Details

Ver. Rel. No.	Release Date	Prepared By	Reviewed By	To Be Approved	Remarks/Revision Details
1.0	16/02/2022	Varadaraju Dileep Kumar 40020532			

Contents

List of Figures

MiniProject – 1: Snakes and Ladders Game [Individual]

Modules:

1. C Programming
2. Makefile

Requirements

4W's and 1 H's

What:

1. Implementation of Offline Game. It can be played two players.

Where:

1. This can be used in our daily lives to make better financial decisions.
2. To be available in Visual Studio Code.

Who:

1. Players who wants to have some fun game Snakes and Ladders.

When:

1. To what fun and get relived from stress.

How:

1. Using C programming language.

High Level Requirements

ID	Description	Status
HLR_1	User should be able to give name as input	Implemented
HLR_2	User should be able to perform his dice	Implemented
HLR_3	When checks into snake head, check to the snake tail	Implemented
HLR_4	When checks into start Ladder, check to the end ladder	Implemented

Low Level Requirements

ID	Description	Status
LLR_1	If player reaches to 100, Its win	Implemented
LLR_2	Other player consider as lost	Implemented

Design

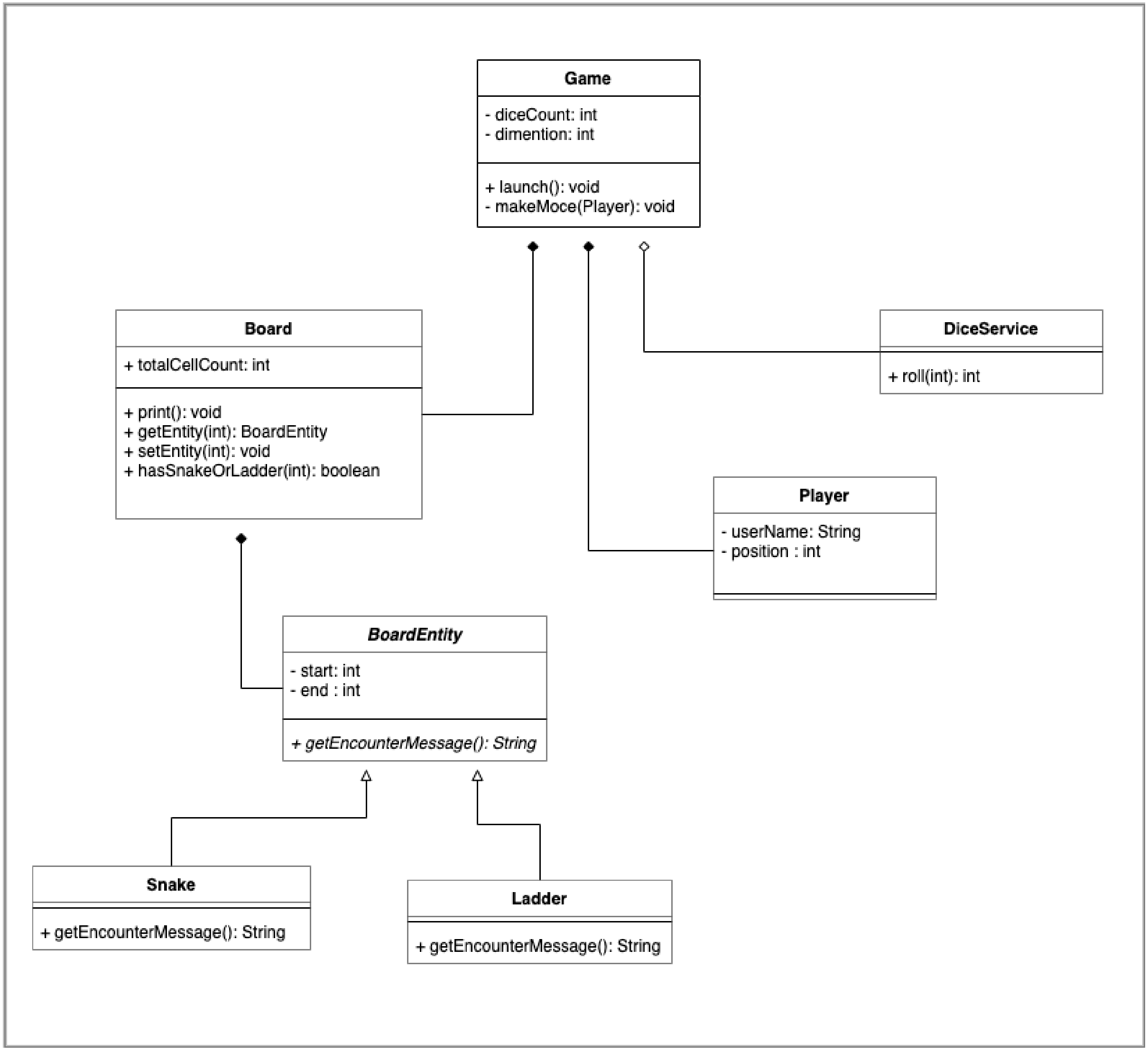


Figure 1 Behaviour Diagram

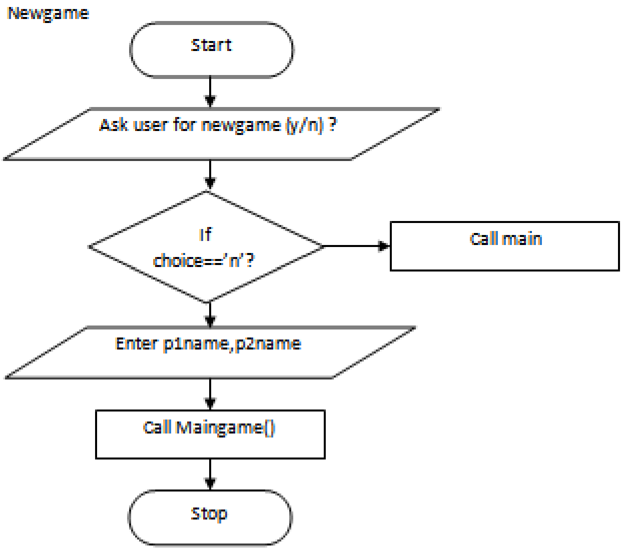


Figure 2 Structure Diagram

Test Plan

High Level Test Plan

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
HLTP_1	10x10 board contains numbers from 1 to 100	Choice	SUCCESS	SUCCESS	Requirement Based
HLTP_2	Dice Value	Choice	SUCCESS	SUCCESS	Requirement Based
HLTP_3	No. of player has to role the dice to move the player to the respective number by dice	Choice	SUCCESS	SUCCESS	Requirement Based
HLTP_4	Player lands on snakes head will be degraded to the tail of the snake	Choice	SUCCESS	SUCCESS	Requirement Based

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
HLTP_5	Player lands on lower ladder will be placed on the upper ladder number on the board	Choice	SUCCESS	SUCCESS	Requirement Based

Low Level Test Plan

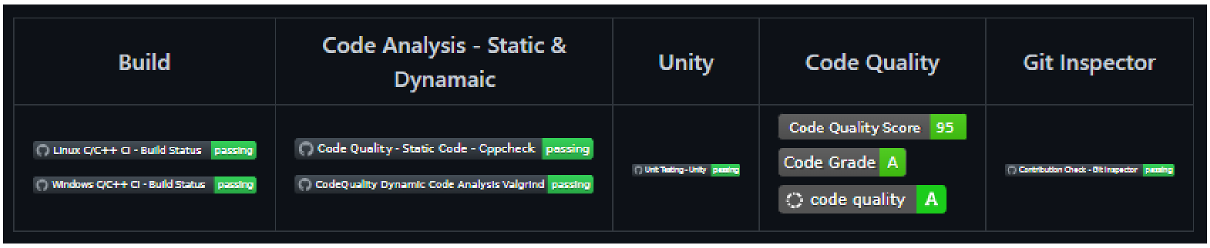
ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
LLTP_1	Dice has value of 1 to 6	Any number between 1 to 6	Random number	0.167	Requirement Based
LLTP_2	Play proceeds with the player1 and player2 placing their marks on number 1. Check if any player/computer roles the dice	Choice	Success	Success	Requirement Based
LLTP_3	Check if a player gets number 6 on dice that player will get another chance of rolling the dice	Choice	Success	Success	Requirement Based
LLTP_4	If player gets numbers expect number 6, the other player gets chance to role the dice vice versa after moving the player on number board	Choice	Success	Success	Requirement Based
LLTP_5	1 is the starting point and 100 is the win	Choice	Success	Success	Requirement Based

