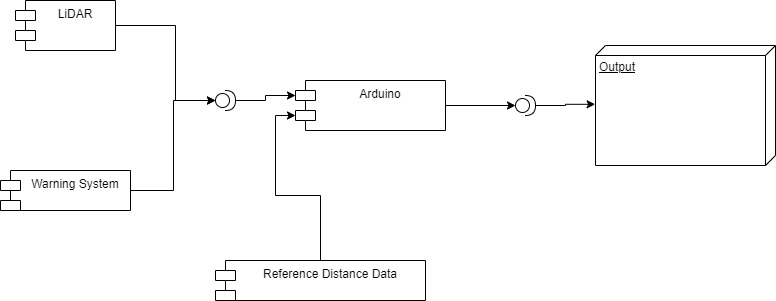
**TEST PLANS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Product ID** | **Description** | **Precondition** | **Expected Input** | **Expected Output** | **Actual Output** |
| H\_01 | Microcontroller | Off state | Power Supply | Switch ON | Switch ON |
| H\_02 | LiDAR Sensor | Off State | Trigger Signal from Controller | Burst of Light Pulses | Burst of Light Pulses |
| H\_03 | Arduino IDE | Initialize the IDE | Including required Libraries | Inclusion of all required libraries into the IDE | Inclusion of all required libraries into the IDE |
| H\_04 | 5V DC Power Supply | Power Off | Supply from portable power source | Powering up the entire system. | Powering up the entire system. |
| H\_01\_L\_01 | LED | Power Off | Signal from Controller | Glowing LED | Glowing LED |
| H\_01\_L\_02 | Electronic Buzzer | Power Off | Signal from Controller | A hooting electronic buzzer | A hooting electronic buzzer |
| H\_01\_L\_03 | 7 segment LCD Display | Power Off | Signal from Controller | LCD Display Power On | LCD Display Power On |
| H\_01\_L\_04 | Vibration Motor | Power Off | Signal from Controller | Activated Vibration Motor | Activated Vibration Motor |
| H\_01\_L\_05 | A physical Mount | Attached to the vehicle | Fixing the sensor to mount | None | None |

