

GENESIS-Learning Outcome & Mini-project Summary Report

Details

Ver. Rel. No.	Release Date	Prepared By	Reviewed By	To Be Approved	Remarks/Revision Details
1.0	17/02/2022	MOHAMMAD ASIF(40020574)			

Contents

Miniproject – 1: Science Quiz Game[Individual].....	6
Modules:	6
Requirements	6
High Level Requirements: -	6
Low Level Requirements: -	7
SWOT Analysis	7
Design:-	8
Test Plan: -	9
High Level Test Plan	10
Low Level Test Plan	10
Implementation and Summary	10
Git Link:	10
Git Dashboard	10
Miniproject 2 – Heat Control System[Individual]	10
Modules	10
Requirements.....	11
High Level Requirements	11
Low Level Requirements	11
Design.....	13
Block Diagram	13
Output.....	15
Implementation and Summary	15
Git Link:	15
Miniproject 3 – Patient Vaccination Information management system [Team].....	16
Modules	16
Requirements.....	16
High Level Requirements:-	17
Low Level Requirements:-.....	17
Design.....	19
Test Plan.....	20
High Level Test Plan	21
Low Level Test Plan	24
Implementation and Summary	25
Git Link:	25

Individual Contribution and Highlights	25
Summary	25
Miniproject 4 – Attendance Automation [Team]	26
Modules	26
Requirements.....	26
High Level Requirements	27
Low Level Requirements	27
Test Plan	28
High Level Test Plan	28
Low Level Test Plan	29
Implementation and Summary	30
Git Link:	30
Git Dashboard	30
Miniproject 5 – Tesla Project [Team]	30
Modules	30
Requirements.....	30
High Level Requirements	31
Low Level Requirements	31
Design.....	32
Miniproject 6 – Wiper Control System[Team]	32
Modules	32
Requirements.....	32
Windshield wipers are controlled by the stalk on the right side of your steering wheel. Simply moving the stalk down will turn your windshield wipers on. Moving the stalk down will turn your wipers on.	33
High Level Requirements	33
Low Level Requirements	33
Design.....	34
Test Plan.....	36
High Level Test Plan	36
Low Level Test Plan	37
Implementation and Summary	37
Git Link:	37
Individual Contribution and Highlights	37
Miniproject 7 – TESLA Project [Team]	38
Modules	38

Requirements.....	38
Design.....	40
Implementation and Summary	40
Git Link:	40
Individual Contribution and Highlights	40
Miniproject 8 – EV Bike [Team]	40
Modules	40
Implementation and Summary	43
Individual Contribution and Highlights	43
Miniproject 9 – Wiper ControlSystem [Individual].....	44
Modules	44
Requirements.....	44
High Level Requirements	44
Low Level Requirements.....	44
Design.....	45
Implementation and Summary	45
Git Link:	45
Individual Contribution and Highlights	46

Miniproject – 1: Science Quiz Game[Individual]

Modules:

1. C Programming
2. Git

Requirements

4W's and 1'H :-

WHO

Every person with electronic devices need games nowadays for refreshment and relaxing. So, anyone can play this game.

WHY

A game which can be used to check or enhance our science knowledge because difficulty level will be increased gradually.

WHEN

One can play this game whenever he/she is free and want some refreshment but also they want to do some productive work to gain knowledge.

WHERE

It can be played by everyone on this game project .

HOW:One can simply enter his/her name and start playing .He/she can check score also

High Level Requirements: -

ID Description Status

HR01: Main Menu Implemented

HR02: Can be played by 1 player Implemented

HR03: Random Questioning Implemented

HR04: Graphical User Interface Future

Low Level Requirements: -

ID: LR01

DESCRIPTION: Main menu should consist 4 options asking for

1: New game

2: View highest score

3: Help for the Game

4: Quit option

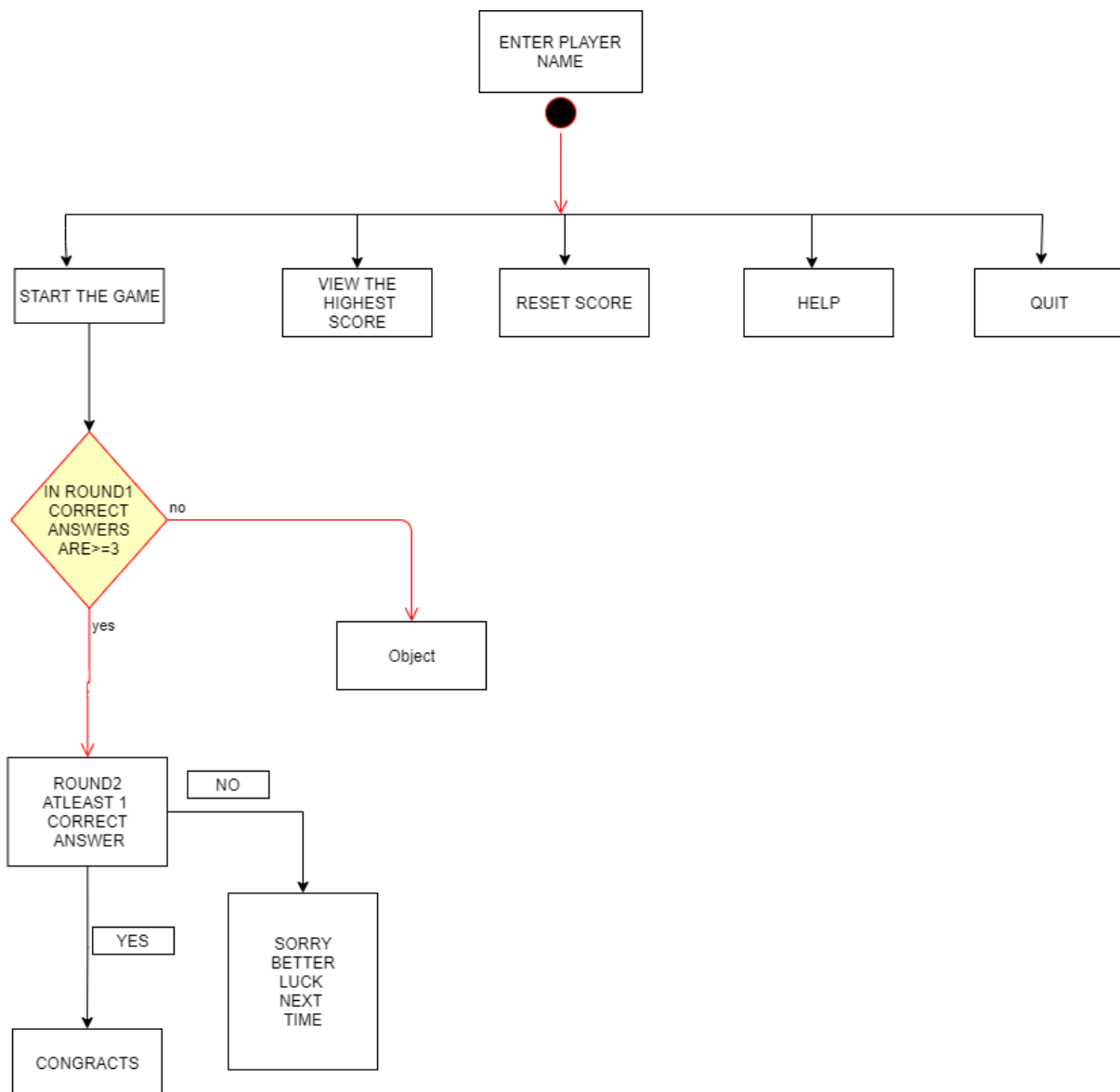
SWOT Analysis

- **STRENGTH:** Easy to use application. Random questions are present and difficulty level of questions will be increased step by step and one can check and reset score also.
- **WEAKNESS:** Lack of graphical user interface(GUI) more than 1 players cannot play.
- **OPPORTUNITIES:** This game is in demand Graphical user interface (GUI) can be implemented and leaderboard will be present then anyone can compare his/knowledge to others it will be a very interesting game people will love to play.
- **THREAT:** Best applications are available using different technologies and competition is high