Implementation and Summary

Git Link:

Link: https://github.com/dileep40020532/M1_Game_SL.git

Git Dashboard

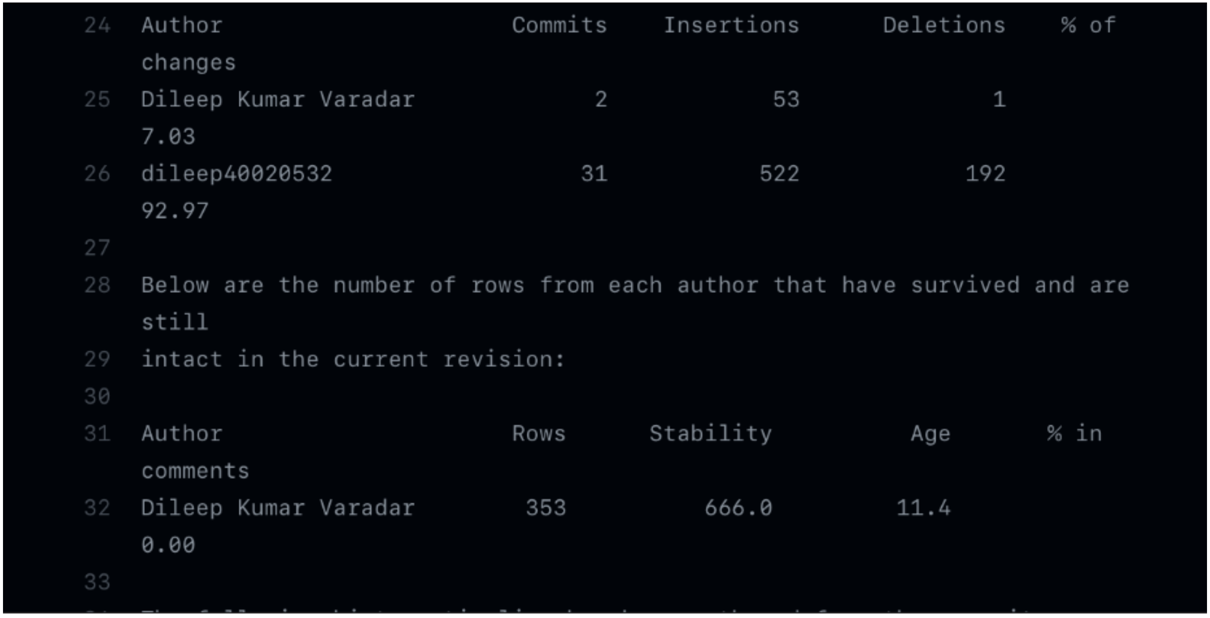


Build	Code Analysis - Static & Dynamaic	Unity	Code Quality	Git Inspector
Linux C/C++ CI - Build Status passing Windows C/C++ CI - Build Status passing	Code Quality - Static Code - Cppcheck passing CodeQuality Dynamic Code Analysis Valgrind passing	Unit Testing - Unity passing	Code Quality Score 95 Code Grade A code quality A	Contribution Check - Git Inspector passing

Figure 3 Git Dashboard

Summary

Git Inspector Summary



```
Author                    Commits     Insertions      Deletions    % of
changes
Dileep Kumar Varadar            2             53              1
7.03
dileep40020532                 31            522            192
92.97

Below are the number of rows from each author that have survived and are
still
intact in the current revision:

Author                    Rows      Stability           Age       % in
comments
Dileep Kumar Varadar       353          666.0          11.4
0.00

```

Figure 4 Git Inspector Summary

MiniProject 2 – Temperature Control Chair [Individual]

Modules

1. C Programming
2. Embedded System
3. SimulIDE
4. Git

Requirements

4W's and 1 H's

Why:

1. To provide better comfort

Where:

1. This can be used in our daily lives.
2. We can use it in the car seat and automotive industries.

What:

1. Automotive Temperature Monitoring System.

When:

1. One can adjust the temperature of chair in winter season.

How:

1. By giving inputs we can operate the temperature of the chair.

High Level Requirements

ID	Description	Status
HLR_1	Microcontroller	Implemented
HLR_2	LCD Display	Implemented
HLR_3	Heat Generation	Implemented

ID	Description	Status
HLR_4	Temperature Sensor	Implemented

Low Level Requirements

ID	Description	Status
LLR_1	Check the person is available or not	Implemented
LLR_2	Select from the menu	Implemented
LLR_3	ATmega328	Implemented
LLR_4	Display the value of temperature of the seat	Implemented