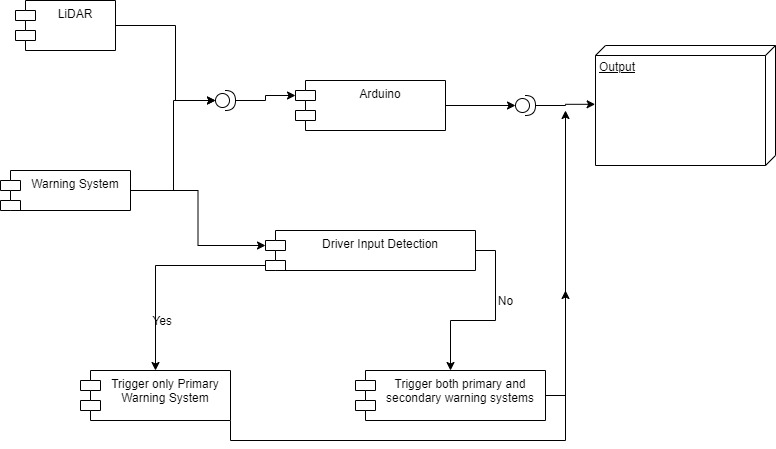
**Table 4: Test Plans**

**DESIGN PLANS FOR THE MAJOR PORJECT USING UML DIAGRAMS**

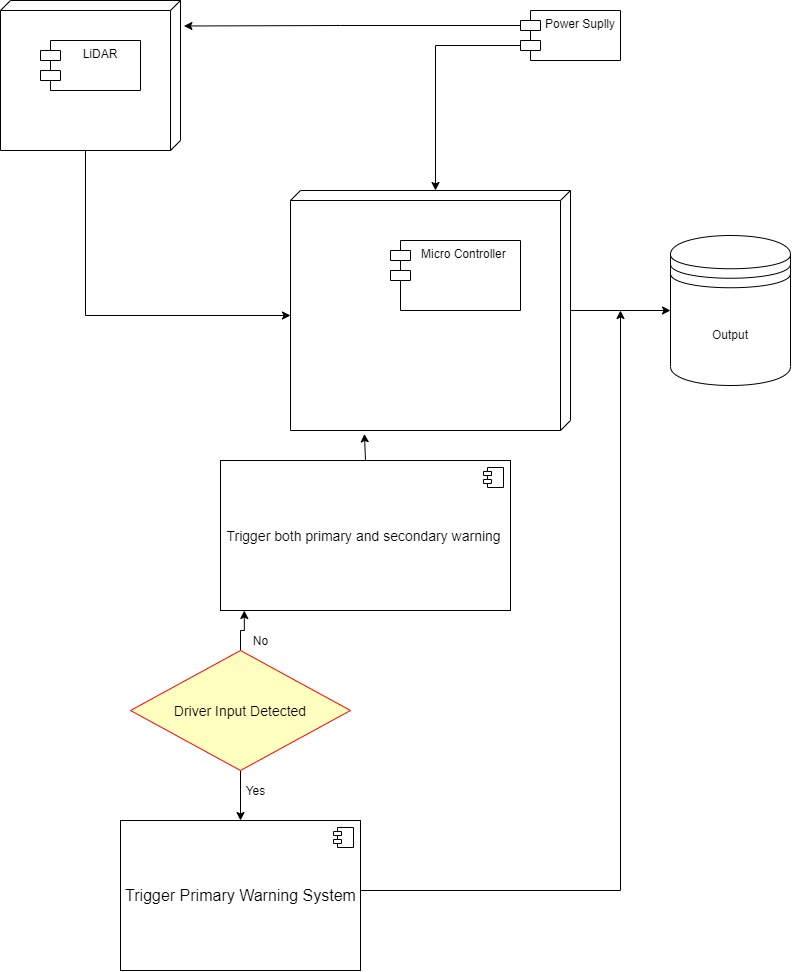
**STRUCTURAL UML DIAGRAMS:**

****

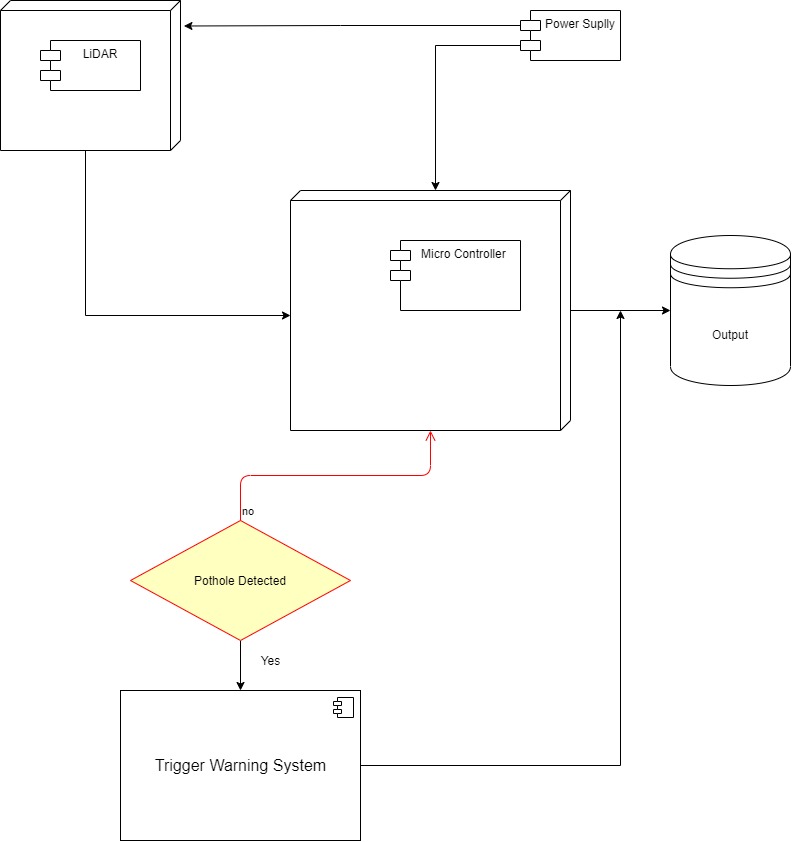
**Fig 1: Component Diagram for High Level Requirements**