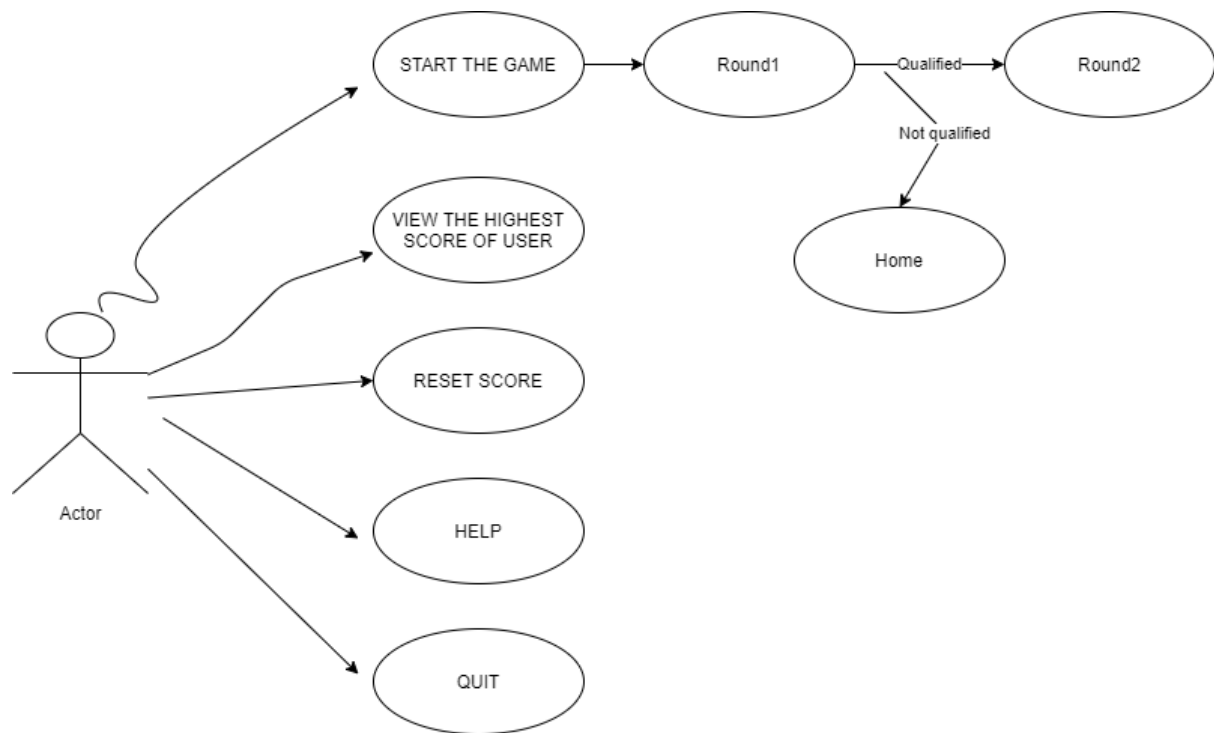
Design:-

Behavioural Diagram

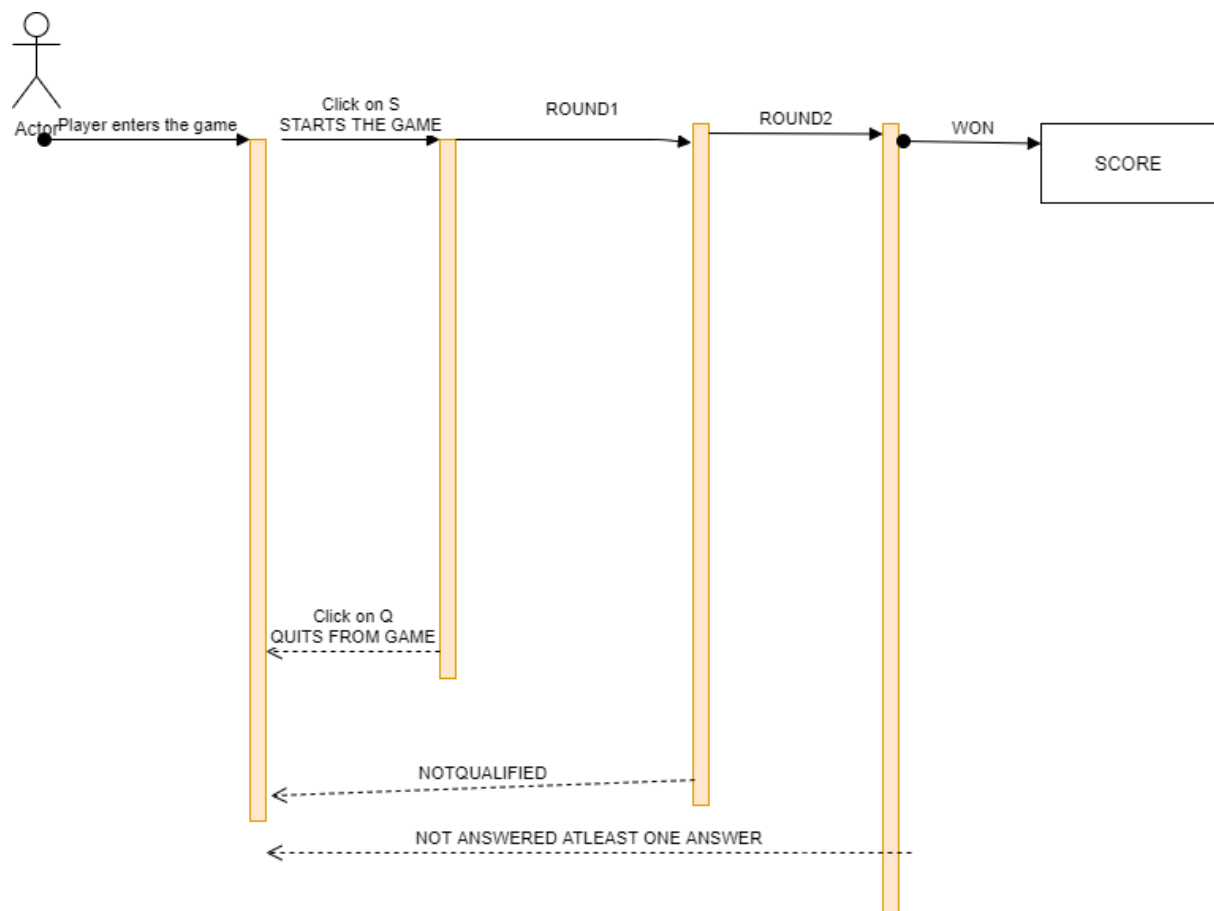
1.Activity Diagram



User Case Diagram



Structural Diagram



Test Plan: -

High Level Test Plan

Test Id	Description	Expected input	Expected Output	Actual Output	Type Of Test
Asif07	User Interface	Character Y	Enter The Answer	Pass	Requirement

Low Level Test Plan

Test_Id	Description	Expected_In put	Expected_ output	Actual_ Output	Type_of_Test
LP01	User View all the rules of the game	Character Y	View Rules and Exits	Pass	Requirement
LP02	user has an option to exit	Character N	Exits	Pass	Requirement
LP03	User Can View the score	Character Y	Open Results	Pass	Requirement

Implementation and Summary

Git Link:

Link: https://github.com/Asif78-00/M1_game_Science-Quiz.git

Git Dashboard

codefactor	A	Code Quality Score	80	Code Grade	B	code quality	B
------------	---	--------------------	----	------------	---	--------------	---

Miniproject 2 – Heat Control System[Individual]

Modules

1. C Programming
2. Embedded System

3. Git

Requirements**4W's and 1 H's****What**

Heating control system in a vehicle

Where

Used in almost all of the passenger vehicles

When

When temperature is low

Why

To maintain body temperature

How

By using sensors

High Level Requirements

ID	Description
HLR1	Microcontroller unit
HLR2	Switches
HLR3	Temperature sensor
HLR4	Heater
HLR5	Display CDD CRO

Low Level Requirements

ID	Description	HLR ID
LLR1	ATMega328	HLR1

ID	Description	HLR ID
LLR2	ADCand Potentiometer	HLR3
LLR3	Thermo electric module	HLR4
LLR4	LCD and LED, PWM	HLR5

SWOT Analysis

Strengths

Robust in nature

low cost

Easily accessible by the any person

High efficiency

Weakness

This system can be used at low to moderate temperature.