Design

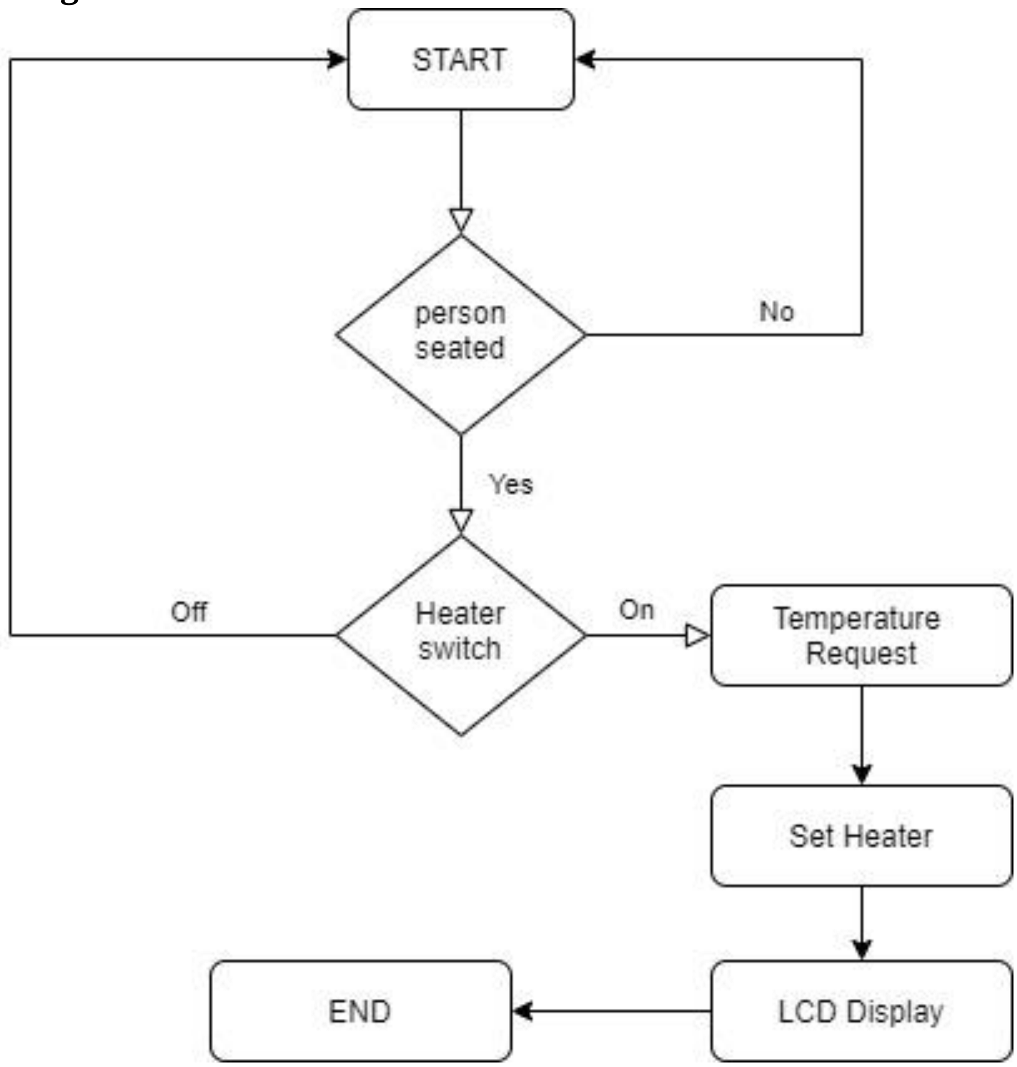


Figure 5 Behaviour Diagram

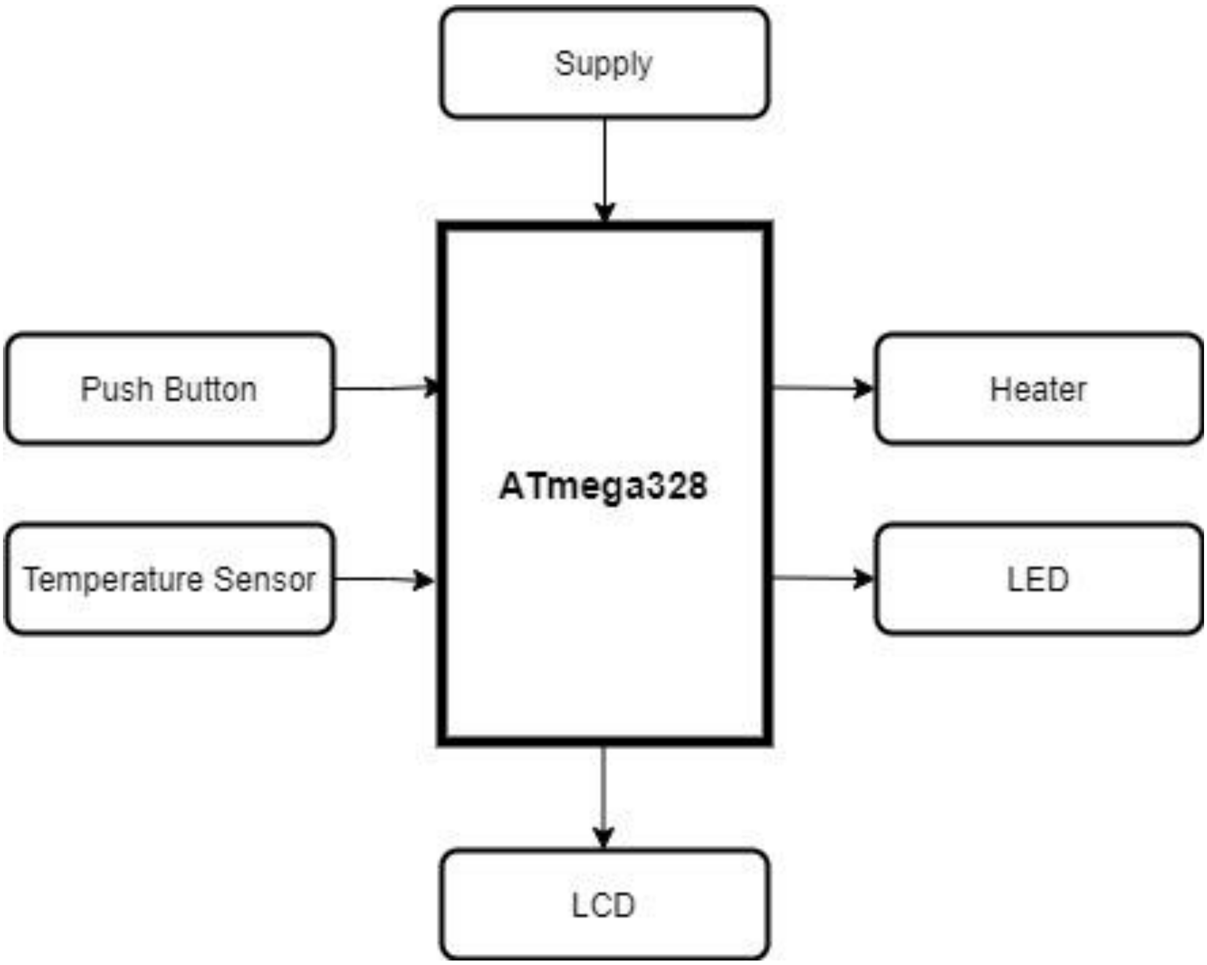


Figure 6 Structure Diagram

Test Plan

High Level Test Plan

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
HLTP_1	Switch On	High	The Program Should Start	SUCCESS	Requirement Based
HLTP_2	Taking Input from The User	Value Input	Should Give Output fo The User	SUCCESS	Scenario Based
HLTP_3	Giving Output from The Input	Value Input from The User	Displays Output	SUCCESS	Boundary Based

Low Level Test Plan

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
LLTP_1	Performing Arithmetic operation '+'	(600, 100)	700	700	Requirement Based
LLTP_2	Performing Arithmetic operation '-'	(800, 100)	700	700	Requirement Based
LLTP_3	Performing Arithmetic operation '*'	(10, 7)	70	70	Requirement Based
LLTP_4	Performing Arithmetic operation '/'	(10, 2)	5	5	Requirement Based

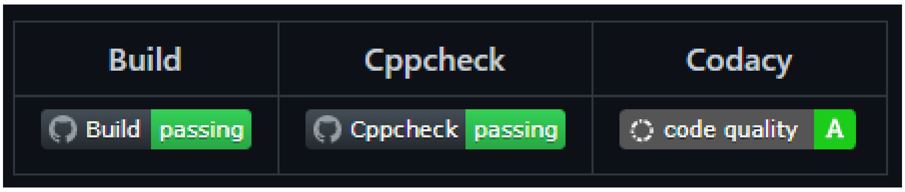
Implementation and Summary

Git Link:

Link: https://github.com/dileep40020532/M2-Embedded_TemperatureControlChair.git

Git Dashboard

Figure 7 Git Dashboard



Build	Cppcheck	Codacy
Build passing	Cppcheck passing	code quality A

MiniProject 3 – NFT Marketplace [Team]

Modules

1. SDLC
2. Git

Requirements

4W's and 1 H's

Why:

1. It can be used by anyone at any place.
2. Digital Items are the Future
3. Individual Creators can use this platform to sell the Digital products.