****

**Fig 2: Component Diagram for Low Level Requirements**

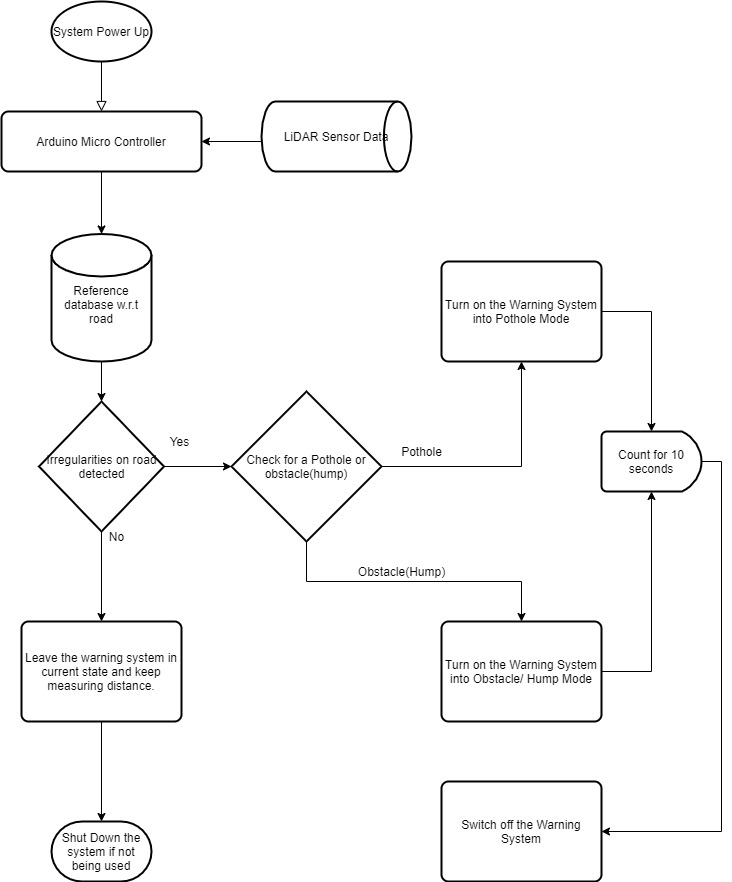
****

**Fig 3: Deployment Diagram for High Level Requirements**

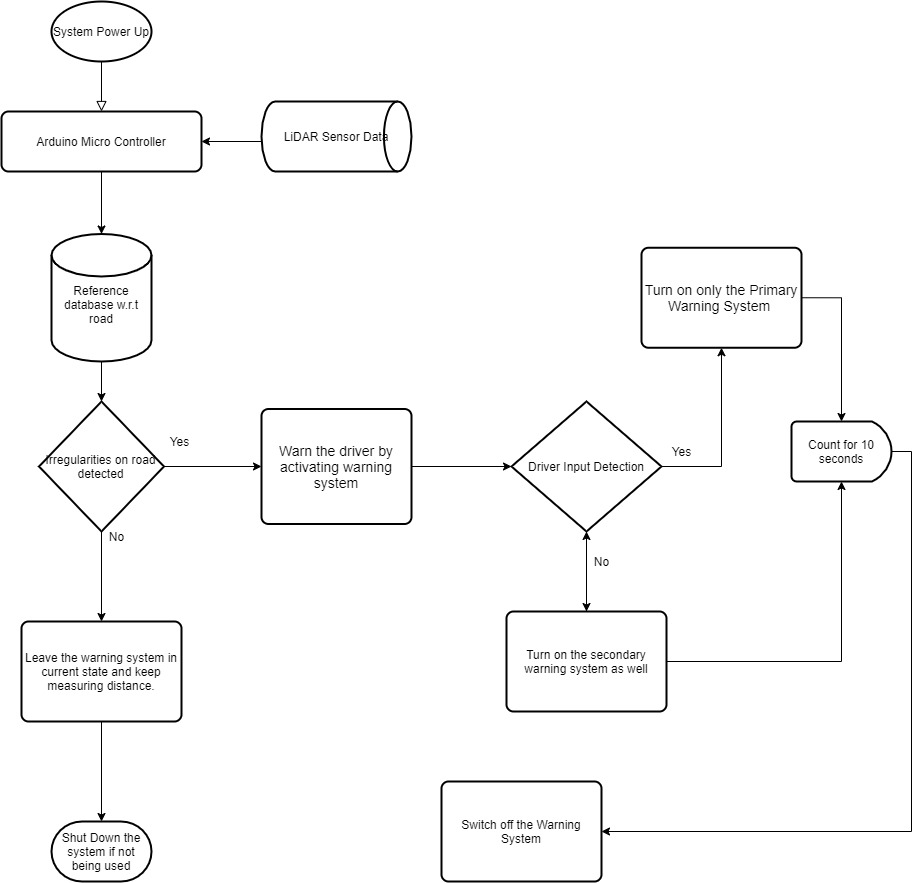
****

**Fig 4: Deployment Diagram for Low Level Requirements**

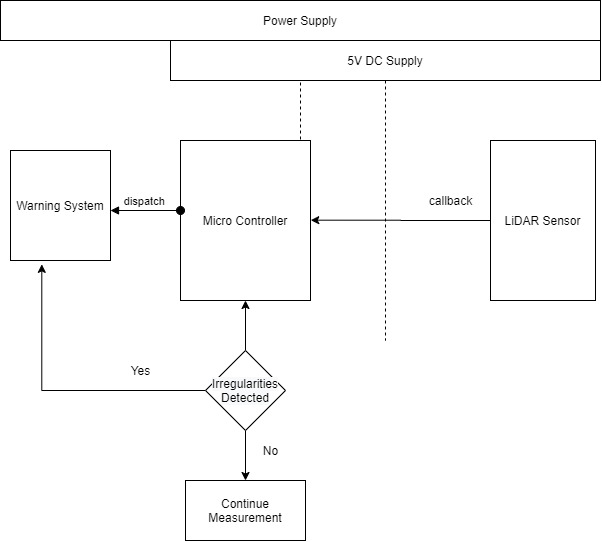
**BEHAVIORAL UML DIAGRAMS:**



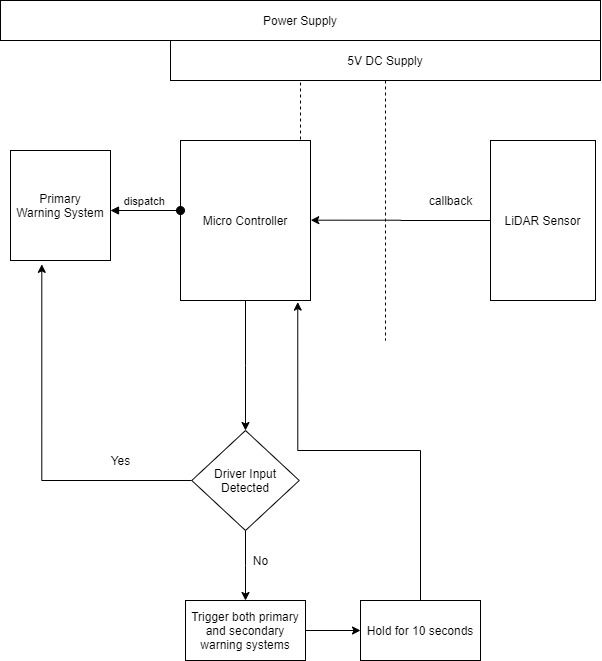
**Fig 5: Activity Diagram for High Level Requirements**

****

**Fig 6: Activity Diagram for Low Level Requirements**

****

**Fig 7: Sequence Diagram for High Level Requirements**

****

**Fig 8: Sequence Diagram for High Level Requirements**

**TEST PLANS FOR THE PROJECT OUTPUT**

**High Level Requirements**

**Requirement Based Test Plans:**

* Check if the controller is receiving 5V DC Power Supply.
* Check if the peripherals are being supplied power from the controller.
* Check the maximum current drawn by each peripheral.
* Check if the maximum current drawn is under the rated limits of the components.
* Check if the LiDAR Sensor receives a minimum of 3.5V which is the operating voltage.
* Check if the warning system is mandatorily connected in a parallel connection.
* Check if the portable power source is charged up to a recommended capacity.
* Check if the frequency of the processor is at sufficient speed.
* Check the executional programming code flashed into the controller.
* Check the working of every component of primary warning system.

**Scenario Based Test Plans:**

* Check how the system reacts to multiple requirement based test plans.
* Check how the system reacts when the power supply is running low.
* Check how the system reacts under sensor malfunction due to power surge.
* Check how the system reacts to malfunction of primary warning systems.
* Check how the system reacts to malfunction of detection sensor.
* Check how the system reacts to low voltage and low current differently.
* Check how the system reacts when there is no response from the primary warning system.

**Boundary Based Test Plans:**

* Check the maximum and minimum voltage levels at which the system works without malfunctions.
* Check the maximum and minimum current levels at which the system works without malfunctions.
* Check the harshest environmental conditions in which the LiDAR sensor sends accurate distance values to the controller.
* Check the maximum and minimum temperatures at which the entire system functions without crippling.
* Check the maximum rainfall density in which the sensor can produce accurate distance values without garbage values.
* Check different angles of attack of the sensor before mounting it to the vehicle.
* Check the maximum distance the sensor can measure before producing garbage values. This can be found in datasheets.
* Check how the system reacts when the sensor is calibrated to its maximum detection range.

**Low Level Requirements**

**Requirement Based Test Plans:**

* Check if the secondary warning system is receiving the required power.
* Check if all the components of the secondary warning system are working properly.