Opportunities

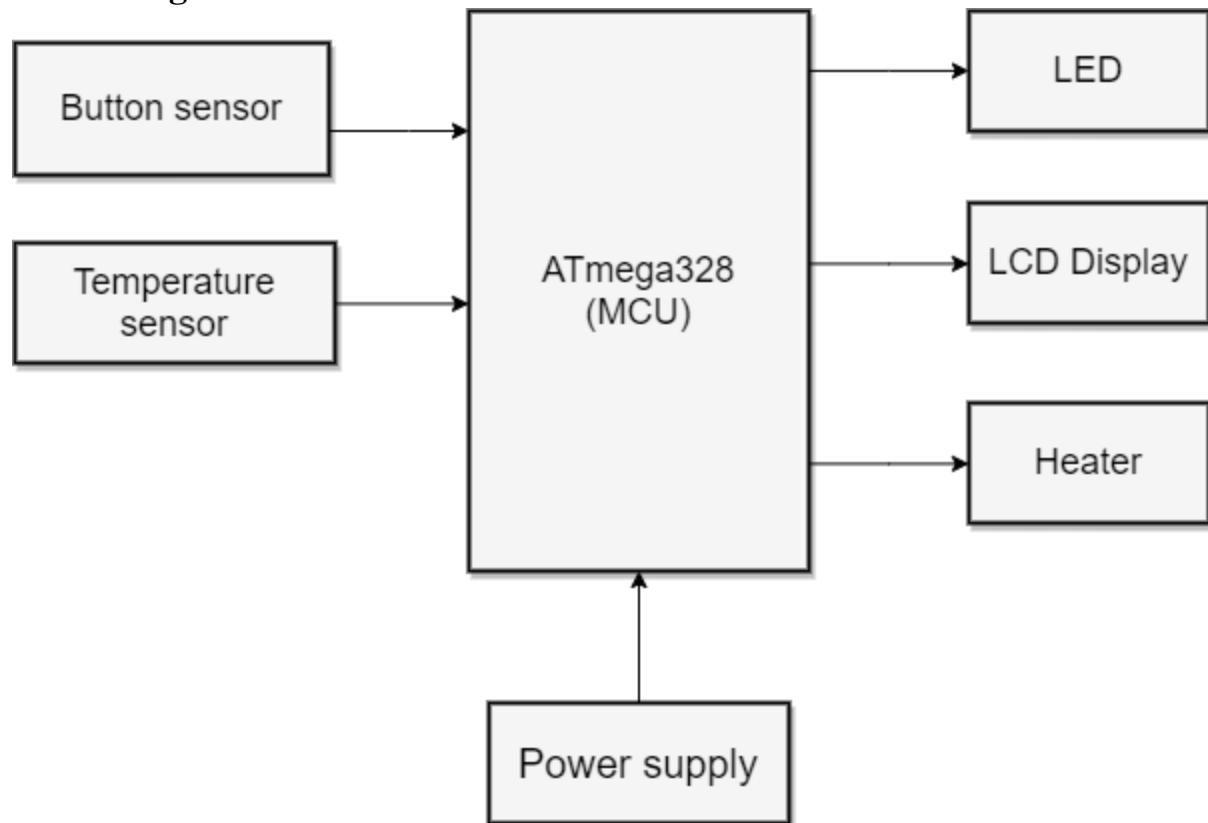
This system can be expanded by adding few more features depending on the user requirement.

Threats

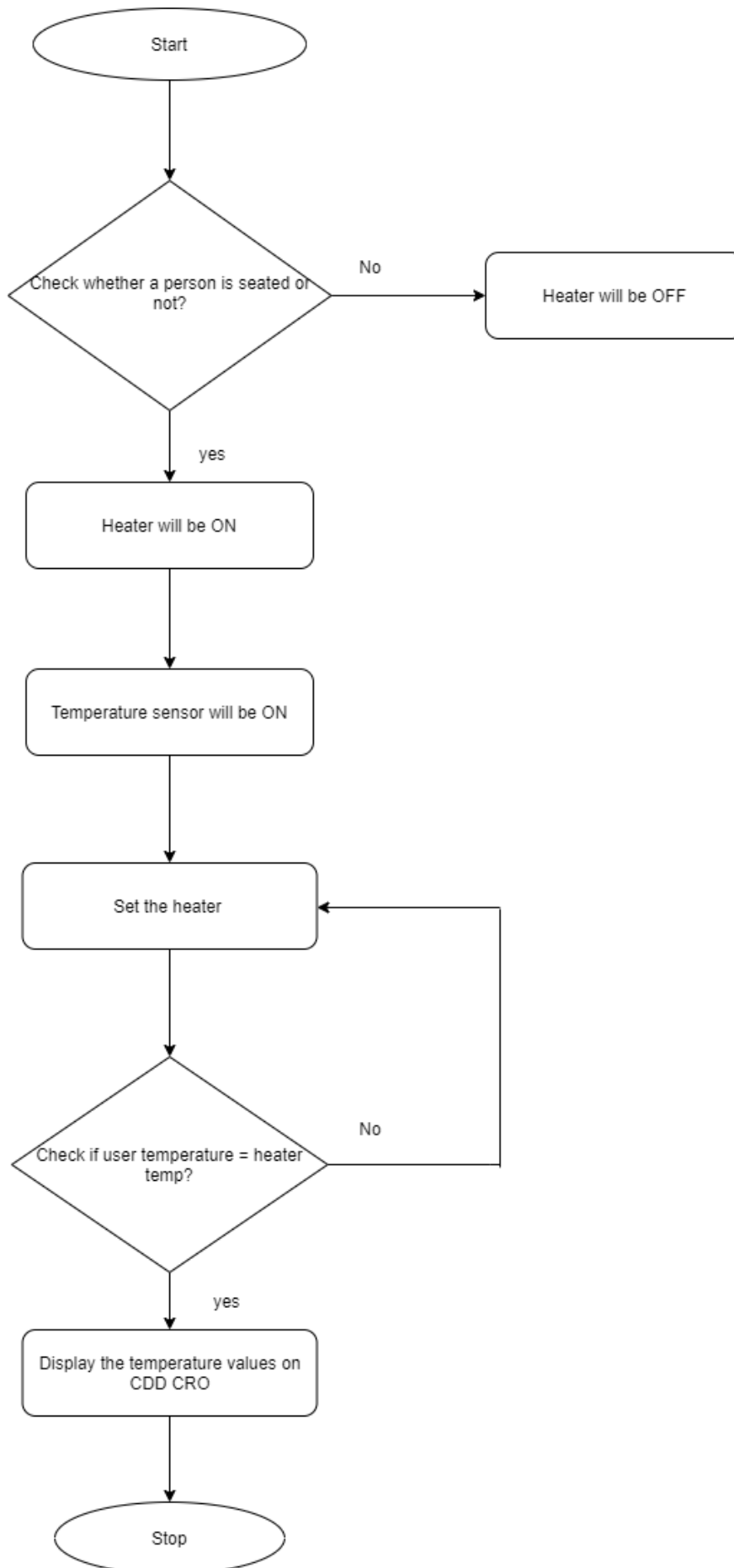
This system cannot be used for very high temperature.

Design

Block Diagram



Flow Chart



Output



Implementation and Summary

Git Link:

Link: https://github.com/Asif78-00/M2_embedded_HeatControlSyst.git

Miniproject 3 – Patient Vaccination Information management system [Team]

Modules

1. SDLC
2. Git

Requirements

4W's and 1 H's

Who:

COVID vaccinations are providing to the general public by all government and private Hospitals, small clinics and Dispensaries in the city.

What:

Storing all the data manually from people getting vaccinated is time consuming. The information is registered through the application as the population is huge, and manual entries for all of them is impossible. With the application, it is going to be easy even for finding out the status of a particular person.

When:

Currently, all people are advised to take the vaccination ever since the Vaccination drive started in February 2021.

Where:

Vaccination Management System, a Medical Solution for Clinics, Hospitals & Doctors.

Allow your patients to book appointments, request video consultations & e-consults.

Hospitals can now remind patients about vaccination updates with the Vaccination Management System.

This management system will be very useful in all the hospitals.

How:

On getting due dose of COVID-19 vaccine, the beneficiary will receive SMS on their registered mobile number.

After all doses of vaccine are administered, a QR code based certificate will also be sent to the registered mobile number of the beneficiary

High Level Requirements:-

ID	Description	Category	Status
HR01	End user can read the patient's record	Technical	Implemented
HR02	End user can add new patient's record	Technical	Implemented
HR03	End user can save the patient's records in file	Technical	Implemented
HR04	Even failure occurs, the data will not be lost	Scenario	Implemented
HR05	End user can delete patient's record	Technical	Implemented
HR06	End user can read the data from file	Technical	Implemented
HR07	End user can update the patient's record	Technical	Implemented
HR08	Storage of data occurs while closing the system	Scenario	Implemented

Low Level Requirements:-

ID	Description	Category	Status
LLR01	Reading patient data should be in 2 ways.first being by searching by id of a patient. By printing all the records available	HLR01	Implemented
LLR02	New record shall be added by providing all the asked information.Id should be unique and	HLR02	Implemented

ID	Description	Category	Status
	validated. file or else patient record should not be accepted		
LLR03	User shall be able to save the files,if file already exists then file and should not overwrite it and if file does not exists then it should create a new file	HLR03	Implemented
LLR04	If opening the file fails, then the system shloud prompt the message "Unable to access file" and should not end the program execution	HLR04	Implemented
LLR05	User need to search by id for the pateint record to be deleted, if no such record is available then "No Record Found" Message should be displayed	HLR05,06	Implemented
LLR07	If opening the file fails, then the system shloud prompt the message "Unable to access file" and should not end the program execution	HLR07	Implemented
LLR08	When user Log off the system perform check and save data to file.If new data in inserted add it to file.If New data is not inserted do not add anything to file	HLR08	Implemented

SWOT analysis

Strength

Reduces manual work

Stores patient records automatically

Saves a lot of time

Records every single detail about vaccine stocks

Appointment scheduling

Weakness

Lack of experience and knowledge of using computer applications

Security issues in digital mode

Lack of time allowed for learning and making use of this software

Opportunities

It improves management facilities

It has a common user friendly interface having several modules

New features could be added in which for instance patient record from PMS can be sent to HMS