Where:

1. This can be used in our daily lives.
2. Can be used for international transactions

Who

1. It can be used by anyone.
2. It can be used as a reference for marketplace.

When:

1. One can buy, sell or create anytime.
2. The project can be used when the anyone wants to buy an NFT.
3. Can be used without any centralised authority

How:

1. By using a crypto wallet anyone can Buy or Bid on NFT.
2. It will be helpful for Digital Creators.

High Level Requirements

ID	Description	Status
HLR_1	Create NFT	Implemented
HLR_2	Sell NFT	Implemented
HLR_3	Bid NFT	Implemented
HLR_4	Buy NFT	Implemented
HLR_5	Contact	Implemented

Low Level Requirements

ID	Description	Status
LLR_1	Sign In	Implemented
LLR_2	Register	Implemented
LLR_3	Connect Wallet	Implemented
LLR_4	Activity	Implemented
LLR_5	Forgot password	Implemented
LLR_6	Signup	Implemented

Design

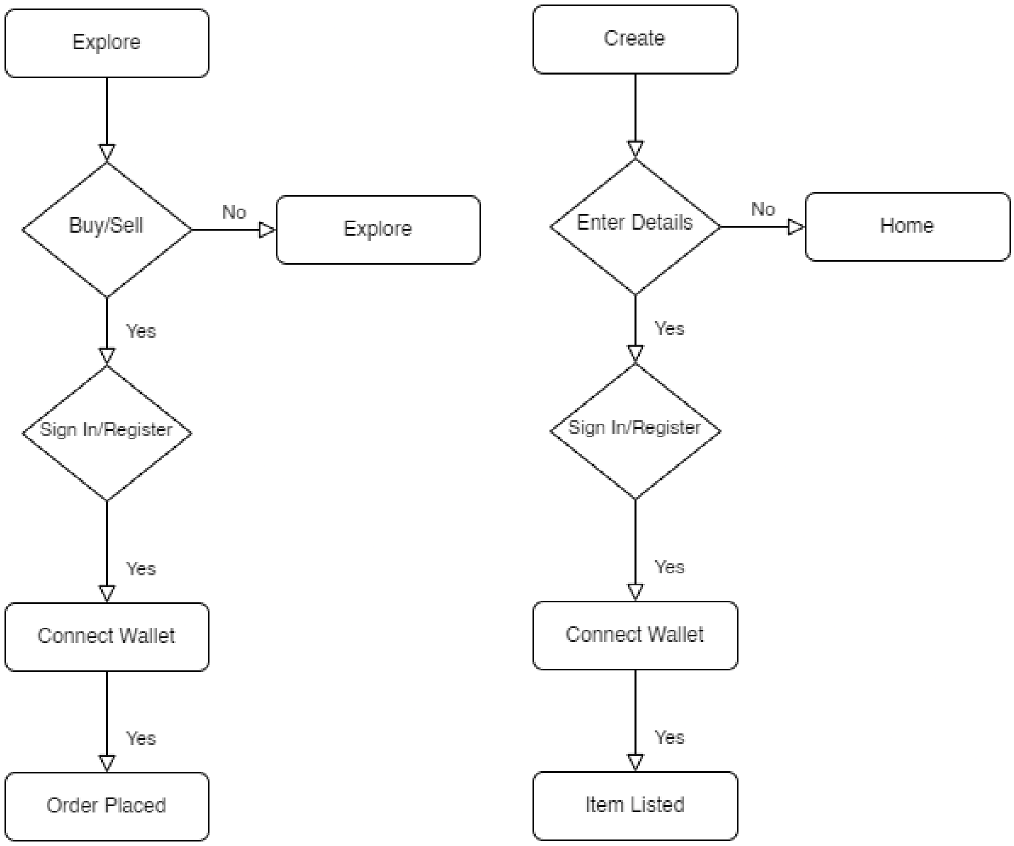


Figure 8 Behaviour Diagram

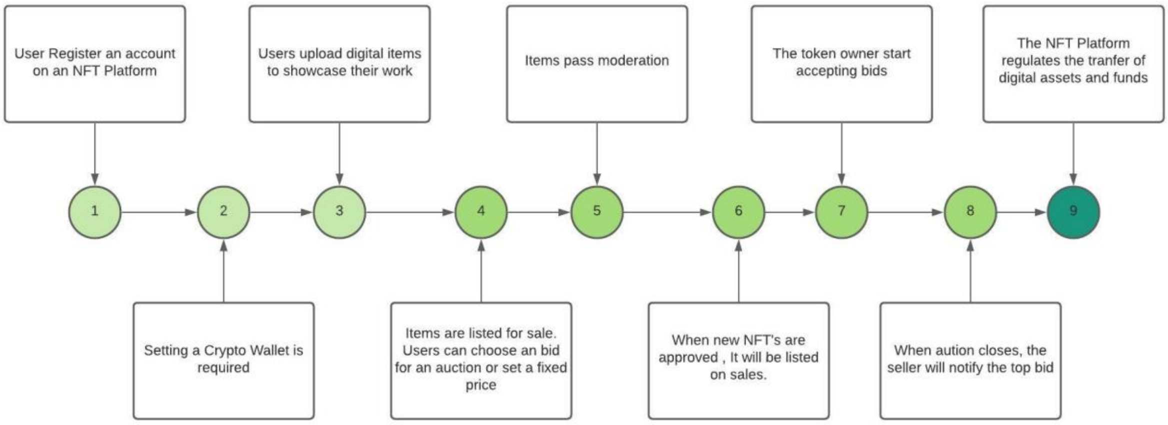

User Register an account on an NFT Platform
Users upload digital items to showcase their work
Items pass moderation
The token owner start accepting bids
The NFT Platform regulates the tranfer of digital assets and funds
1
2
3
4
5
6
7
8
9
Setting a Crypto Wallet is required
Items are listed for sale. Users can choose an bid for an auction or set a fixed price
When new NFT's are approved , It will be listed on sales.
When aution closes, the seller will notify the top bid

Figure 9 UserFlow Diagram

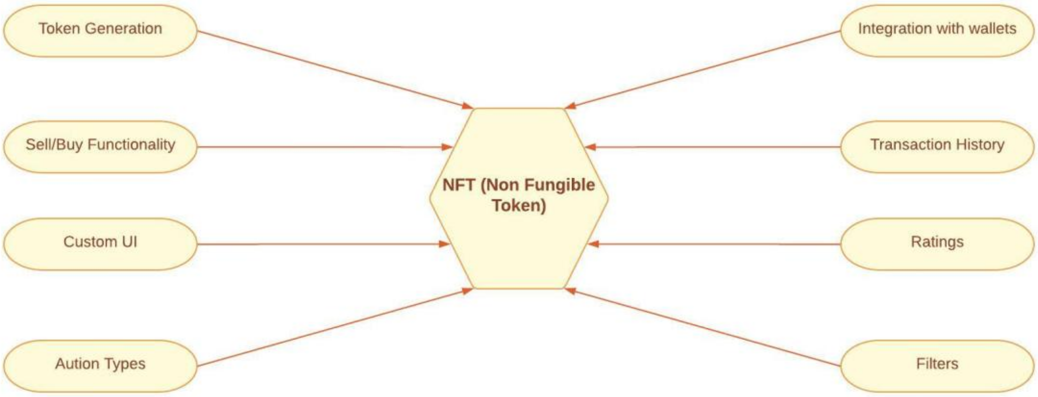


Figure 10 Structure Diagram

Test Plan

High Level Test Plan

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
HLTP_1	Create NFT	Click	SUCCESS	SUCCESS	Requirement Based
HLTP_2	Sell NFT	Click	SUCCESS	SUCCESS	Requirement Based
HLTP_3	Bid NFT	Click	SUCCESS	SUCCESS	Requirement Based
HLTP_4	Buy NFT	Click	SUCCESS	SUCCESS	Requirement Based
HLTP_5	Contact	Click	SUCCESS	SUCCESS	Requirement Based
HLTP_6	Sign In	Click	SUCCESS	SUCCESS	Requirement Based
HLTP_7	Register	Click	SUCCESS	SUCCESS	Requirement Based

Low Level Test Plan

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
LLTP _1	Connect Wallet	Click	SUCCESS	SUCCESS	Requirement Based
LLTP _2	Activity	Click	SUCCESS	SUCCESS	Requirement Based
LLTP _3	Forgot password	Click	SUCCESS	SUCCESS	Requirement Based
LLTP _4	To check whether none of the fields should be empty	Empty value in the input module	Prompt message mandatory field missing	SUCCESS	Requirement Based
LLTP _5	E-mail ID should be in the perfect format i.e. group2@gmail.com	group2@gmail.com	Prompt message invalid E-mail ID	SUCCESS	Requirement Based