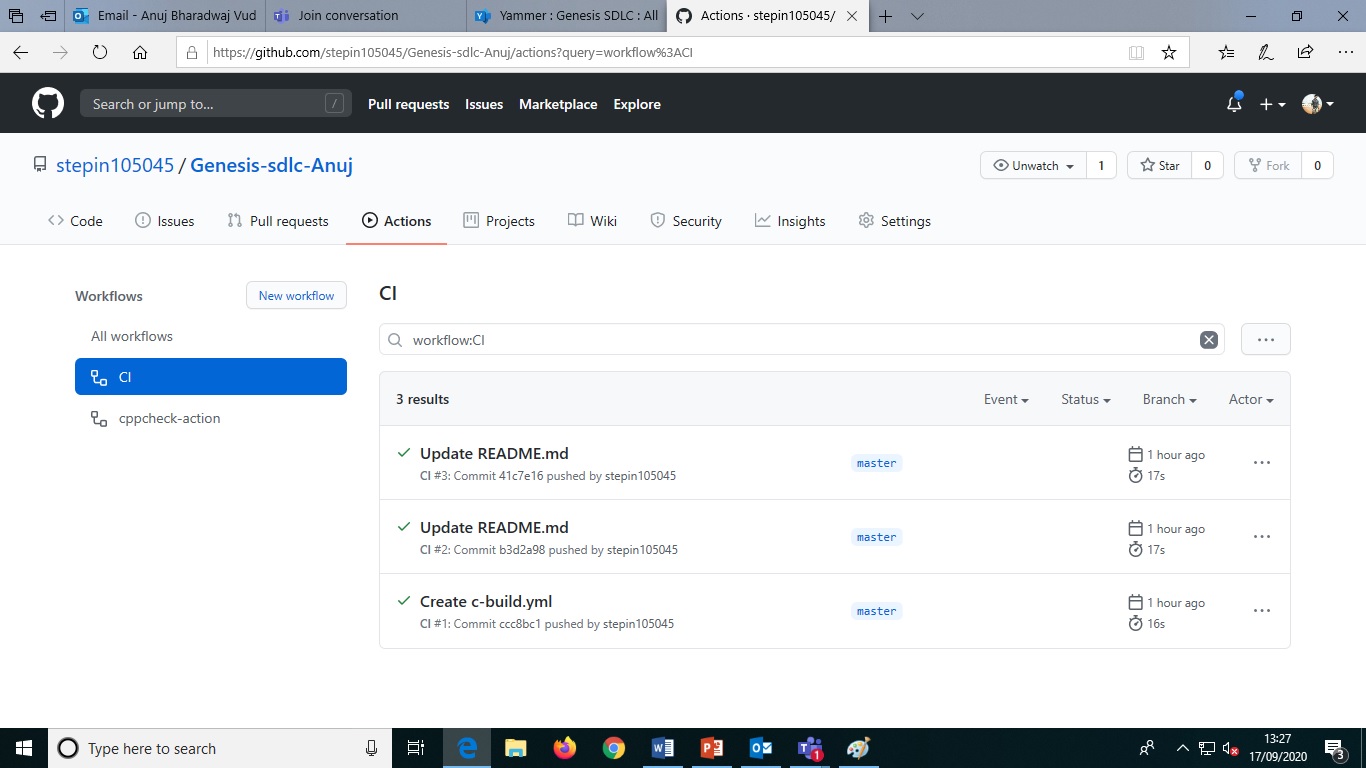
**Scenario Based Test Plans:**

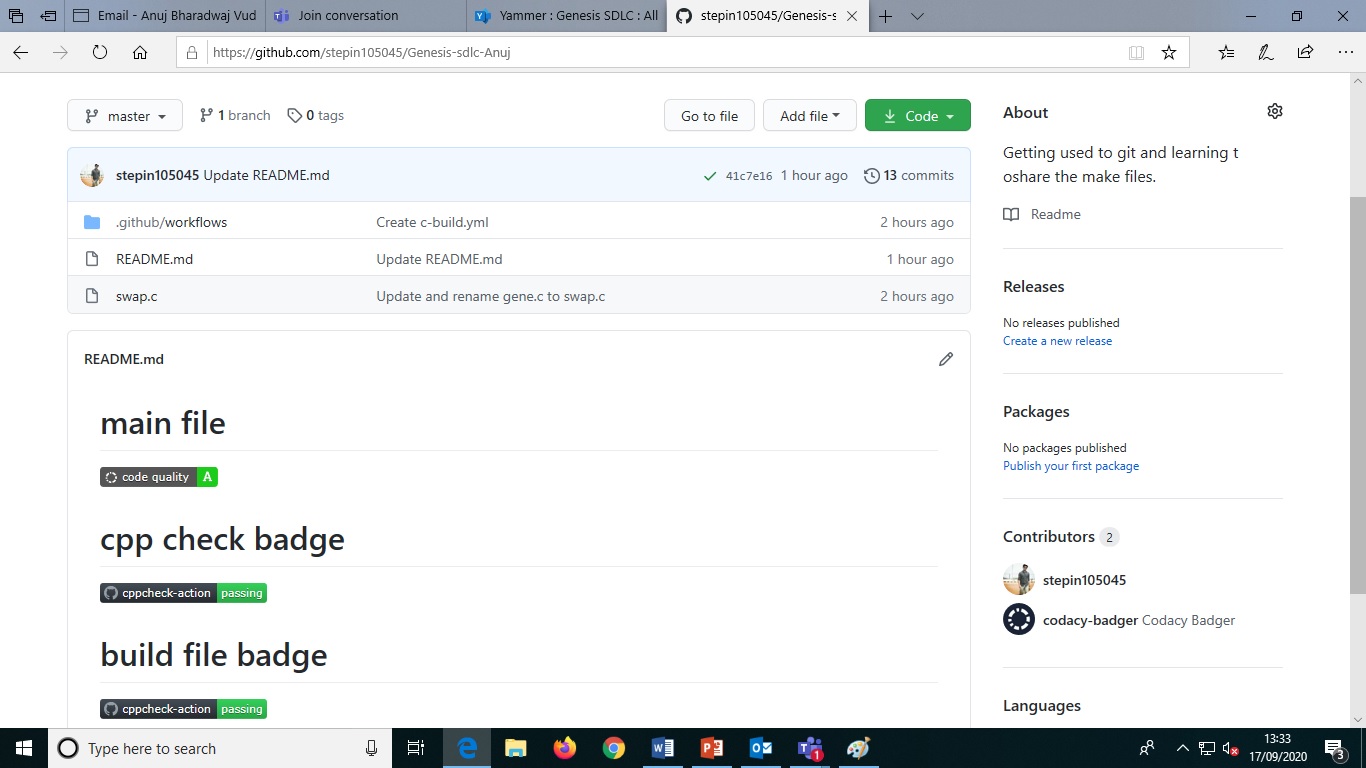
* Check how quick the secondary warning system swings into action after being called upon.