This automated functions of online management make productivity effective

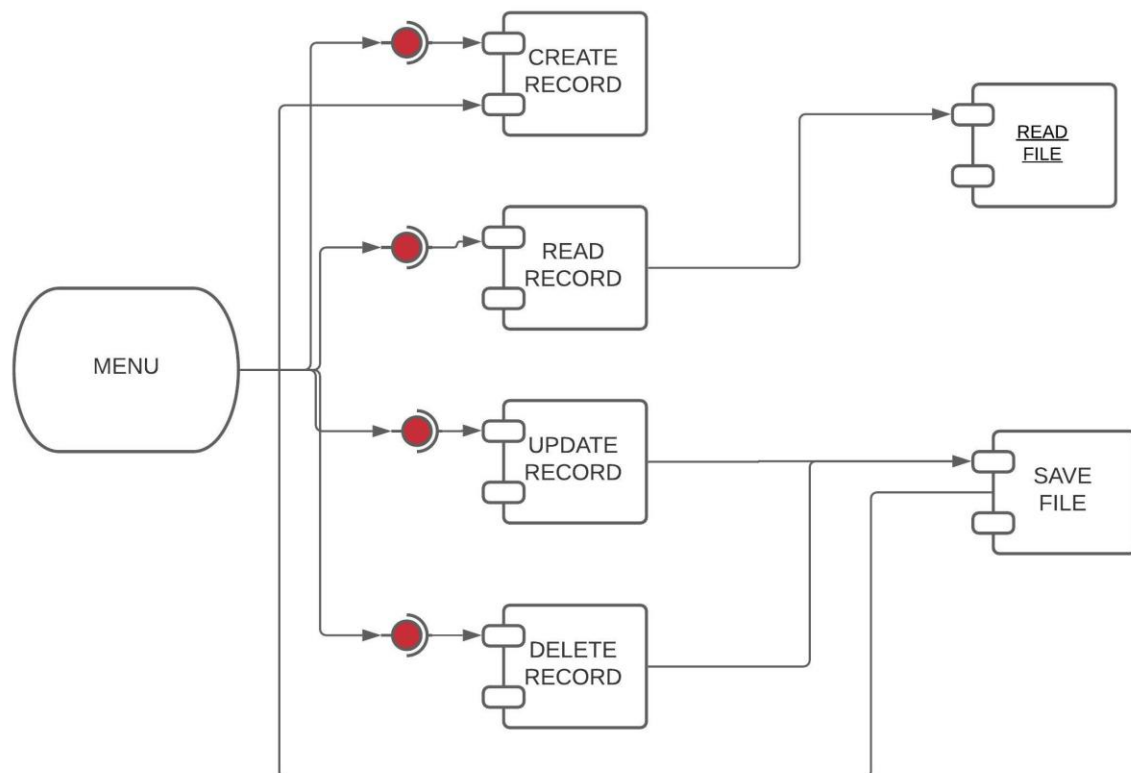
Threats

Any security related issues or news may pose a threat

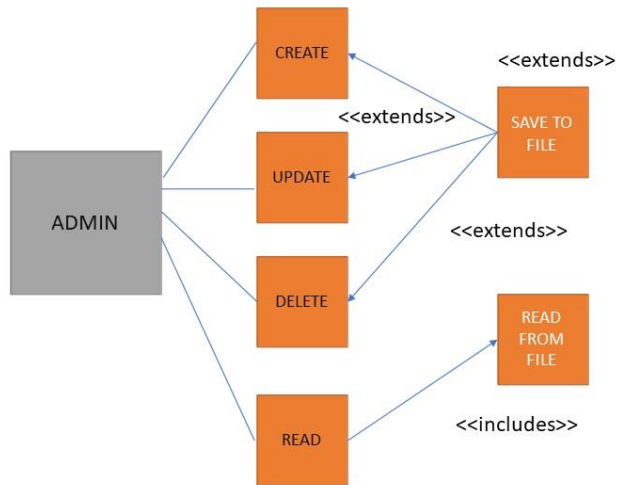
Tough competition

Design

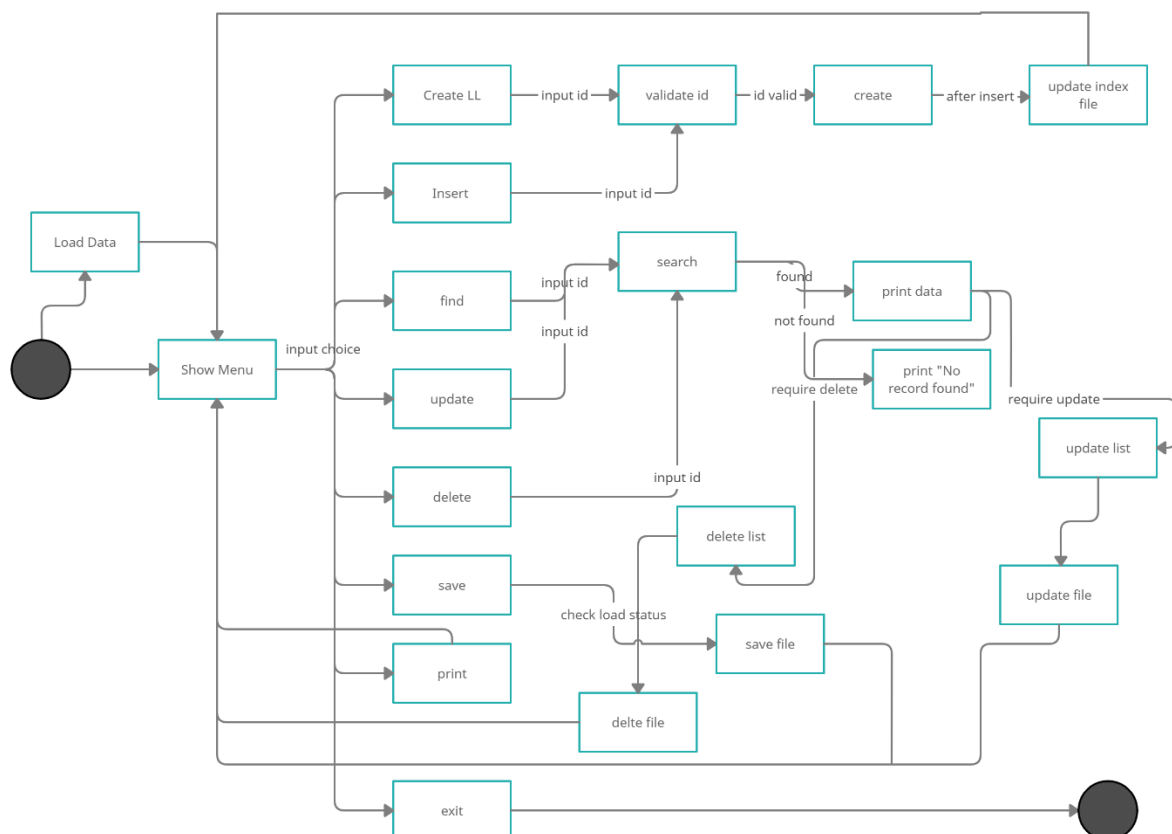
Component



User Case diagram



State Diagram



Test Plan

High Level Test Plan

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Type Of Test
H_01	Check if Linked List is created or not	(1). NULL Pointer (2). Unique id (3). First name (4). Last name (5). Height (6). Weight (7). Age (8). Insurance Status (9). vaccine code	Pointer to head node	PASS	Requirement based
H_01_01	Check LL initialized from a file	(1). Head Pointer (2). File Pointer	LL should be initialized from a file	PASS	Scenario/Technical
H_02	Check Insertion of new data in list	(1). Head Pointer (2). Unique id (3). First name (4). Last name (5). Height (6). Weight (7). Age (8). Insurance Status (9). vaccine code	SUCCESS	SUCCESS	Requirement based
H_02_01	Check if during insertion id gets stored in file	(1). File name (2). file mode (3). File Pointer	SUCCESS	SUCCESS	Requirement based
H_02_02	Check if during insertion no head exists	(1). File name (2). file mode (3). File Pointer	NO_HEAD_EXISTS	NO_HEAD_EXISTS	Technical
H_03	Check if records are displayed properly	(1). Head Pointer	SUCCESS	SUCCESS	Requirement based

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Type Of Test
H_03_01	Check if records in file are displayed properly	(1). File Pointer	SUCCESS	SUCCESS	Technical
H_04	Check if search by Patient id is working correct	(1). Head pointer (2). Id (3). Result Pointer (4). Flag	SUCCESS	SUCCESS	Requirement based
H_05	Check if record is updated properly	(1). Head pointer (2). Id (3). Field to be updated (4). Flag	SUCCESS	SUCCESS	Requirement based
H_05_01	Check if record is also updated in File	(1). File Pointer (2). Id	SUCCESS	SUCCESS	Technical
H_06	Deleting Record	(1). Head pointer (2). Id	Pointer to Head node	PASS	Requirement based
H_06_01	If record is only present in List, then delete from List	(1). Head pointer (2). Id	Pointer to head node	PASS	Technical
H_06_02	If record is only	(1). File pointers (2). Id	SUCCESS	SUCCESS	Technical