Implementation and Summary

Git Link:

Link: https://github.com/GENESIS2021Q1/Applied_SDLC-Dec_Team_2

Live Project Link: https://alrichroshan.com/nft

Individual Contribution and Highlights

Summary

1. Designed Homepage
2. Search Option
3. Header
4. Footer
5. Integrating All Pages Together

Role in Project Team

1. Designer: Designed Webpages Using HTML, CSS, JavaScript
2. Integrator: Integrated All the Pages Together
3. Tester: Testing the Webpage Performance and Bugs

Miniproject 4 – Attendance Automation [Team]

Modules

1. Python
2. Git

Requirements

High Level Requirements

ID	Feature	Status
HLR_01	GUI	Not Implemented
HLR_02	Attendance Status	Implemented
HLR_03	User Details	Implemented
HLR_04	User load sheet	Implemented
HLR_05	Output file generation	Implemented

Low Level Requirements

ID	Feature	High Level ID	Status
LLR_01	GUI should allow user to enter inputs	HLR_01	Not Implemented
LLR_02	Input Files for Different Sessions	HLR_01	Not Implemented
LLR_03	User can get the Attendance Status	HLR_02	Implemented
LLR_04	User can enter status input to get the Attendance Status	HLR_02	Implemented
LLR_05	User can get the user details	HLR_03	Implemented
LLR_06	User will get the details after the successful attendance entry	HLR_03	Implemented

ID	Feature	High Level ID	Status
LLR_07	User can load different sheets	HLR_04	Implemented
LLR_08	User can also modify the existing sheets as it is dynamic	HLR_04	Implemented
LLR_09	Output file gets generated	HLR_05	Implemented
LLR_10	Multiple files can be generated with different inputs	HLR_05	Implemented

Test Plan

High Level Test Plan

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
HLTP_01	Attendance Status	User Input	SUCCESS	SUCCESS	Requirement Based
HLTP_02	User details	User Input	SUCCESS	SUCCESS	Requirement Based
HLTP_03	User load sheet	User Input	SUCCESS	SUCCESS	Requirement Based
HLTP_04	Output file generation	User Input	SUCCESS	SUCCESS	Requirement Based

Low Level Test Plan

ID	HLTP ID	Description	Expected I/P	Actual O/P	Type Of Test
LLTP_01	HLTP_01	User can get Attendance Status	SUCCESS	SUCCESS	Requirement Based
LLTP_02	HLTP_01	User can enter Status input to	SUCCESS	SUCCESS	

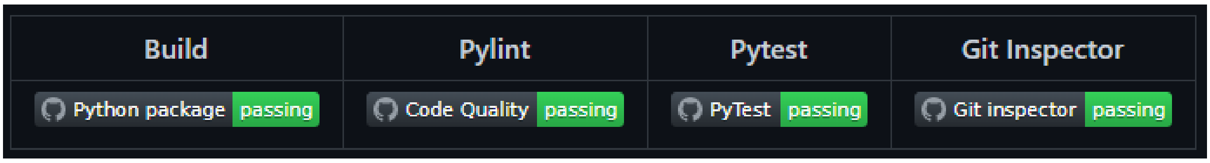
ID	HLTP ID	Description	Expected I/P	Actual O/P	Type Of Test
		get the Attendance Status			
LLTP_03	HLTP_02	User can get the User details	SUCCESS	SUCCESS	Requirement Based
LLTP_04	HLTP_02	User will get the details after the successful attendance	SUCCESS	SUCCESS	Requirement Based
LLTP_05	HLTP_03	User can load different sheets	SUCCESS	SUCCESS	Requirement Based
LLTP_06	HLTP_03	User can also modify the existing sheets as it is dynamic	SUCCESS	SUCCESS	Requirement Based
LLTP_07	HLTP_04	Output file gets generated	SUCCESS	SUCCESS	Requirement Based
LLTP_08	HLTP_04	Multiple files can be generated with different inputs	SUCCESS	SUCCESS	Requirement Based

Implementation and Summary

Git Link:

Link: https://github.com/alrichroshan/Attendance_Automation_Team_14

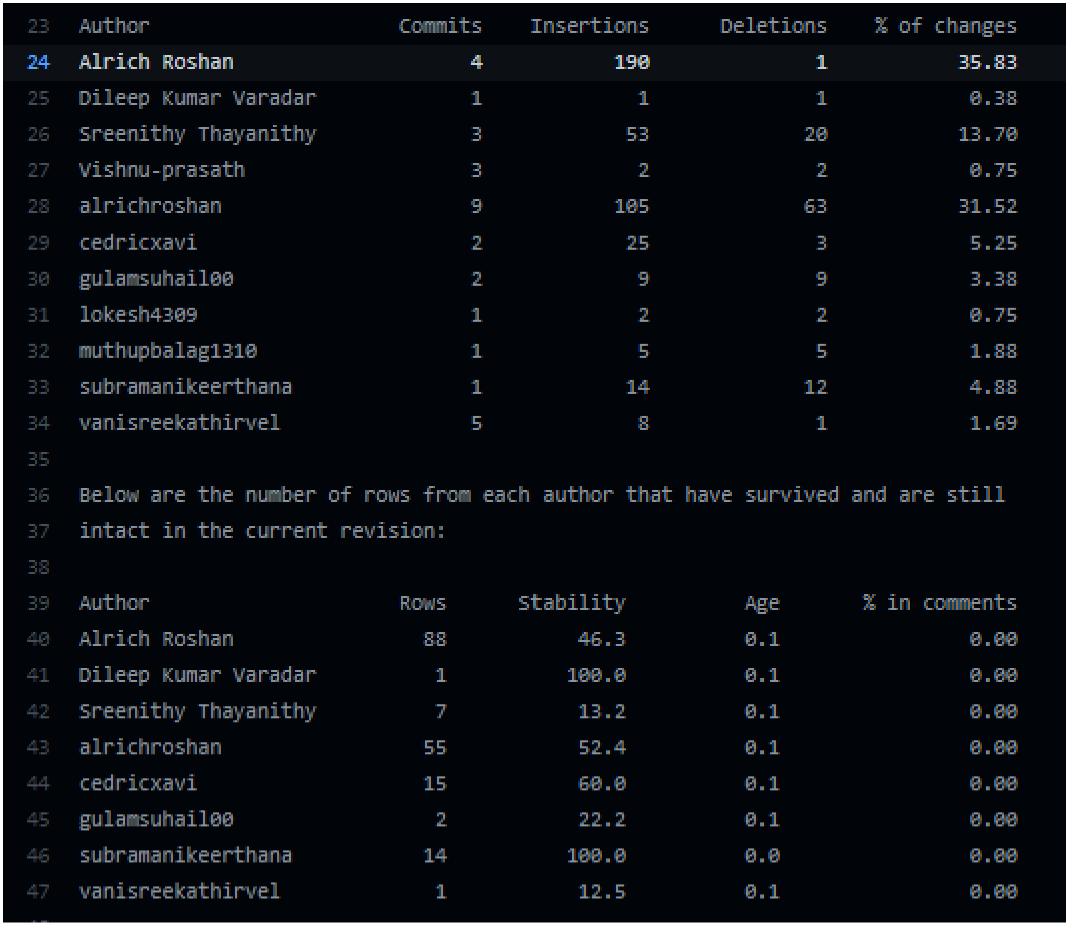
Git Dashboard



Build	Pylint	Pytest	Git Inspector
Python package passing	Code Quality passing	PyTest passing	Git inspector passing

Figure 11 Git Dashboard

Git Inspector Summary



```
Author                    Commits   Insertions      Deletions    % of changes
Alrich Roshan                   4          190              1           35.83
Dileep Kumar Varadar            1            1              1            0.38
Sreenithy Thayanithy            3           53             20           13.70
Vishnu-prasath                  3            2              2            0.75
alrichroshan                    9          105             63           31.52
cedricxavi                      2           25              3            5.25
gulamsuhail00                   2            9              9            3.38
lokesh4309                      1            2              2            0.75
muthupbalag1310                 1            5              5            1.88
subramanikeerthana              1           14             12            4.88
vanisreekathirvel               5            8              1            1.69

Below are the number of rows from each author that have survived and are still
intact in the current revision:

Author                     Rows      Stability          Age       % in comments
Alrich Roshan                88           46.3          0.1                0.00
Dileep Kumar Varadar          1          100.0          0.1                0.00
Sreenithy Thayanithy          7           13.2          0.1                0.00
alrichroshan                 55           52.4          0.1                0.00
cedricxavi                   15           60.0          0.1                0.00
gulamsuhail00                 2           22.2          0.1                0.00
subramanikeerthana           14          100.0          0.0                0.00
vanisreekathirvel             1           12.5          0.1                0.00
```

Figure 12 Git Inspector Summary

Individual Contribution and Highlights

1. Improved implementation of Python Programming
2. Source code management using GitHub

Role in Project Team

1. Programmer: Done Programming for Attendance Automation
2. Integrator: Integrated all the codes
3. Tester: Writing Testcases and testing the integrated code

MiniProject 5 – Mahindra Project [Team]

Modules

1. MATLAB
2. Git

Requirements

We have implemented following features

1. Adaptive Cruise Control System
2. Auto AC Temperature Control
3. Exterior Lighting Control
4. Sunroof Control
5. Auto Transmission Control
6. Wiper Control System
7. Anti – Lock Breaking System
8. Door Lock Control System

Design

This project was implemented using Matlab.

MiniProject 6 – Wiper Control System [Team]

Modules

1. C Programming
2. STM32

Requirements

4W's and 1'H

Who:

1. Users who drives the vehicles can use.

What:

1. Wipers may be powered by a variety of means, although most in use today are powered by an electric motor through a series of mechanical components, typically two 4-bar linkages in series or parallel**.**

Why:

1. Used to remove rain,snow from a vehicles front windows

2. To ensure the driver's safety.

Where:

1. This project helps the users to achieve the clear path when there is a change of weather.

2. Used in four wheelers.

How:

The wiper system is controlled using rain sensor, temperature sensor and SMT32 microcontroller

High Level Requirements

ID	Description	Status
HLR_1	These systems detect droplets of rain on the windshield and automatically turn on and adjust the wiper system in accordance to the level of precipitation.	Implemented

ID	Description	Status
HLR_2	A windscreen wiper or windshield wiper is a device used to remove rain, snow, ice and dust from a windscreen or windshield.	Implemented
HLR_3	Quality and reliability wiper systems meet the highest technical requirements and are the basis for vehicles with sophisticated features.	Implemented
HLR_4	Almost all motor vehicle, including trains, aircraft and watercraft, are equipped with such wipers, which are usually an essential requirement.	Implemented
HLR_5	Our project brings forward this system to automate the wiper system having no need for manual intervention.	Implemented

Low Level Requirements

ID	**Description**	Status
LLR_1	A new mechatronic reversing system can now be used to clean the windshield with two wiper arms, whereby one wiper arm is powered directly and the other indirectly using a connection link.	Implemented
LLR_2	Wiper motor is automatically ON during the time of rainfall and dust	Implemented
LLR_3	Existing system manually used control stalk to activate wiper and the process of pulling up wiper is difficult to be handled.	Implemented
LLR_4	Lower level parsing. Under the hood, the Requirement class does most of the heavy lifting. class requirements.	Implemented
LLR_5	These systems detect droplets of rain on the windshield and automatically turn on and adjust the wiper system,similerly the dust	Implemented

ID	Description	Status
	particals detected and wiped off.	

Design

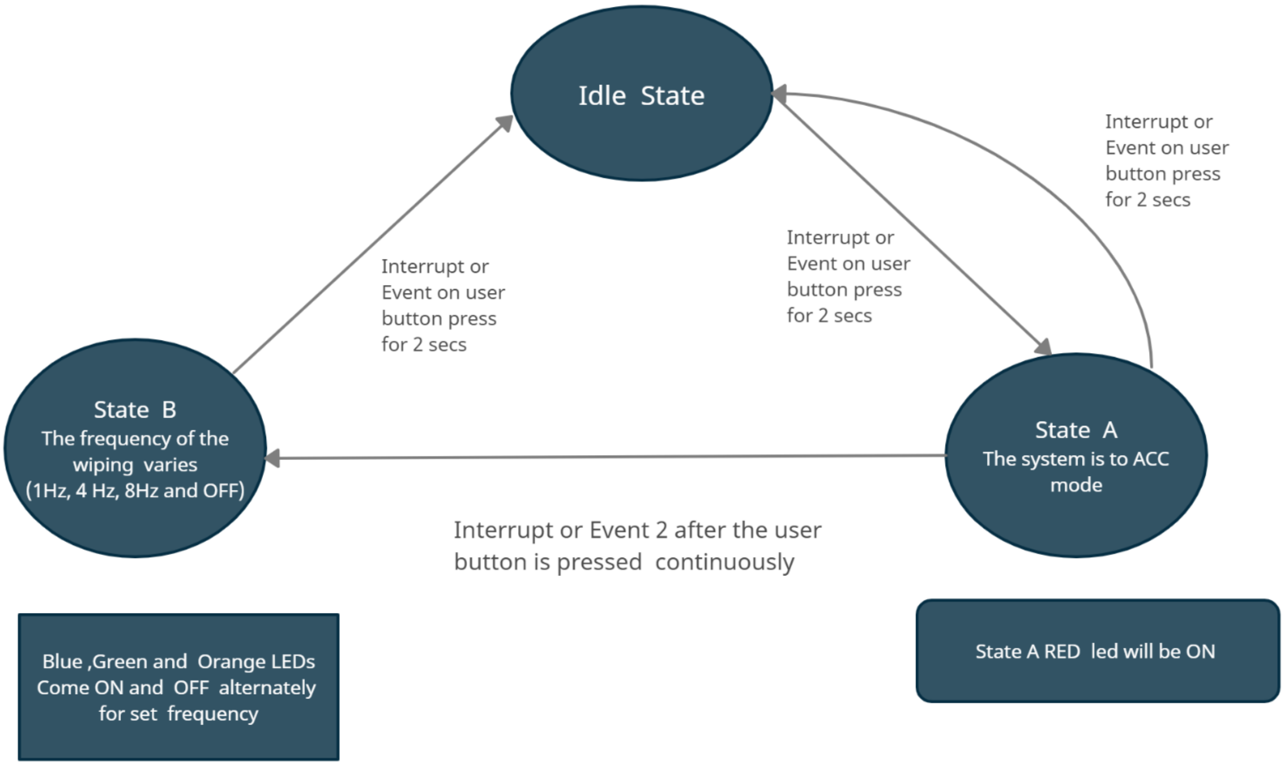

Idle State
Interrupt or Event on user button press for 2 secs
Interrupt or Event on user button press for 2 secs
Interrupt or Event on user button press for 2 secs
State B
The frequency of the wiping varies
(1Hz, 4 Hz, 8Hz and OFF)
State A
The system is to ACC mode
Interrupt or Event 2 after the user button is pressed continuously
Blue ,Green and Orange LEDs Come ON and OFF alternately for set frequency
State A RED led will be ON

Figure 13 Structure Diagram

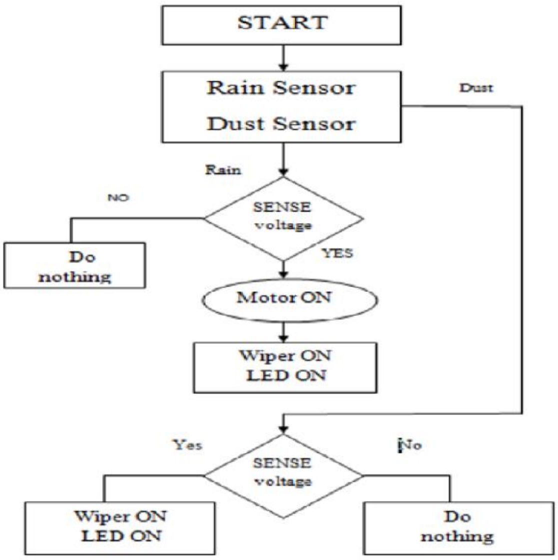


Figure 14 Behavior Diagram

Test Plan

High Level Test Plan

ID	Description	Output	Type of Test
HLTP_1	Press and hold the button to put the Ignition key position in ACC mode	System Enters ACC State	Requirement Based
HLTP_2	Different wiper frequencies to be set (1Hz, 4Hz & 8Hz)	Responds Based on Input	Requirement Based
HLTP_3	Hold the button to put the system in Idle state	Enters Idle State	Requirement Based

Low Level Test Plan

ID	Description	Output	HLTP ID	Type of Test
LLTP_1	Hold the button for 2 sec to bring the ignition key position at ACC mode	Red LED-ON	HLTP_1	Requirement Based
LLTP_2	Hold the button for 2 sec to go back to the Idle state	Red LED-OFF	HLTP_1, HLTP_3	Requirement Based
LLTP_3	Press the button one time to set frequency to 1Hz	Blue LED-ON	HLTP_2	Requirement Based
LLTP_4	Press the button second time to set frequency to 4Hz	Green LED-ON	HLTP_2	Requirement Based
LLTP_5	Press the button third time to set frequency to 8Hz	Orange LED-ON	HLTP_2	Requirement Based
LLTP_6	Press the button fourth time to turn OFF the wiper action	All LED OFF except Red	HLTP_2	Requirement Based
LLTP_7	Hold the button for 2 sec to bring ignition key position at Lock state	Red LED-OFF	HLTP_3	Requirement Based

Implementation and Summary

Git Link:

Link: https://github.com/GENESIS-2022/MasteringMCU-Team77.git

Individual Contribution and Highlights

1. Wiper System using C Programming
2. Source code management using GitHub

Role in Project Team

1. Programmer: Done Programming for Wiper System
2. Integrator: Integrated all the codes
3. Tester: Writing Testcases and testing the integrated code

Miniproject 7 – Mercedes Benz Project [Team]

Modules

1. Automotive Systems
2. Git

Requirements

In this Mercedes Benz project we have taken following features and I have contributed to Wiper Control System Feature

1. Power Windows
2. Door Locking System
3. Wiper Control System

• High level Requirements

ID	Description
HLR1	Obstacle sensor: when obstacle detected the operation will be delayed for 10 seconds
HLR2	Both driver and passengers control: operated with the first received command. (either by driver or passengers)
HLR3	Power windows using Key: Pressing unlock button for 5 seconds opens windows of the car

• Low level requirements

ID	Description
LLR1	Semi controlled - manually/automatic

Design

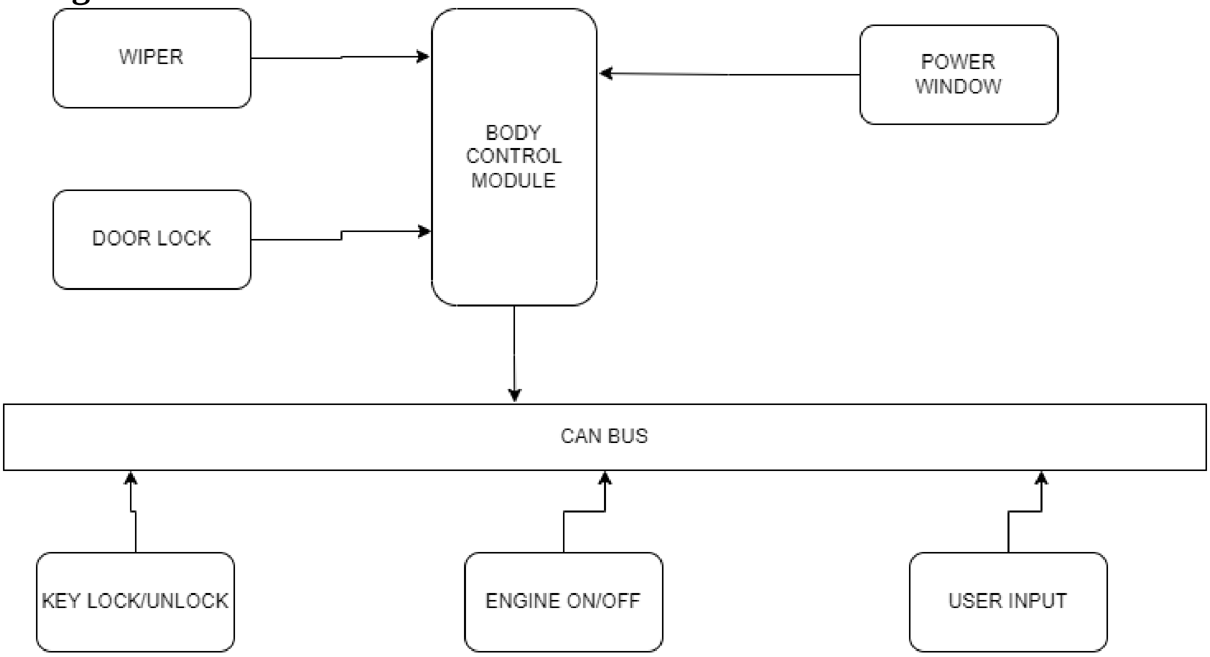


Figure 15 Structure Diagram

Implementation and Summary

Git Link:

Link: https://github.com/tirupathi29/Automotive_Mercedes_Project.git

Individual Contribution and Highlights

1. Wiper System Case Study
2. Module design for a body control

Role in Project Team

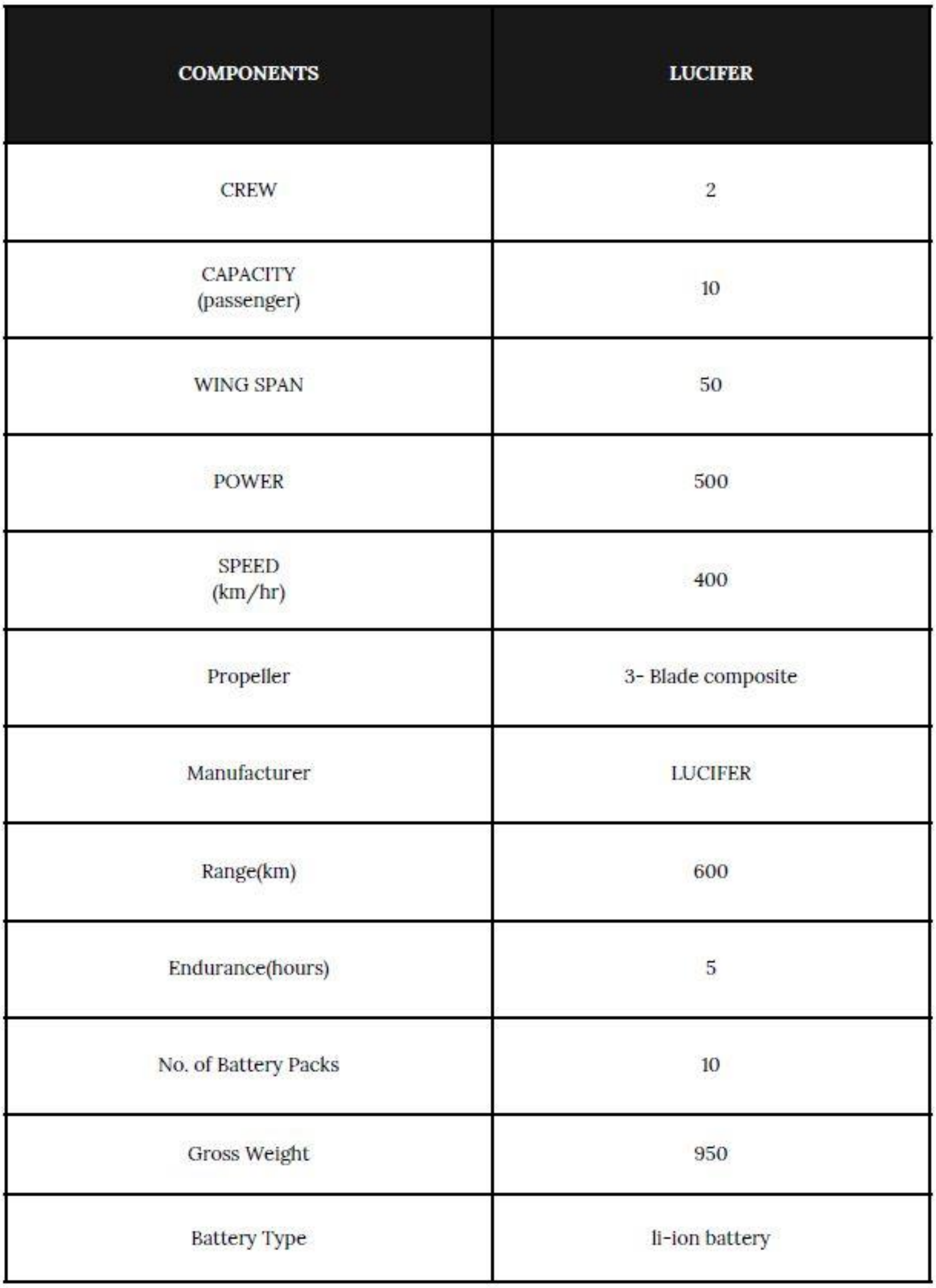
1. Designer: Done Designing for Project
2. Researcher: Done case study for wiper System

Miniproject 8 – EV Small Aircraft [Team]

Modules

1. Matlab
2. Matlab Script

Requirements



COMPONENTS	LUCIFER
CREW	2
CAPACITY (passenger)	10
WING SPAN	50
POWER	500
SPEED (km/hr)	400
Propeller	3- Blade composite
Manufacturer	LUCIFER
Range(km)	600
Endurance(hours)	5
No. of Battery Packs	10
Gross Weight	950
Battery Type	li-ion battery

Implementation and Summary

Submission: Submitted in GEA Learn

Individual Contribution and Highlights

1. Done in Matlab Script

Role in Project Team

1. Done Matlab scripting for EV Bike
2. Researcher: Done case study for EV Bike

Miniproject 9 – Wiper Control System [Individual]

Modules

1. Autosar
2. Git

Requirements

High Level Requirements

ID	Description	Status
HLR_1	Press and hold the button to put the ignition key position in ACC mode	Implemented
HLR_2	Different wiper frequencies to be set (1Hz, 4Hz, & 8Hz)	Implemented
HLR_3	Hold the button to put the system in idle state	Implemented

Low Level Requirements

ID	Description	HLTP ID	Status
LLR_1	Hold the button for 2 sec to bring the ignition key position at ACC mode	HLR_1	Implemented
LLR_2	Hold the button for 2 sec to go back to the idle state	HLR_1, HLR_3	Implemented
LLR_3	Press the button one time to set the frequency to 1Hz	HLR_2	Implemented
LLR_4	Press the button second time to set the frequency to 4Hz	HLR_2	Implemented
LLR_5	Press the button third time to set the frequency to 8Hz	HLR_2	Implemented
LLR_6	Press the button four time to turn OFF the wiper action	HLR_2	Implemented
LLR_7	Hold the button for 2 sec to bring ignition key position at lock state	HLR_3	Implemented

Design

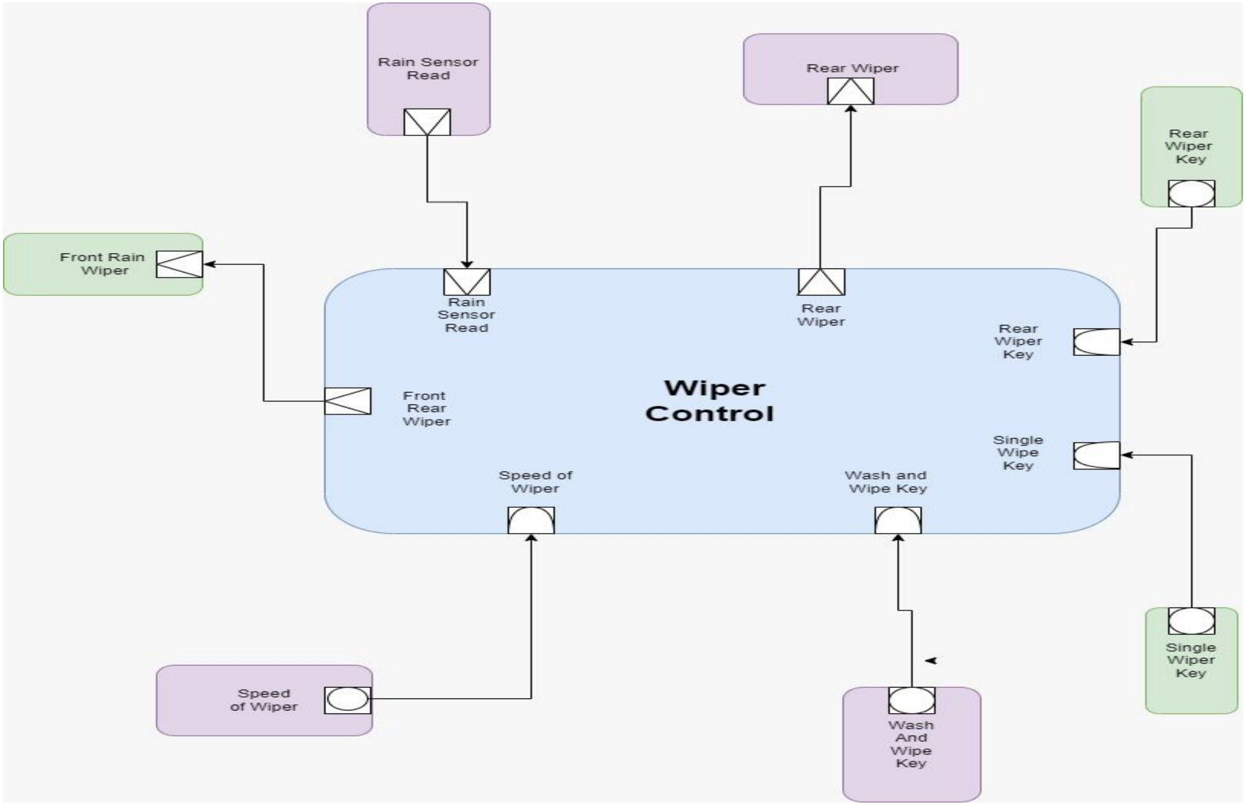


Figure 16 VFB Diagram

Implementation and Summary

Git Link:

Link: https://github.com/dileep40020532/WiperControlSystem_40020532_DPS.git

Individual Contribution and Highlights

1. Wiper Cotrol Case Study
2. Atomic Sw Component
3. SWC Internal Behavior
4. SWC Implementation