* Check how the system behaves if there is a malfunction in the primary warning system.

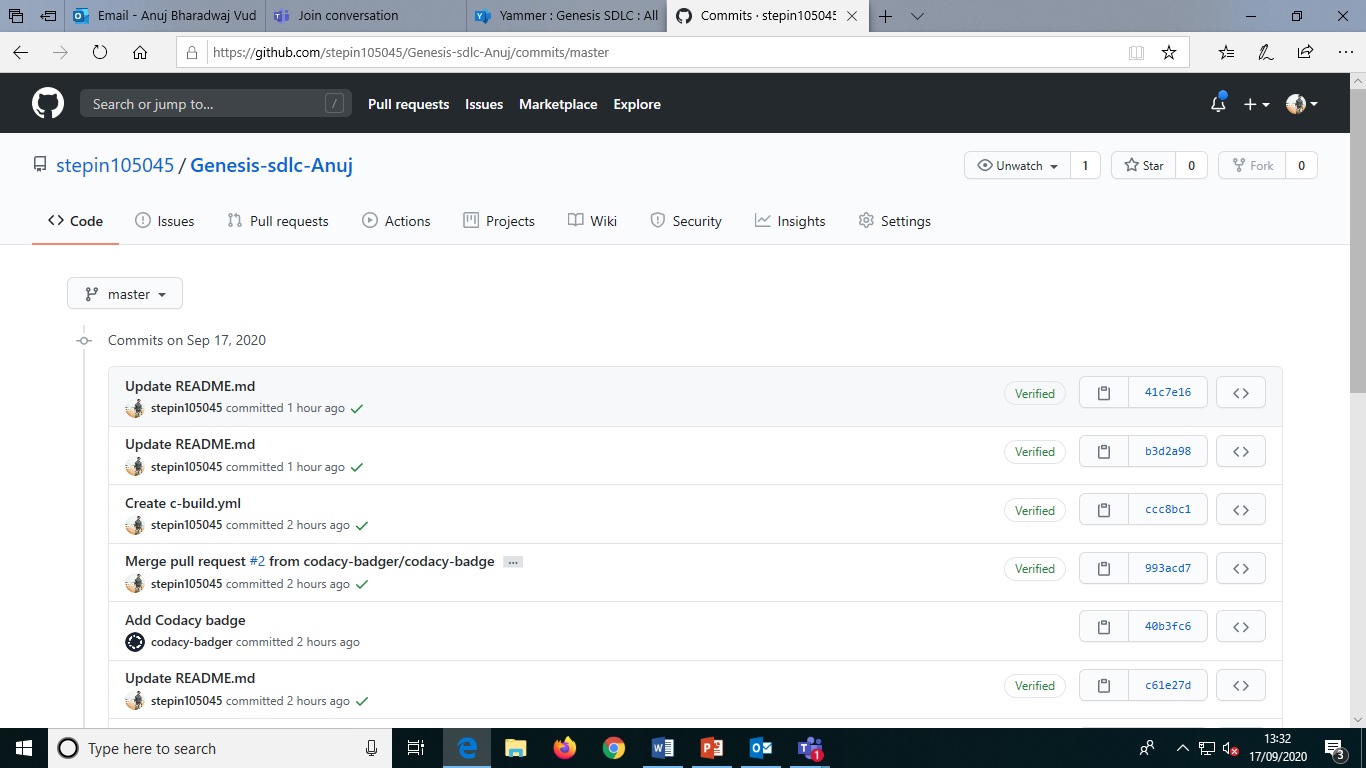
**Boundary Conditions Based Test Plans:**