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Type Of Test
	present in File, then delete from File				
H_06_03	If record is deleted, then Index from Index File should also be deleted	(1). File pointers (2). Id	SUCCESS	SUCCESS	Technical
H_07	When required list can be stored in file	(1). File pointers (2). Head Pointer (3). Flag	SUCCESS	SUCCESS	Requirement based
H_08	When program shuts down records should be saved in File	(1). File pointers (2). Head Pointer	SUCCESS	SUCCESS	Requirement based
H_08_01	When program Shuts down all allocated Memory Locations	(1). Head Pointer	No Memory Leaks	FAIL	Technical

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Type Of Test
	should be freed				

Low Level Test Plan

Test ID	HLT ID	Description	Exp IN	Exp OUT	
L_01	H_02	During insertion check if ID is unique in INDEX.DAT file	(1). File Pointer (2). ID	SUCCESS	SUCCESS
L_01_02	H_02	Id f during insertion id already exists, do not allow insertion	(1). File Pointer (2). ID	ID_ALREADY_EXISTS	ID_ALREADY_EXISTS
L_03	H_02,H_01,H_06,H_07	Check if file is properly opened during program execution	(1). File Name (2). File Mode (3). File Pointer	SUCCESS	SUCCESS
L_04	H_07,H_08	if data is loaded from file during startup then writing of file should begin from the start of file	(1). File pointers (2). Head Pointer (3). Flag	SUCCESS	SUCCESS
L_05	H_06	If there is only one node in list then deletion from beginning algorithm should work	(1). File pointers	SUCCESS	SUCCESS

Test ID	HLT ID	Description	Exp IN	Exp OUT	
L_06	H_06	If first node is being deleted then deletion from beginning algorithm should work	(1). File pointers	SUCCESS	SUCC

Implementation and Summary

Git Link:

Link: https://github.com/GENESIS2021Q1/Applied_SDLC-Dec_Team_3.git

Individual Contribution and Highlights

Summary

1. Architecture (Sequence)
2. Implementation (src, unity)
3. Learning resources
4. Folder structure
5. Report

Miniproject 4 – Attendance Automation [Team]

Modules

1. Python
2. Git

Requirements

4W's and 1 H's

Who:

1. User may easily get attendance history of a particular student

Where:

1. An online attendance application also makes it easier for the HR team to manage employee's leave.
1. All you need is sign into the system to see how much leave you have left.
2. Attendance software also facilitates the management of all matters related to attendance

What

1. An automatic attendance system is an educational ERP system that records the student's attendance in an institution.
2. Unlike the conventional attendance system, the automatic attendance software enables the faculty to record, store, and monitor students' attendance history & manage the classroom efficiently.

When:

1. Attendance management systems have been in existence for as long as one can remember.
2. Although the tools and processes we use to monitor them have changed, the crux of the concept remains the same.
3. In simple terms, attendance management is the process of keeping track of all the employees in an organisation, monitoring their work times, having all leave requests and calendars collated in one place for easy management, and tracking daily expenses.

How:

1. The system helps the faculty to easily find out defaulters in a single click.

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
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

ID	Feature	High Level ID	Status
LLR_10	Multiple files can be generated with different inputs	HLR_05	Implemented

SWOT Analysis

Strength

Accuracy

Increased Productivity

Records all the data automatically

Provides accurate results

Easy to use

Weakness

Is expensive

Requires maintenance

Needs power supply to work

May fail to comply with changing regulations

Opportunities

It has a common user friendly interface having several modules

This automated functions in attendance automation system make productivity effective

Threats

Tough competition

Any security related issues or news may pose a threat

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

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
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 get the Attendance Status	SUCCESS	SUCCESS	
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	SUCCESS	SUCCESS	Requirement Based





ID	HLTP ID	Description	Expected I/P	Actual O/P	Type Of Test
		existing sheets as it is dynamic			
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: [sunilkora31/oops_with_python_Attendance-Automation_team-15: Attendance Automation \(github.com\)](https://github.com/sunilkora31/oops_with_python_Attendance-Automation_team-15)

Git Dashboard

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

Miniproject 5 – Tesla Project [Team]

Modules

1. MATLAB
2. Git

Requirements

We have implemented following features

1. Auto-Lock Door Control System
2. Battery Monitoring System

3. Discharge Control
- 4. State of Charge(my part)**
5. Temperature Control System
6. Voltage Control
7. Warning Control
8. Charge rate or Discharge Control System

State Of Charge:

One feature of the BMS is to keep track of the state of charge (SOC) of the battery. The nominal capacity is given by the manufacturer and represents the maximum amount of charge that can be stored in the battery. In general, the SOC of a battery is defined as the ratio of its current capacity () to the nominal capacity ()

$$\text{SOC} = \frac{Q_{\text{rem}}}{Q_{\text{max}}} * 100\%$$

High Level Requirements

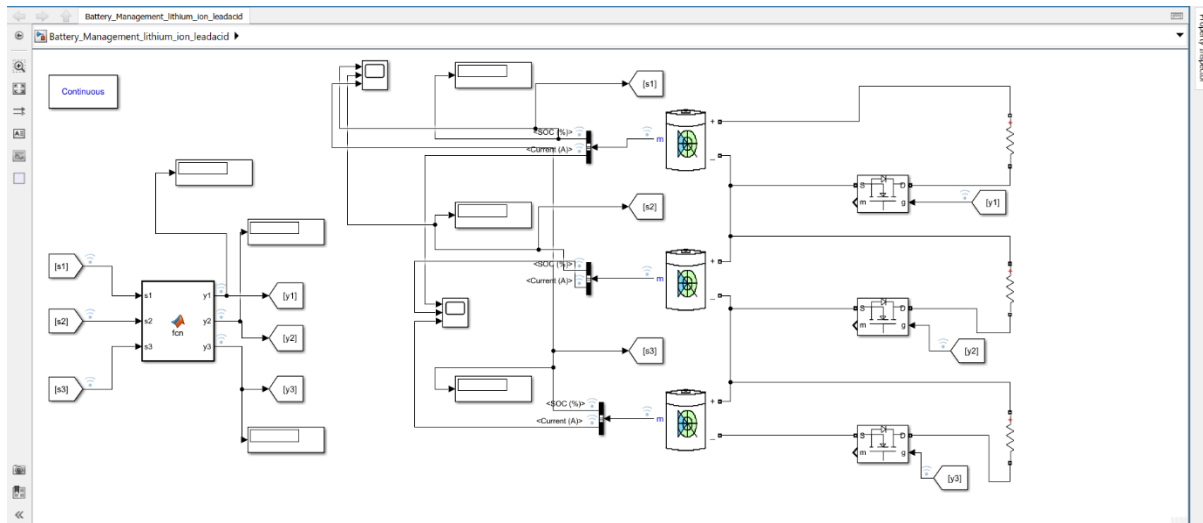
- 1.battery
- 2.display cell
- 3.indicator
- 4.sensor

Low Level Requirements

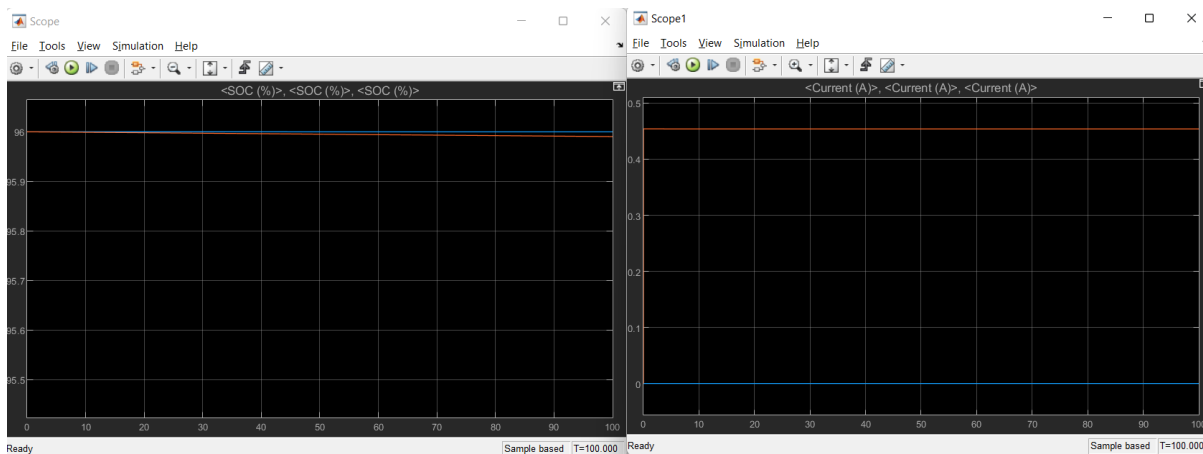
- 1.measurement of voltage
- 2.measurement of current
- 3.measurement of temperature
- 4.measurement of internal resistance

Design

This project was implemented using MATLAB.



Output



Miniproject 6 – Wiper Control System[Team]

Modules

1. C Programming
2. STM32

Requirements

4W's and 1'H

Who:

A wiper speed control system for an automotive wiper controls the operational speed of a wiper in accordance with rain conditions.

What:

Vehicles are now available with driver-programmable intelligent windscreen wipers that detect the presence and amount of rain using a rain sensor.

When:

Whenever the water hit a dedicated sensor that located on windscreen, it will send a signal to move on the wiper motor. Once water is not detected by sensor, the wiper will automatically stop. This will help the driver to give more concentration and reduce the car accident probability.

Where:

It is located underneath the dashboard, above the brake and accelerator pedal, and is responsible for the complete operation of the windshield wiper system.

How:

Windshield wipers are controlled by the stalk on the right side of your steering wheel. Simply moving the stalk down will turn your windshield wipers on. Moving the stalk down will turn your wipers on.

High Level Requirements

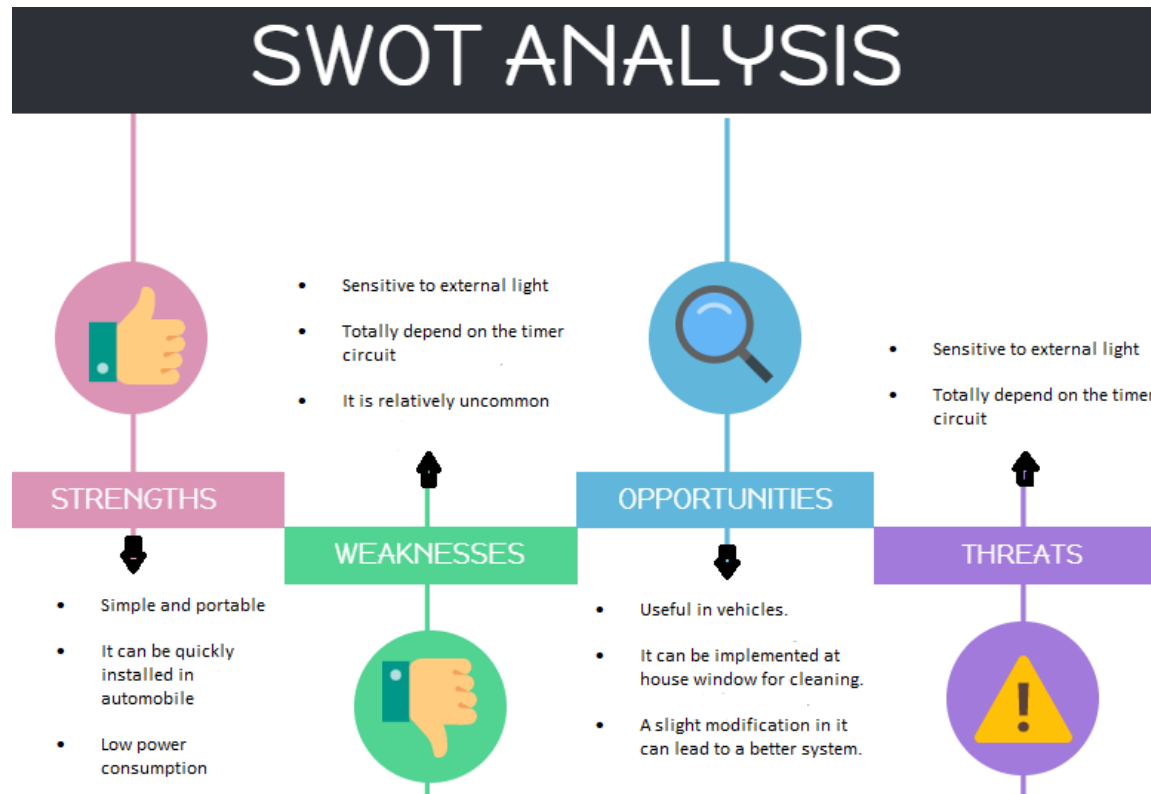
ID	Description	Present status
HLR1	The system detects droplets of rain on the windsheild and automatically turns ON	Implemented
HLR2	The sensor will activate the wiper motor	Implemented
HLR3	The stand drop off and rotate clockwise or anticlockwise when motor is activated	Implemented
HLR4	The wiper shall be able to operate for two minutes on a dey windscreen	Implemented

Low Level Requirements

LLR ID	HLR ID	Description	Status
LLR1	HLR1	Whenever the water hit a dedicated sensor that is loacted on windscreen,it will senda signal to turn on the wiper motor.	Implemented
LLR1	HLR1	The sensor will activate when the vehicle windscreen is wet due to moisture,raindrop or mud	Implemented
LLR1	HLR1	The sensor will stop once the water is not detected by the sensor	Implemented
LLR3	HLR1	The windscreen wiper feild shall cover not less than 94% of the primary area	Implemented

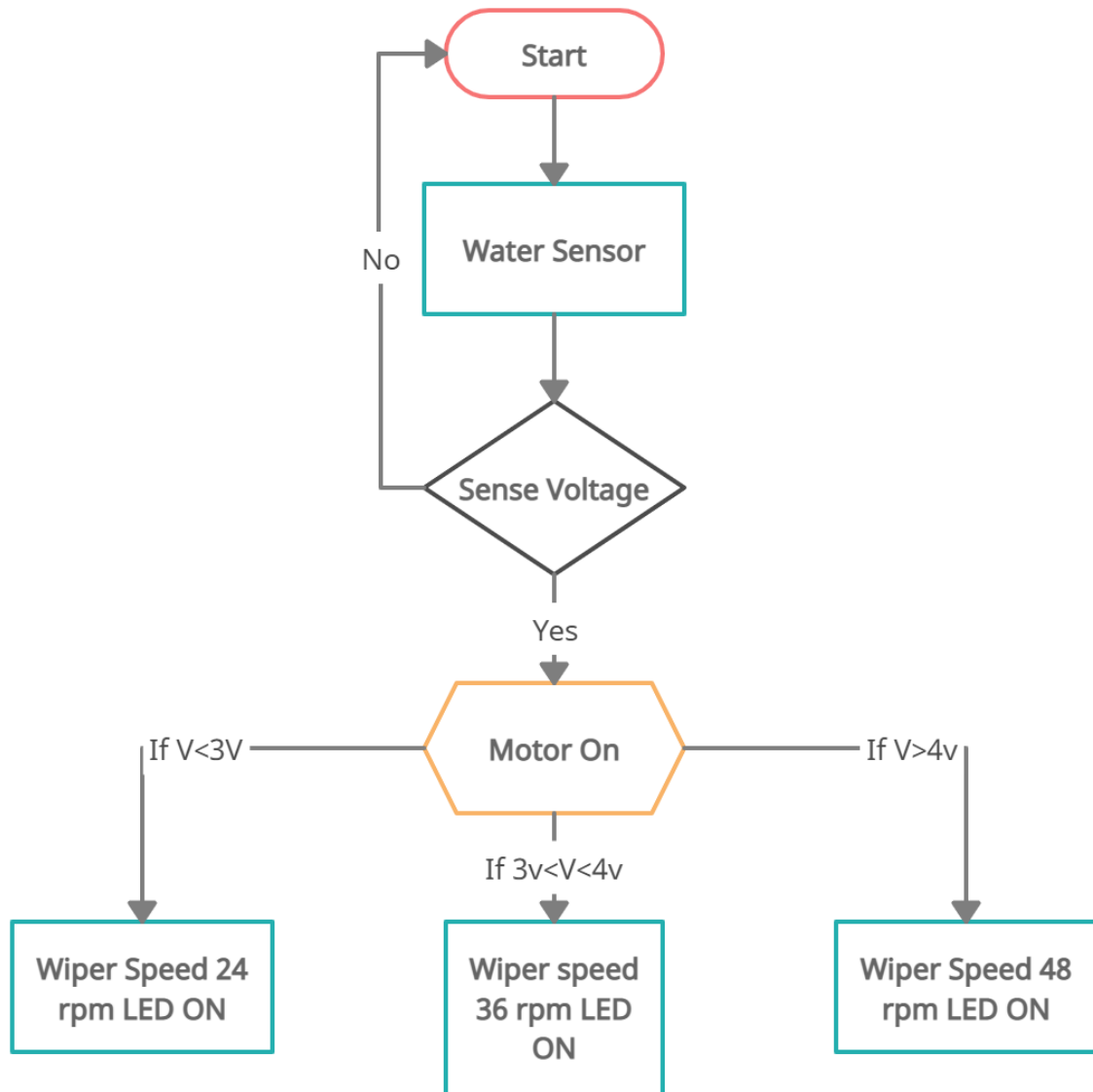
LLR3	HLR2	The windscreen wiper feild shall cover not less than 80% of the secondary area	Implemented
------	------	--	-------------

SWOT Analysis

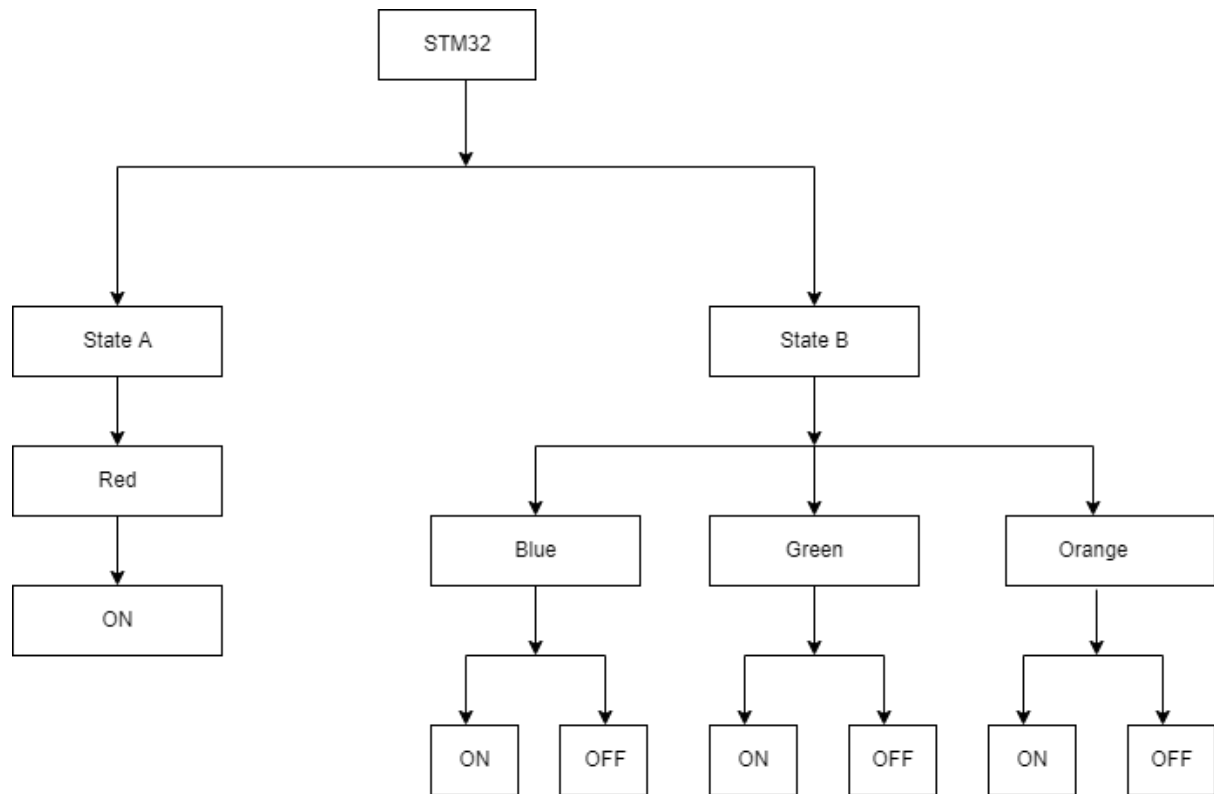


Design

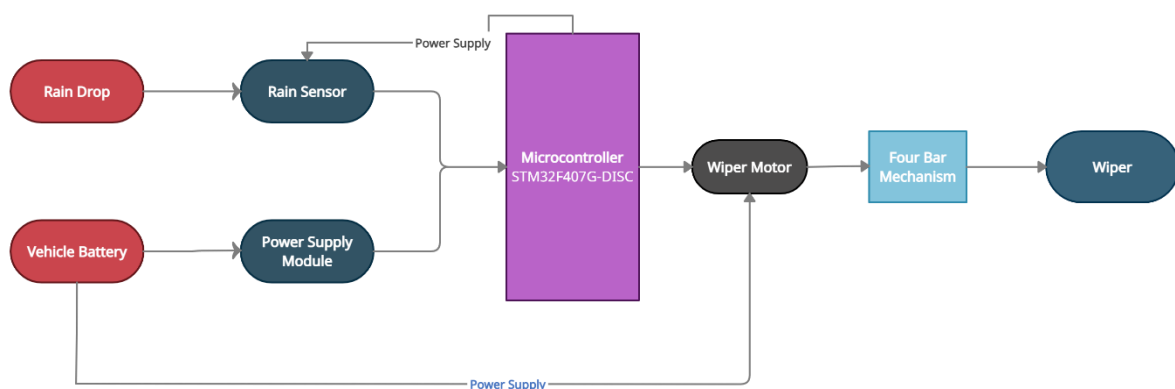
Behavioural Diagram:



Low Level Diagram



Structural Diagram:



Test Plan

High Level Test Plan

Test Id	Description	Status	Type Of Test
---------	-------------	--------	--------------

HLR1	Pressing the switch for the red led to glow	Pass	Technical
HLR2	Double pressing the switch for second led to glow	Pass	Technical
HLR3	Pressing the switch for 10 sec for reset	Pass	Technical

Low Level Test Plan

Test Id	Description	Status	Type Of Test
LLR1	Display of quam stm microcontroller	Pass	Technical
LLR2	Timer of 15 sec for blinking	Pass	Technical

Implementation and Summary

Git Link:

Link: <https://github.com/GENESIS-2022/MasteringMCU-Team10.git>

Individual Contribution and Highlights

1. Requirements
2. Implementation(inc)
3. Images & Videos (Output video)

Miniproject 7 – TESLA Project [Team]

Modules

1. Automotive Systems
2. Git

Requirements

In this TESLA project we have taken following features and I have contributed to Wiper Control System

1. Power Window
2. Mirror Control
3. Door Locking Control
4. Interior Lighting System
5. Wiper Control System

Requirements

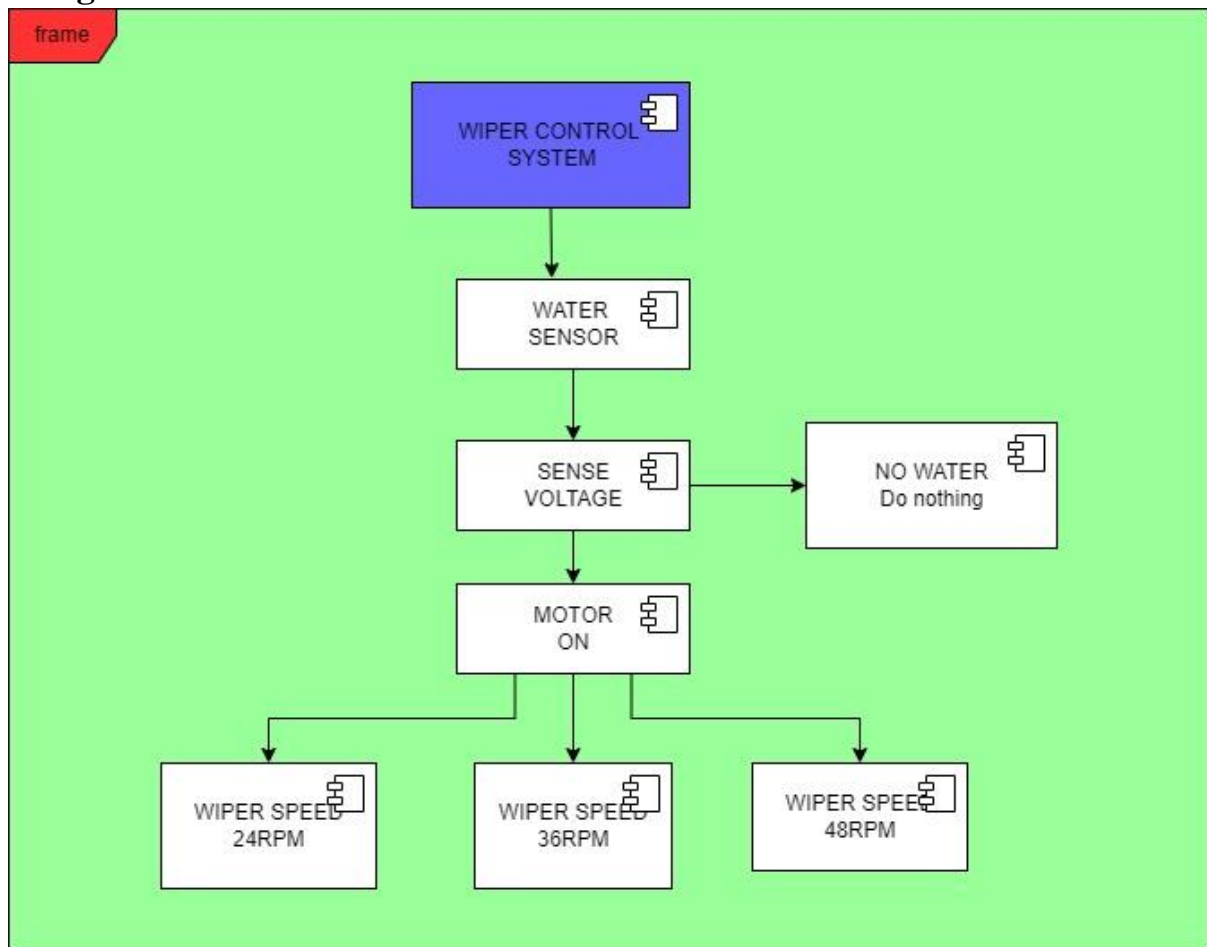
High Level Requirements

High level Requirement	Description
HLR1	Press & hold the button to put the ignition key position in ACC mode
HLR2	Different wiper freq to be set (1Hz, 4Hz & 8Hz)
HLR3	When rainfall starts, wipers should get start working automatically

Low Level Requirements

Low level Requirement	Description
LLR1	Hold the button for 2sec to bring the ignition key position at ACC mode
LLR2	Hold the button for 2sec to go back to the idle state
LLR3	Press the button one time to set freq to 1H

Design



Implementation and Summary

Git Link:

Link: https://github.com/Okayvvk/automotive_team_tesla.git

Individual Contribution and Highlights

1. Wiper Control System Case Study
2. Source code management using GitHub

Miniproject 8 – EV Bike [Team]

Modules

1. MATLAB
2. MATLAB Script

Project Team 2 EV Bike:

Authors:

- Ashwin Ramesh (PS No: 40020503)
- Balaji P (PS No: 40020526)
- Manikandan E (PS No: 40020545)
- Vettri Selvam (PS No: 40020491)
- Alrich Roshan S J (PS No: 40020515)
- Mohammad Asif (PS No: 40020574)
- Sharma V (PS No: 40021022)

Introduction:

In this project we are going to design and develop an EV Bike by comparing two existing models in the market and analysing the difference between them and compare them against our own design.

Requirements:

OLA S1 Pro Vs Ather 450X:

Motor Specifications:

Component	OLA S1 Pro	Ather 450X
Motor Type	Mid Drive IPM	PMS
Motor Power	8.5 kW	5.4 kW
Motor Torque	58 Nm	22 Nm
Top Speed	115 km/hr	80 km/hr
Acceleration from 0 to 60 km/hr	5 seconds	8.5 seconds

Battery Specifications:

Component	OLA S1 Pro	Ather 450X
Battery Type	Lithium-ion	Lithium-ion
Range	181 Km	70 Km
Battery Charging Time	6.30 Hours	5.15 Hours
Battery Capacity	3.97 kWh	2.4 kWh
No of Cells	35,750	21,700

Brake Specification:

Component	OLA S1 Pro	Ather 450X
Braking system	Combi Brake System	Combi Brake System
Front Brake	Disc	Disc
Front Disc Size	220 mm	200 mm
Rear Brake	Disc	Disc
Rear Disc Size	180 mm	190 mm
Front Calliper Type	3 Piston	3 Piston
Rear Calliper Type	Single Piston	Single Piston

Wheel Specifications:

Wheel Type	Alloy	Alloy
Front Wheel Size	12 inches	12 inches
Rear Wheel Size	12 inches	12 inches
Front Tyre Size	110/70 - R12	90/90 - R12
Rear Tyre Size	110/70 - R12	90/90 - R12
Tyre Type	Tubeless	Tubeless

Suspension Specifications:

Component	OLA S1 Pro	Ather 450X
Front Suspension	Single Fork	Telescopic
Rear Suspension	Mono shock	Mono shock

Dimensions & Chassis Specifications:

Component	OLA S1 Pro	Ather 450X
Kerb Weight	125 kg	108 kg
Overall Length	1,859 mm	1,800 mm
Overall Width	712 mm	700 mm
Overall Height	1,160 mm	1,250 mm
Wheelbase	1,359 mm	1,278 mm
Ground Clearance	165 mm	160 mm
Seat Height	792 mm	765 mm
Chassis Type	Tubular	Precision Machined Hybrid Chassis

Implementation and Summary

Submission: Submitted in GEA Learn

Individual Contribution and Highlights

1. Done in MATLAB Script

Miniproject 9 – Wiper ControlSystem [Individual]

Modules

1. Autosar
2. Git

Requirements

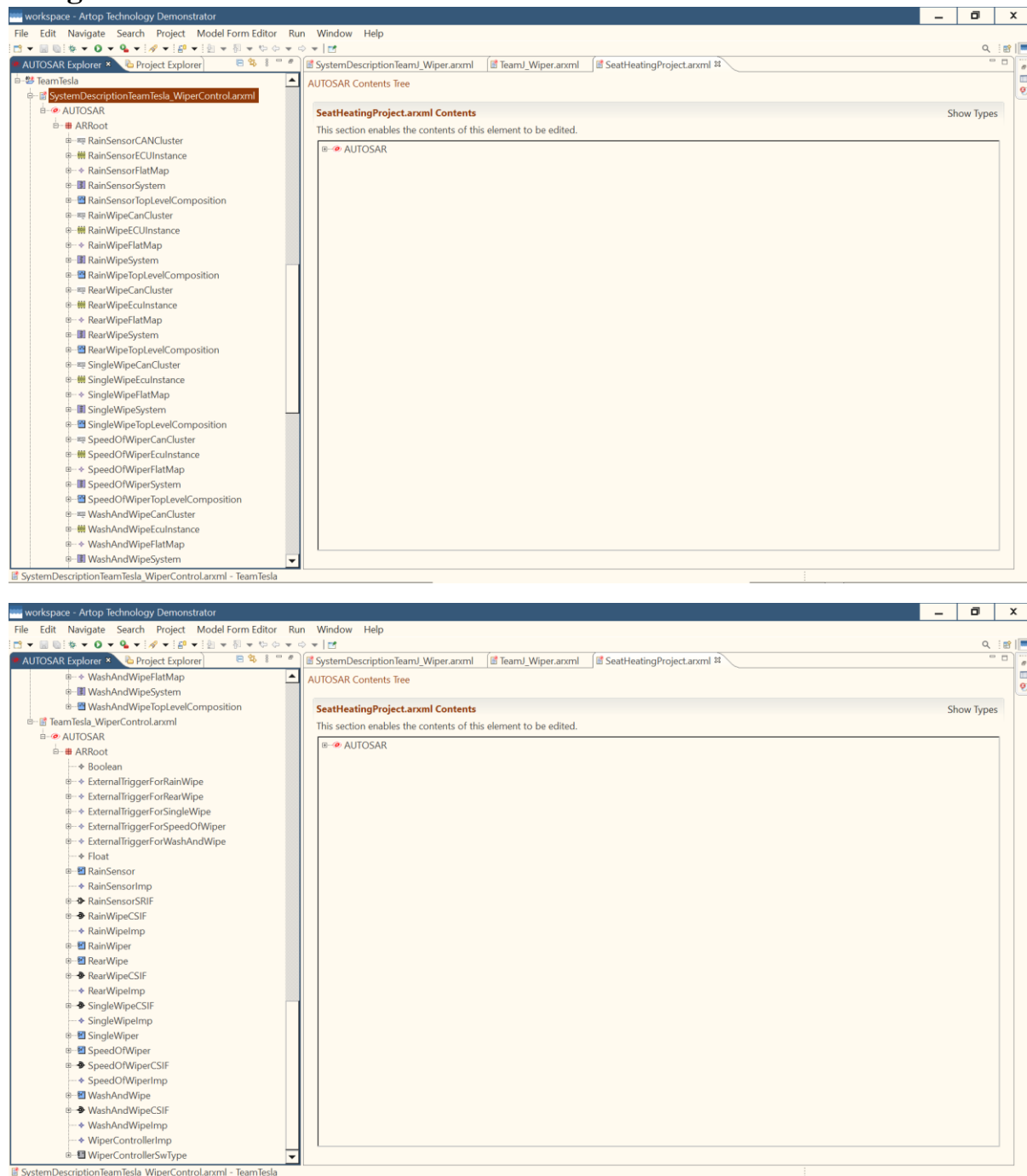
High Level Requirements

High level Requirement	Description
HLR1	Press & hold the button to put the ignition key position in ACC mode
HLR2	Different wiper freq to be set (1Hz, 4Hz & 8Hz)
HLR3	When rainfall starts, wipers should get start working automatically

Low Level Requirements

Low level Requirement	Description
LLR1	Hold the button for 2sec to bring the ignition key position at ACC mode
LLR2	Hold the button for 2sec to go back to the idle state
LLR3	Press the button one time to set freq to 1H

Design



Implementation and Summary

Git Link:

Link: [automotive_team_tesla/40020574 MohammadAsif DPS at main · Okayvvk/automotive_team_tesla \(github.com\)](https://github.com/Okayvvk/automotive_team_tesla)

Individual Contribution and Highlights

1. Wiper Control System Case Study
2. Source code management using GitHub
3. AtomicSwComponent
4. SWCInternalBehavior
5. SW Implementation