* Check how the entire system behaves when only the secondary warning system works due to some technical issues.
* Check how the system reacts to the harsh environment, temperature and humidity.

**SCREENSHOTS OF GITHUB REPOSITORY**







**APPENDIX**

The code for the project can be found below:

#include<stdio.h>

int main() {

double first, second, temp;

printf("Enter first number: ");

scanf("%lf", &first);

printf("Enter second number: ");

scanf("%lf", &second);

// Value of first is assigned to temp

temp = first;

// Value of second is assigned to first

first = second;

// Value of temp (initial value of first) is assigned to second

second = temp;

printf("\nAfter swapping, firstNumber = %.2lf\n", first);

printf("After swapping, secondNumber = %.2lf", second);

return 0;

}

Link to the GitHub repository:

<https://github.com/stepin105045/Genesis-sdlc-Anuj>

**AGILE MODEL**

Agile is an incremental and iterative approach of designing and developing a software or a system. The agile model is found very useful for projects which do not have a defined timeline and the customer requirements are prone to change in course of time. This model enables the customers to modify the requirements up to a certain time limit. The Manifesto for Agile Software Development is based on twelve principles:

* Customer satisfaction by early and continuous delivery of valuable software.
* Welcome changing requirements, even in late development.
* Deliver working software frequently (weeks rather than months)
* Close, daily cooperation between business people and developers
* Projects are built around motivated individuals, who should be trusted
* Face-to-face conversation is the best form of communication (co-location)
* Working software is the primary measure of progress
* Sustainable development, able to maintain a constant pace
* Continuous attention to technical excellence and good design
* Simplicity—the art of maximizing the amount of work not done—is essential
* Best architectures, requirements, and designs emerge from self-organizing teams
* Regularly, the team reflects on how to become more effective, and adjusts accordingly

Many industrial giants which are the top and important players in their respective fields, adopt to the Agile methodology. This is used to continuously monitor and report progress on the system development to their respective clients.

**User Stories**

User stories are the smallest units of user functionality in agile which can be delivered in one agile sprint. They are typically estimated using story pointed and defined using INVEST criteria. User stories should deliver a vertical slice of functionality to the customer that is valuable and complete by the end of an iteration. A user story must deliver particular value to the user and must be describable in simple language that outlines the desired outcome.

Let’s look at some user stories of some real time projects and scenarios belonging to big players in the industry.

**User Story 1:**

As a standard requirement, a certain car manufacturing company wants a 4 channel Anti-Lock Braking System controlled using a single, fast and reliable Electronic Control Unit and Pneumatic or Hydraulic Actuators whichever is better and more suitable for the vehicle in discussion.

The THEME at large in this story is to design a system which meets the expectations of improvements to the current technologies related to that field.

This falls under the EPIC of device manufacturing for the entire automobile industry.

**User Story 2:**

A certain airplane manufacturing company has quoted a requirement of an entire fly by wire control system design which can reduce the lag in effect of an action executed in the cockpit of the airplane.

The THEME at large in this story is to design a system which meets the expectations of improvements to the current technologies related to that field.

This falls under the EPIC of device manufacturing for the entire aviation industry.

**User Story 3:**

A certain company was given a requirement quote to design a software which can identify lapses in efficiency of a manufacturing plant and suggest alternative methods of operations to reduce the inefficiency and wastage from the same.

The THEME at large in this story is to design a software which meets the expectations to find out the lapses as defined by the customer.

This falls under the EPIC of software design and development for manufacturing industry.

**REFERENCES**

[1] Hsiu-Wen Wang (2015). “A Real-Time Pothole DetectionApproach for Intelligence Transportation System.”*Mathematical Problems in Engineering*, Article ID 869627.

[2] Seung-Ki Ryu (2015). “Image-Based Pothole Detection System for ITS Service and Road Management System.” *Mathematical Problems in Engineering*, Article ID 968361.

[3] Youngtae Jo, Seung-ki Ryu. (2015). “Pothole Detection using a Black Box Camera .” ISSN 1424-8220.

[4] Prof. R.M. Sahu, Laxmi Panchal. (2017). “Automatic Detection of Potholes and Humps on Roads to Aid Drivers.” IJARIIE-ISSN(O)-2395-4396, Vol 3 Issue 2 2017.

[4] Shambhu Hedge (2014). “Pothole Detection and Inter Vehicular Communication.” *2014 IEEE International Conference on Vehicular Electronics and Safety,* ISBN 978-1-4799-1882-9.

[5] B. G. Shivaleelevathi (2019). “Design and Development of an Intelligent System for Pothole and Hump Identification on Roads.” *International Journal of Recent Technology and Engineering,* Vol 8, Issue 3, ISSN 2277-3878.

**MULTIPURPOSE CALCULATOR**

**Requirements Mapping:**

|  |  |
| --- | --- |
| **Name** | **Description** |
| H\_01 | Operand1 |
| H\_02 | Operand 2 |
| H\_03 | Basic Arithmetic Ops (add, sub, mul, div) |
| H\_01\_L\_01 | Find out if number is Prime |
| H\_01\_L\_02 | Find out if number is positive |
| H\_01\_L\_03 | Find out if number is negative |
| H\_01\_L\_04 | Find out if number is equal to zero |
| H\_01\_L\_05 | Find out if the Modulo of two numbers |
| H\_01\_L\_06 | Find out if a number is even |
| H\_01\_L\_07 | Find out if a number is odd |
| H\_01\_L\_08 | Find out the percentage of the number |
| H\_01\_L\_09 | Find out the factorial of given number |
| H\_01\_L\_09 | Find out area of a parallelogram |

**Table 3: Requirements Mapping**

**UML DIAGRAM**

# 

Figure 4: UML Class Diagram

**TEST PLANS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Product ID** | **Description** | **Precondition** | **Expected Input** | **Expected Output** | **Actual Output** |
| H\_01 | Operand 1 | Uninitiated | Initiate | Initiated | Initiated |
| H\_02 | Operand 2 | Uninitiated | Initiate | Initiated | Initiated |
| H\_03 | Arithmetic Operators | Undefined | Define basic operators | Defined basic operators | Defined Basic Operators |
| H\_01\_L\_01 | Prime number | Not verified | Any integer number | Verify if prime number or not | Verify if prime number or not |
| H\_01\_L\_02 | Positive Number | Not verified | Any integer number | Verify if positive number or not | Verify if positive number or not |
| H\_01\_L\_03 | Negative Number | Not verified | Any integer number | Verify if negative number or not | Verify if negative number or not |
| H\_01\_L\_04 | Zero | Not verified | Any integer number | Verify if number is 0 or not | Verify if number is 0 or not |
| H\_01\_L\_05 | Even number | Not verified | Any integer number | Verify if even number or not | Verify if even number or not |
| H\_01\_L\_06 | Odd number | Not verified | Any integer number | Verify if odd number or not | Verify if odd number or not |
| H\_01\_L\_07 | Percentage | Not calculated | 2 integer numbers | Calculate percentage of a number | Calculate percentage of a number |
| H\_01\_L\_08 | Factorial | Not calculated | Any integer number | Calculate the factorial of a number | Calculate the factorial of a number |
| H\_01\_L\_08 | Area | Not calculated | Two float nums | Calculate the area of a parallelogram | Calculate the area of a parallelogram |

**Table 4: Test Plans**

**GITHUB REPOSITORY**

The link to the GitHub repository: <https://github.com/99002642>

**SCREENSHOTS**

Commit History (Number of Commits): 62

# C:\Users\Mahavir\Desktop\gen\calc ops\git rep screen.JPG

Makefile.yml :

# C:\Users\Mahavir\Desktop\gen\calc ops\makefile screen.JPG

Badges Acquired from Codacy, Cunit, Cpp Check, CI:

