GENESIS - Learning Outcome & Mini-project Summary Report

L&T Technology Services

Details

Ver. Rel. No.	Release Date	Prepared By	Module Name	To Be Approved	Remarks/Revision Details
1.0	16/02/2022	Vettri Selvam M 40020491	C Programming On Multiple Platforms		
1.0	16/02/2022	Vettri Selvam M 40020491	Essesntials of Embedded Systems		
1.0	16/02/2022	Vettri Selvam M 40020491	Applied SDLC and Software Testing		
1.0	16/02/2022	Vettri Selvam M 40020491	Applied Model Based Design Module		
1.0	16/02/2022	Vettri Selvam M 40020491	OOPS with Python		
1.0	16/02/2022	Vettri Selvam M 40020491	Mastering Microcontrollers with Embedded Driver Development Module		
1.0	16/02/2022	Vettri Selvam M 40020491	Overview of Automotive Systems		
1.0	16/02/2022	Vettri Selvam M 40020491	Applied Control Systems and Vehicle Dynamics		
1.0	16/02/2022	Vettri Selvam M 40020491	Classic Autosar Basic to Intermediate		

Contents

List of Figures

Miniproject – 1: Employee Record System [Individual]

Modules:

1. C Programming
2. Git

Requirements

4W's and 1 H's

Who:

• All the organisation who has lot’s of employee in their different units and their different services.

What:

• Earlier Many Companies were inefficient due to prolonged time for entering N numbers of employee records manually. • This project gives the opportunity to enter all the required information and deploy into the system easily

When:

• As the business of the organisation increases number of employees increases . • With the increases of employee their data management require.

Where:

• This problem is surfaced in all the organisation in the world.

How:

• This project takes in all the input values and yields out the management parameter

High Level Requirements

ID	Description	Category	Status
HR01	The Application should allow user to make a choice between 1 to 5.	Textual	IMPLEMENTED
HR02	The Application should allow user to enter info.	Textual	IMPLEMENTED

ID	Description	Category	Status
HR03	The Application should allow user to list all the employee records.	Textual	IMPLEMENTED
HR04	The Application should allow user to modify the employee records.	Textual	IMPLEMENTED
HR05	The Application should allow user to delete employee records	Textual	IMPLEMENTED
HR06	The Application should allow user to exit from application.	Textual	IMPLEMENTED

Low Level Requirements

ID	Description	HLR ID	Status
LR01	The application will ask user to choose between 1 to 5	LR01	IMPLEMENTED
LR02	The application will ask user to enter info such as Enter name, Enter Salary, Enter Age.	LR02	IMPLEMENTED
LR03	The application will show all details of employees.	LR03	IMPLEMENTED
LR04	The application will ask user enter name (for modifying).	LR04	IMPLEMENTED
LR05	The application will ask user to enter name (for deleting).	LR05	IMPLEMENTED
LR06	The user can exit from application by choosing 5.	LR06	IMPLEMENTED

Design

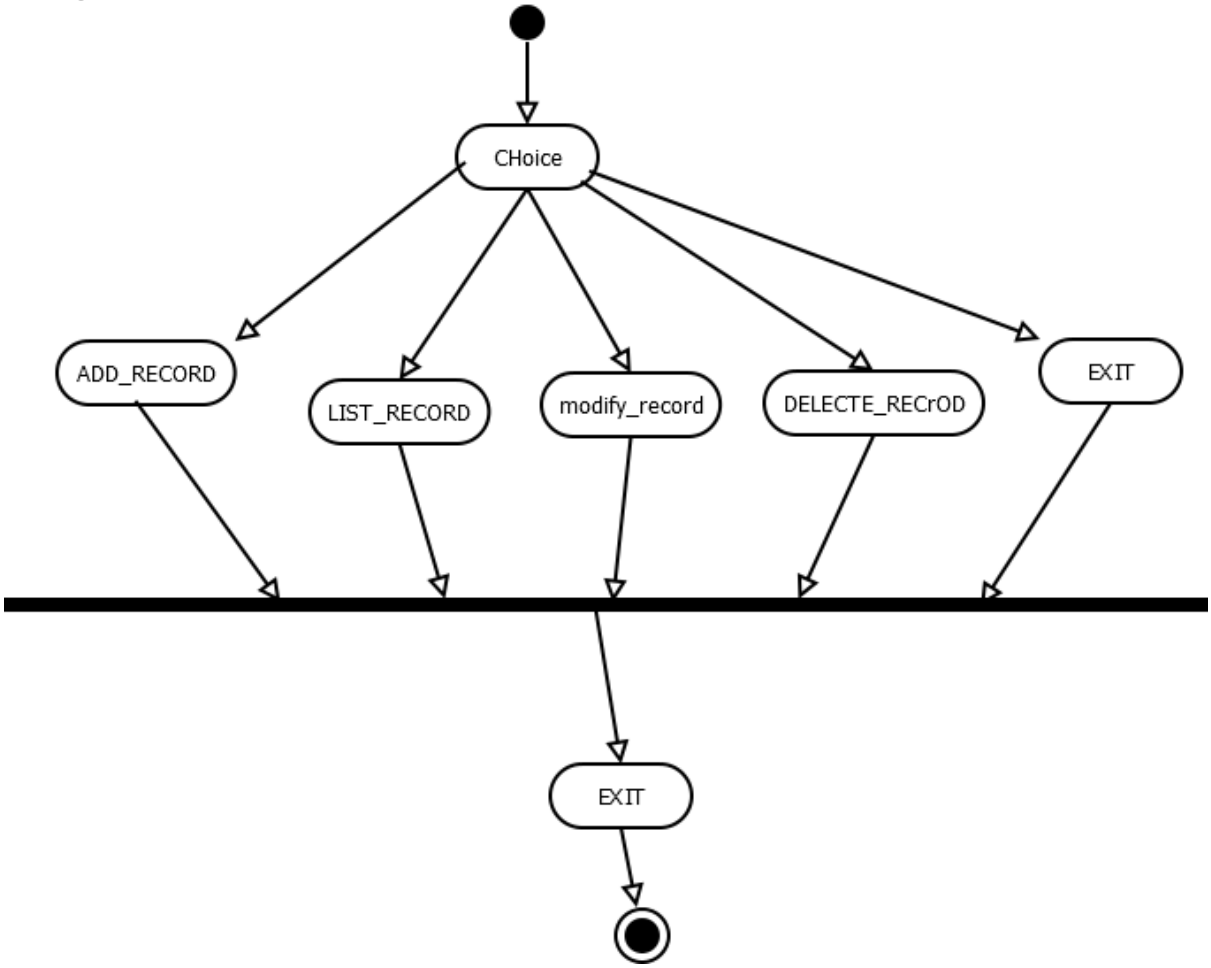

Figure 1 Behaviour Diagram

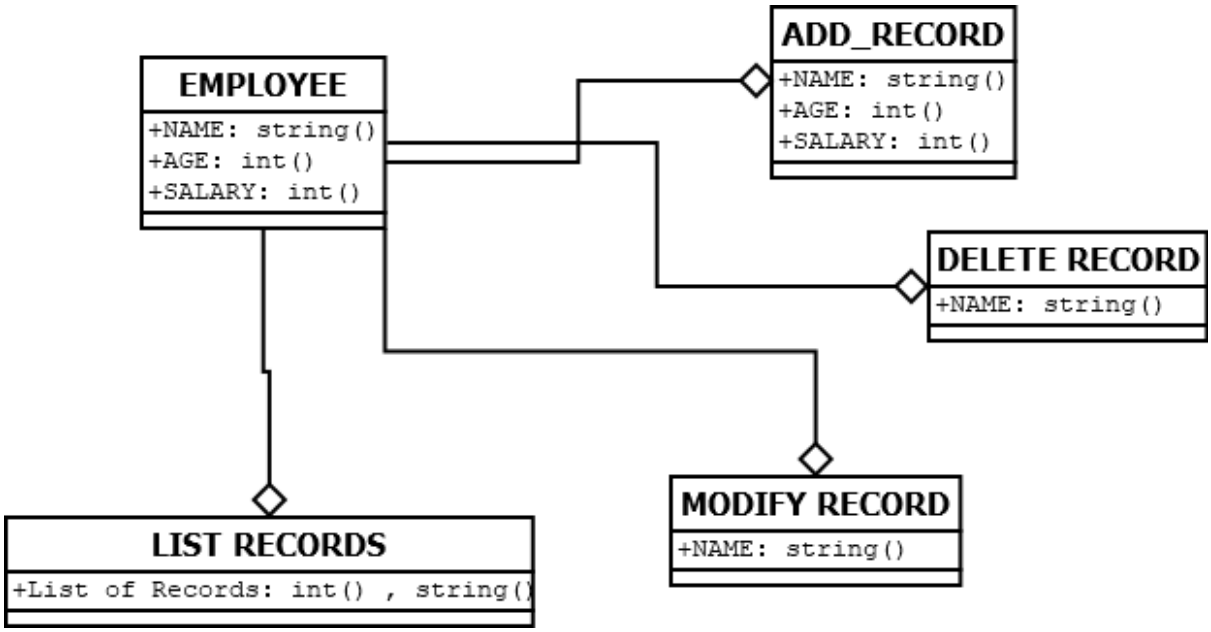

Figure 2 Structure Diagram

Test Plan

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Type of Test
TC_01	for entering name	enter name: abc	abc	abc	PASS
TC_02	for entering name	enter name: abc	abc	-	FAIL
TC_03	for entering age	enter age: 12	12	12	PASS
TC_04	for entering age	enter age: 12	12	-	FAIL
TC_05	for entering salary	enter salary: 5000	5000	5000	PASS
TC_06	for entering salary	enter salary: 5000	5000	-	FAIL
TC_07	for listing record	abc 12 5000	abc 12 5000	abc 12 5000	PASS
TC_08	for listing record	abc 12 5000	abc 12 5000	abc 5000 12	Fail
TC_09	for listing record	abc 12 5000	abc 12 5000	12 abc 5000	FAIL
TC_10	for deleting record	abc	abc	xyz	FAIL
TC_11	for deleting record	abc	abc	abc	PASS

Implementation and Summary

Git Link:

Link: https://github.com/vettri1827/M1_application_EmployeRecordSystem

Git Dashboard

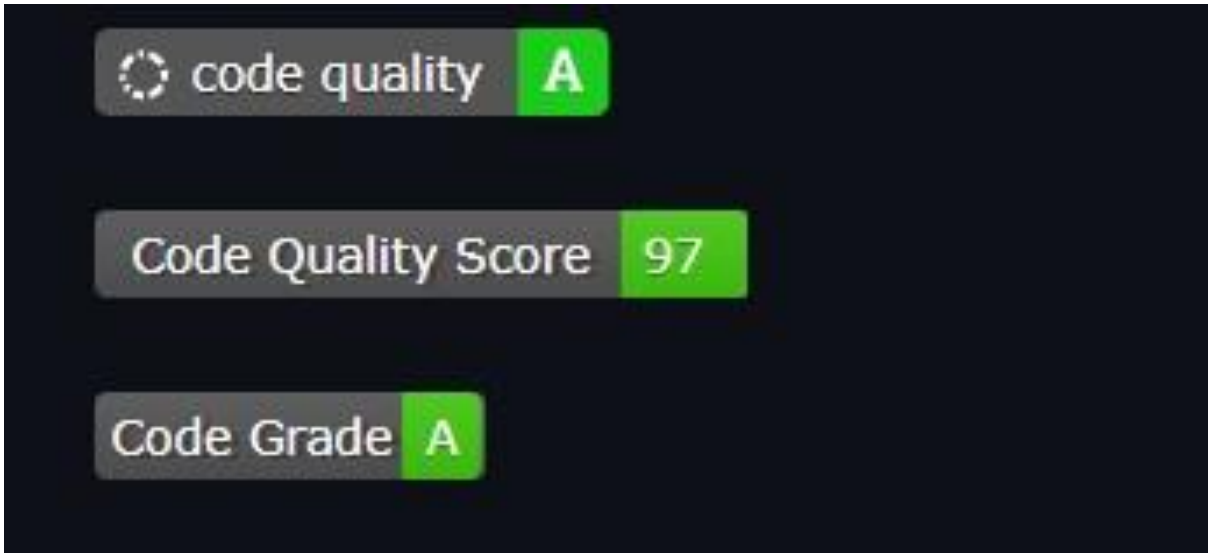

Figure 3 Git Dashboard

Miniproject 2 – Automated Water Tank Overflow Control [Individual]

Modules

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

Requirements

4W's and 1 H's

Who

This Product can be used by any consumers because its simple user friendly.

What

It makes water tank fill and cut off water automatically.

When

When the water level is low the tank fill automatic.

Where

This sysstem can be used in any place like home,office,park,etc...

How

The system controled by atmega328 which act ass brain.

High Level Requirements

ID	Description	Status
HLR1	display level	Implemented
HLR2	flow water when level low	Implemented
HLR3	stop water flow when level is reached	Implemented

Low Level Requirements

ID	Description	HLR ID	Status
LLR1	if low level switch open before uplevel the water won't flow	Implemented	LLR1
LLR2	the system works without errors	Implemented	LLR2

Design

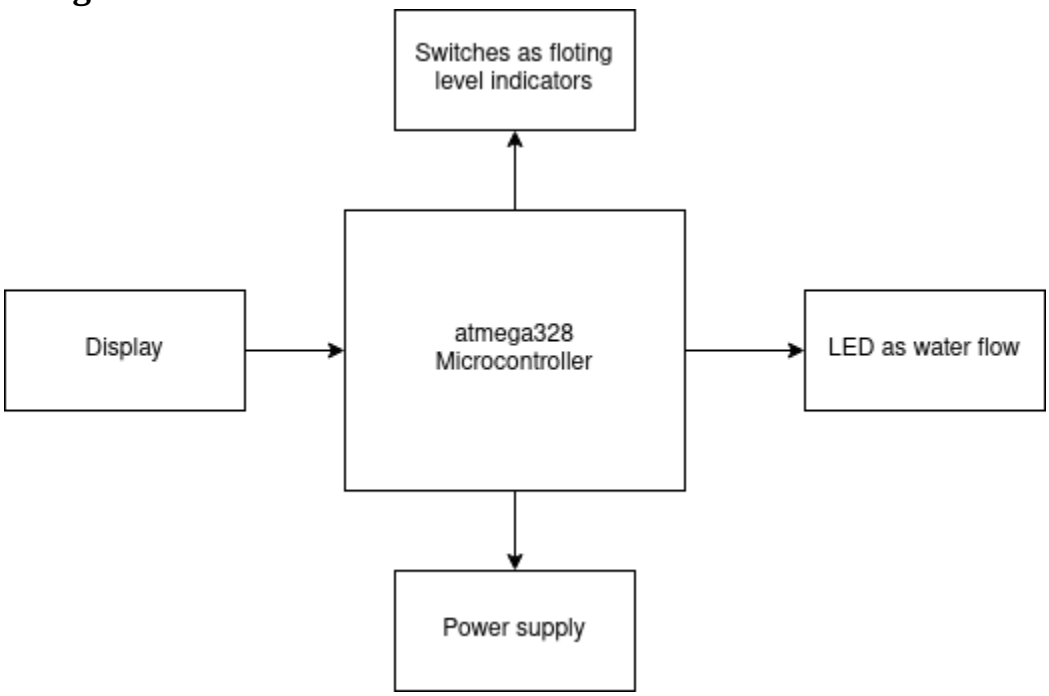

Figure 4 Behaviour Diagram

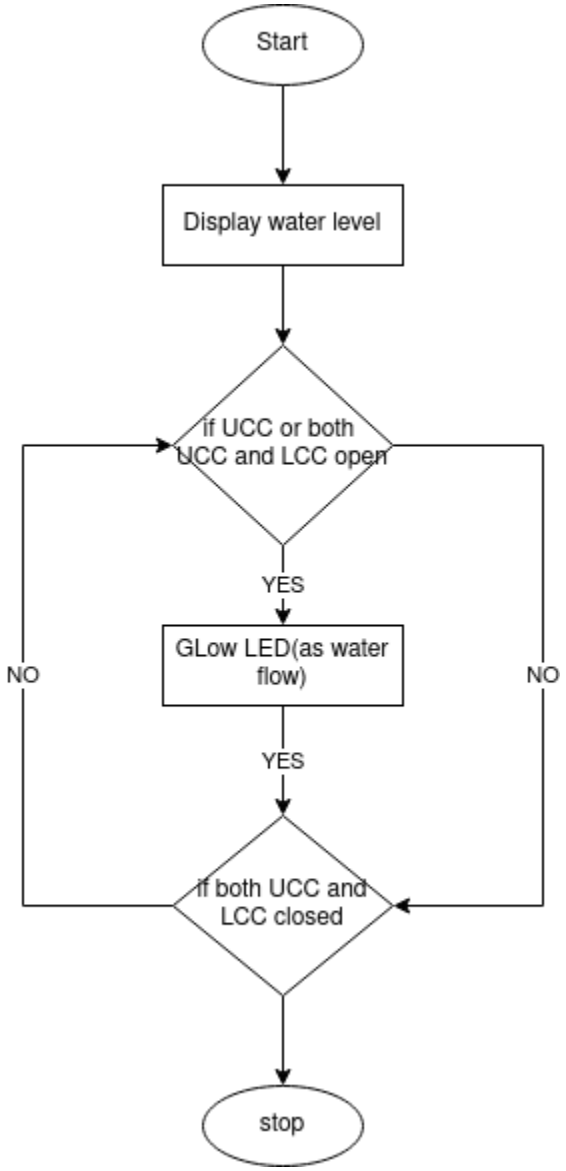

Figure 5 Structure Diagram

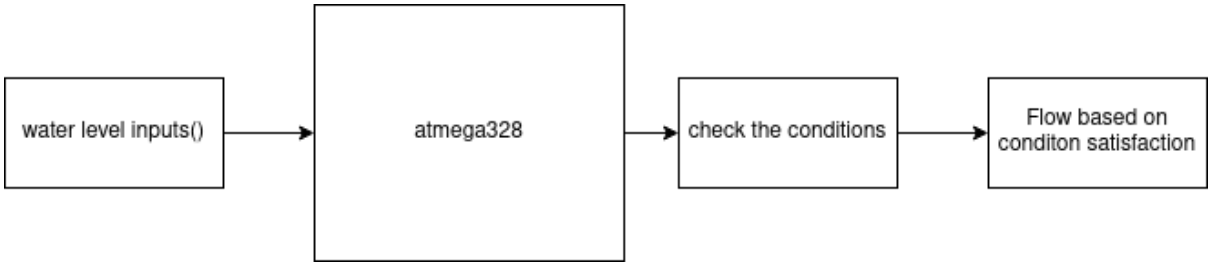

Figure 6 Block Diagram

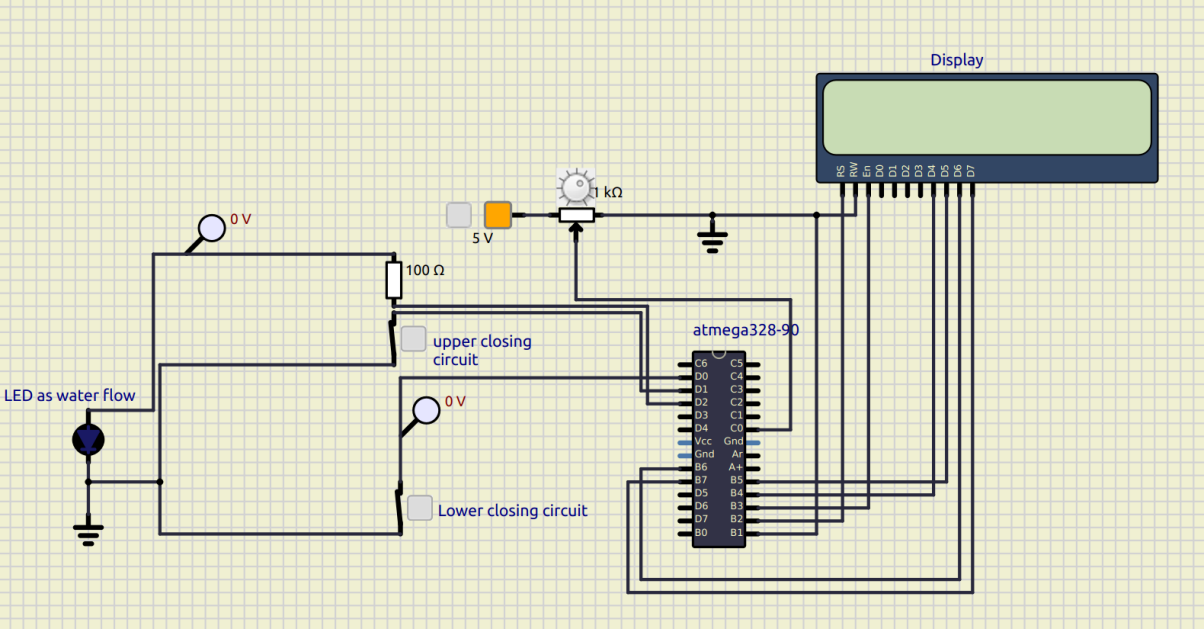

Figure 7 Simulation

Test Plan

High Level Test Plan

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Type of Test
H_01	Display Level Of Water	NA	0L - 3000L	700L	Pass
H_02	LED Glows when both Switches Open	NA	LED Glows	LED Glows	Pass
H_03	LED OFF when both Switches Closed	NA	LED OFF	LED OFF	Pass

Low Level Test Plan

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Type of Test
L_01	LED Glows when UCC only open	NA	LED Glows	LED Glows	Pass
L_02	Level of water varies from 0L - 3000L	NA	varies from 0L - 3000L	varies from 0L-3000L	Pass

Implementation and Summary

Git Link:

Link: https://github.com/vettri1827/M2-Embeded_Water_Tank_Overflow_Indicator

Git Dashboard

Figure 8 Git Dashboard

Miniproject 3 – Chatter Bot [Team]

Modules

1. SDLC
2. Git

Requirements

4W's and 1 H's

Why:

1. Software application used to conduct an online chat conversation via text or speech.
2. A Computer program which simulates a natural human conversation.

Where:

1. Retail and E-Commerce industries.
2. Used in Healthcare.

Who

1. Clients who need assistance.
2. Peoples who need support.

When:

1. To Provide faster and cheaper assistance to client.
2. To be Increasingly comfortable with Technology.

How:

1. Customers who are dealing with their problems late at night, chatbot are blessing as they can work around the clock.
2. During conversations with the customers, chat box provides a bridge between sales and customer team.

High Level Requirements

ID	Description	Category	Status
HLR_1	Chatterbot	Technical	Implemented

Low Level Requirements

ID	Description	HLR ID	Status
LLR_1	Process input	HLR_1	Implemented
LLR_2	Logic adapter 1	HLR_1	Implemented
LLR_3	Logic adapter 2	HLR_1	Implemented

Design

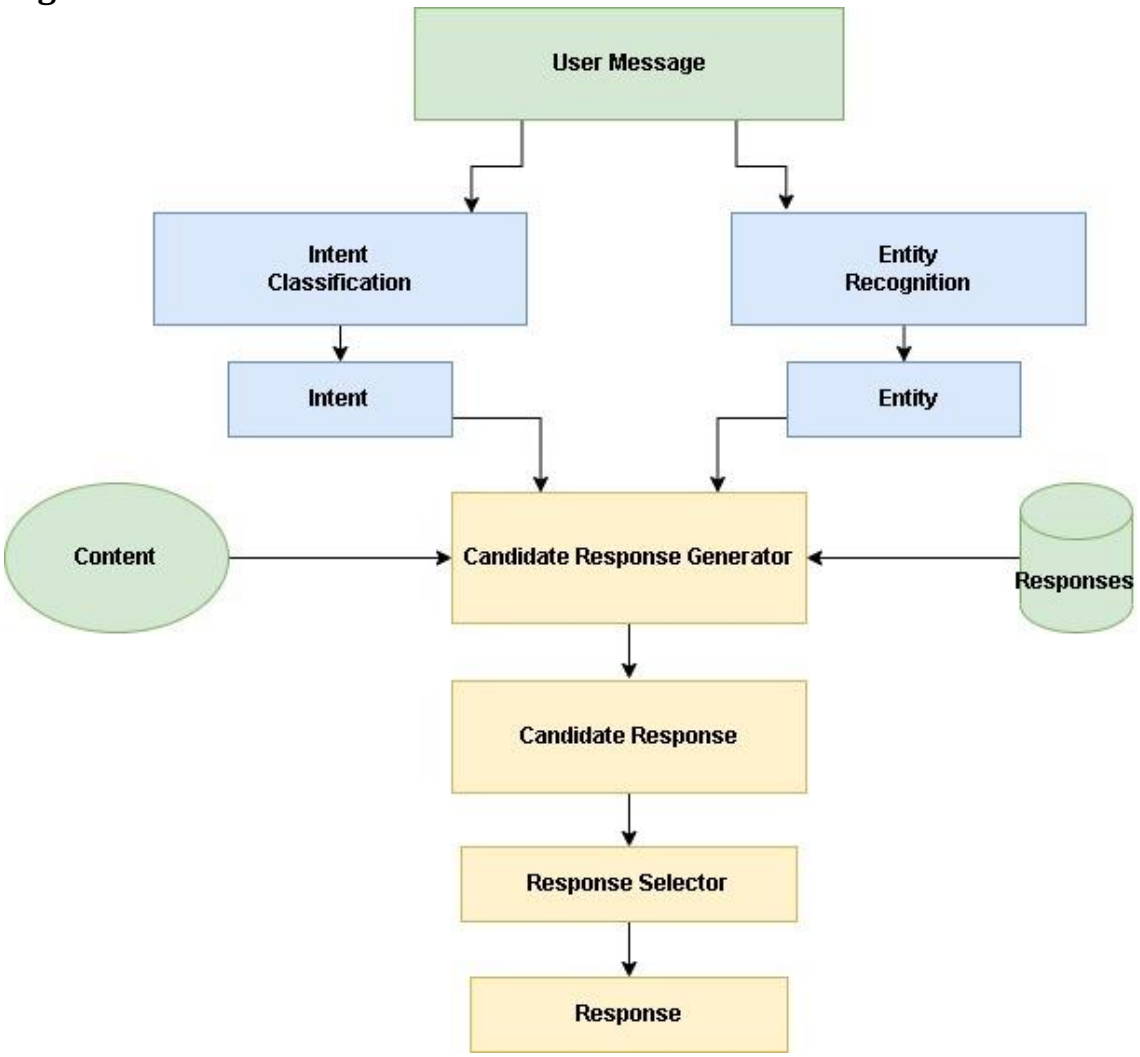

Figure 9 Behaviour Diagram

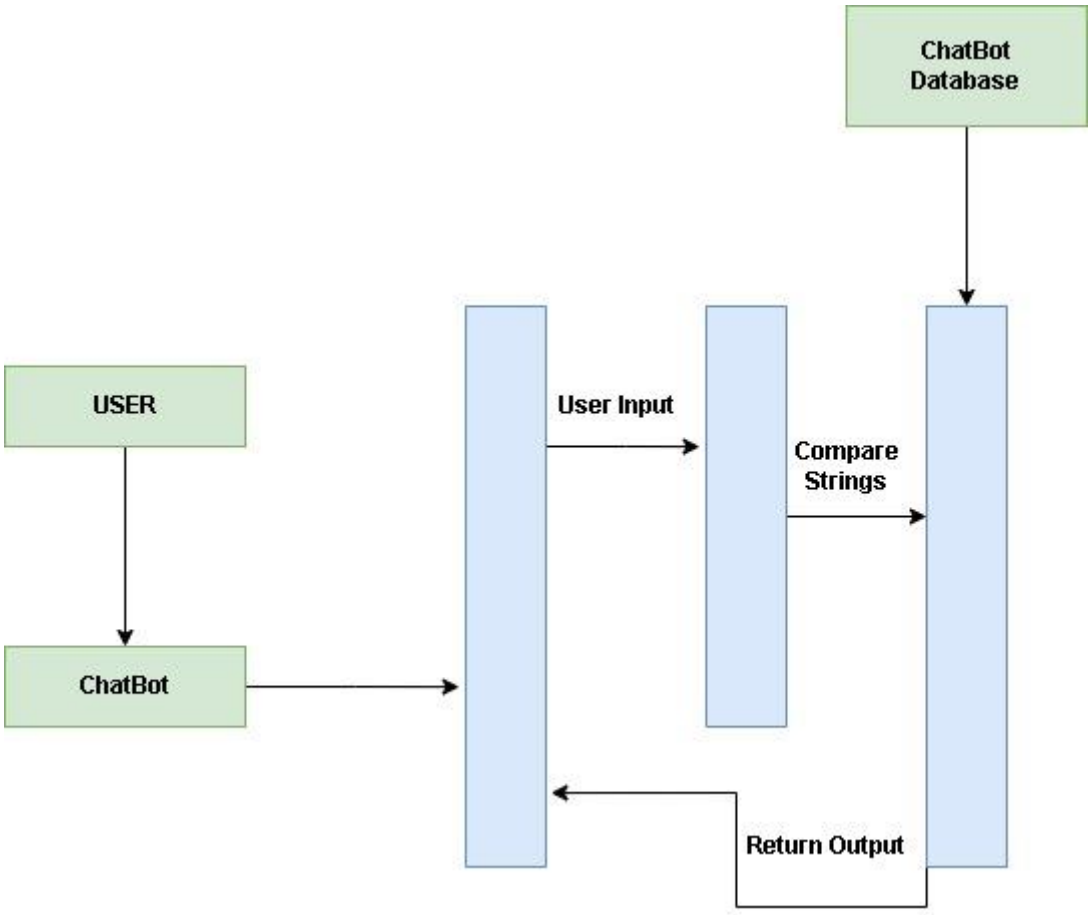

Figure 10 User Flow Diagram

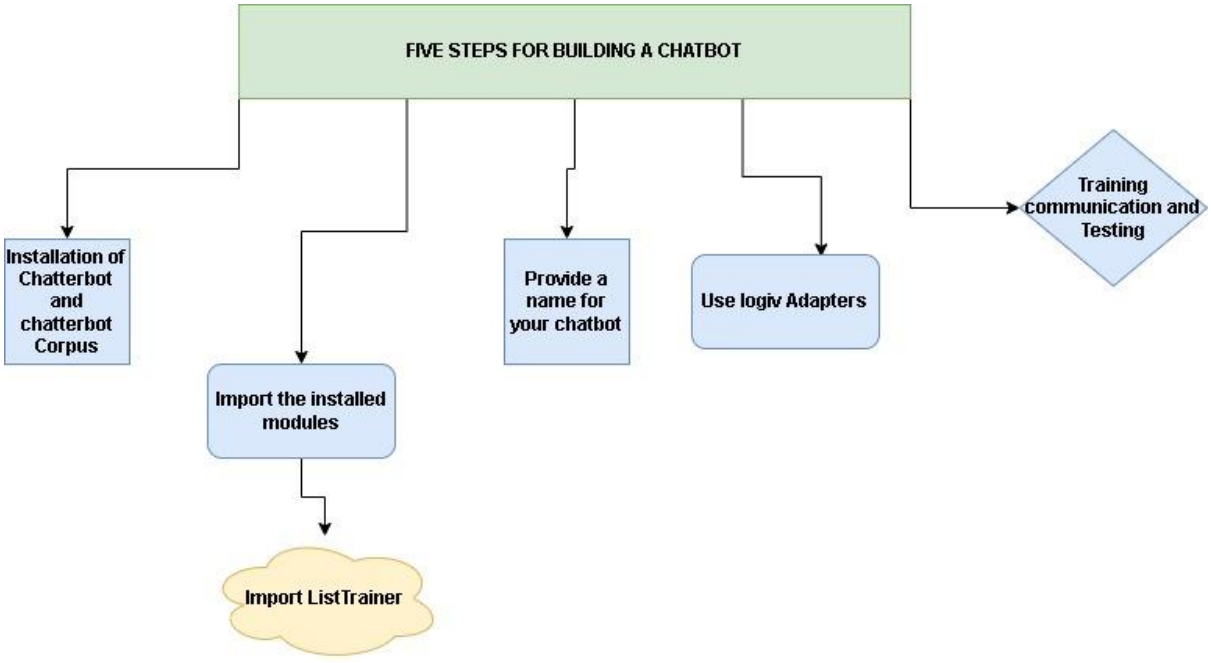

Figure 11 Structure Diagram

Test Plan

High Level Test Plan

Test ID	Description	Exp I/P	Exp O/P	Actual Out	Type Of Test
HLTP_1	Get input	User input	Return user input to the Process input	SUCCESS	Requirement Based
HLTP_2	Read input	Process input	Return a response related to the given User input	SUCCESS	Requirement Based
HLTP_3	Get output	Process input	Return the response from the Process input to the user	SUCCESS	Requirement Based

Low Level Test Plan

Test ID	HLTP ID	Description	Exp IN	Exp OUT	Actual Out	Type Of Test
LLTP_1	HLTP_1	The inputs can be given only by using console, API, speech recognition, etc.	User input	SUCCESS	SUCCESS	Requirement Based
LLTP_2	HLTP_2	Select a known statement that most closely matches the given User input	Process input	SUCCESS	SUCCESS	Requirement Based

Test ID	HLTP ID	Description	Exp IN	Exp OUT	Actual Out	Type Of Test
LLTP_2.1	HLTP_2	Return a known response to the selected match and a confidence value based on the matching	Process input	SUCCESS	SUCCESS	Requirement Based
LLTP_3	HLTP_3	Return the response to the user only by using console, API, speech recognition, etc.	User input	SUCCESS	SUCCESS	Requirement Based

Implementation and Summary

Git Link:

Link: https://github.com/GENESIS2021Q1/Applied_SDLC-Dec_Team_1

Individual Contribution and Highlights

Summary

1. Implementation
2. Testing

Role in Project Team

1. Implementation: Implemented a python code for test file.
2. Testing: Tested the Chatter Bot using spell checking.

Miniproject 4 – Attendance Automation [Team]

Modules

1. Python
2. Git

Requirements

High Level Requirements

ID	Description	Status
HLR_1	Attendance Status	Implemented
HLR_2	User Details	Implemented
HLR_3	User load Sheet	Implemented
HLR_4	Output File Generation	Implemented

Low Level Requirements

ID	Description	HLR ID	Status
LLR_1	User can get the attendance status	HLR_1	Implemented
LLR_2	User can enter status input to get the attendance status	HLR_1	Implemented
LLR_3	User can get the user details	HLR_2	Implemented
LLR_4	User will get the details after the successful attendance	HLR_2	Implemented
LLR_5	User can load different sheets	HLR_3	Implemented
LLR_6	User can modify the existing sheets as it is dynamic	HLR_3	Implemented
LLR_7	Output file gets generated	HLR_4	Implemented

Test Plan

High Level Test Plan

ID	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
HLTP_1	Attendance Status	User Input	SUCCESS	SUCCESS	Requirement Based
HLTP_2	User details	User Input	SUCCESS	SUCCESS	Requirement Based
HLTP_3	User load sheet	User Input	SUCCESS	SUCCESS	Requirement Based
HLTP_4	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_1	HLTP_1	User can get Attendance Status	SUCCESS	SUCCESS	Requirement Based
LLTP_2	HLTP_1	User can enter Status input to get the Attendance Status	SUCCESS	SUCCESS	Requirement Based
LLTP_3	HLTP_2	User can get the User details	SUCCESS	SUCCESS	Requirement Based
LLTP_4	HLTP_2	User will get the details after the successful attendance	SUCCESS	SUCCESS	Requirement Based
LLTP_5	HLTP_3	User can load different sheets	SUCCESS	SUCCESS	Requirement Based
LLTP_6	HLTP_3	User can also modify the existing sheets as it is dynamic	SUCCESS	SUCCESS	Requirement Based

ID	HLTP ID	Description	Expected I/P	Actual O/P	Type Of Test
LLTP_7	HLTP_4	Output file gets generated	SUCCESS	SUCCESS	Requirement Based

Implementation and Summary

Git Link:

Link: https://github.com/kavinvignes/GENESIS2021-OOPS_Python-Attendance_Automation-Team_13

Git Dashboard

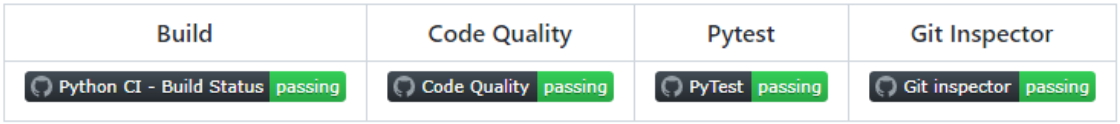

Figure 12 Git Dashboard

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. Tester: Writing Testcases for the main program.

Miniproject 5 – Toyota Project [Team]

Modules

1. Matlab
2. Git

Requirements

We have implemented following features

1. Adaptive Cruise Control System
2. Anti Lock Braking System
3. Thermal Management System
4. Door Locking system
5. Climate Control system
6. Lane Assist System
7. Power Window

Temperature Control System

The Battery Thermal Management System (BTMS) is the device responsible for managing/dissipating the heat generated during the electrochemical processes occurring in cells, allowing the battery to operate safely and efficiently.

The lithium-ion battery in electric vehicles is an important energy storage device that requires proper temperature control system. A considerable amount of heat is generated by the battery cells owing to their internal resistance during charging and discharging, especially for peak vehicle loads. This focus on developing a smart controlled temperature control solution in which a vapor compression system is integrated. A lumped-parameter cylindrical battery thermal model is developed with a Kalman observer to estimate the transient changes in the temperatures of the battery surface, the battery core, and the cooling air flowing around the cells. The optimal cooling air temperature of the battery is investigated using optimal control theory. A model predictive controller is then introduced to regulate the refrigerant compressor and to track the ideal cooling air temperature. The power consumption of the thermal management system and the behaviour of the internal temperature.

Design

This project was implemented using Matlab Design

Implementation and Summary

Git Link:

Link: https://github.com/RAshwin990/Team_Toyota

Individual Contribution and Highlights

1. Thermal management system case study
2. Designed and implemented using matlab and the file is in github.

Role in Project Team

1. Designer: Done Designing for Thermal management system using matlab.

Miniproject 6 – Wiper Control [Team]

Modules

1. C Programming
2. STM32

Requirements

4W's and 1'H

WHO

- Person who is driving the vehicle can able to use the wiper system.

WHAT

- 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.

WHERE

- Used in four wheelers.
- Used in heavy vehicles.

WHY

- Used to remove rain, snow from a vehicle front window
- To ensure the driver's safety.

High Level Requirements

ID	Description
HLR1	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.
HLR2	A windscreen wiper or windshield wiper is a device used to remove rain, snow, ice and dust from a windscreen or windshield.
HLR3	Quality and reliability wiper systems meet the highest technical requirements and are the basis for vehicles with sophisticated features.
HLR5	Almost all motor vehicle, including trains, aircraft and watercraft, are equipped with such wipers, which are usually an essential requirement.
HLR6	Our project brings forward this system to automate the wiper system having no need for manual intervention.

Low Level Requirements

ID	Description
LLR1	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.
LLR2	Wiper motor is automatically ON during the time of rainfall and dust
LLR3	Existing system manually used control stalk to activate wiper and the process of pulling up wiper is difficult to be handled.
LLR4	Lower level parsing. Under the hood, the Requirement class does most of the heavy lifting. class requirements.
LLR5	These systems detect droplets of rain on the windshield and automatically turn on and adjust the wiper system similarly the dust partical's detected and wiped off.

Design

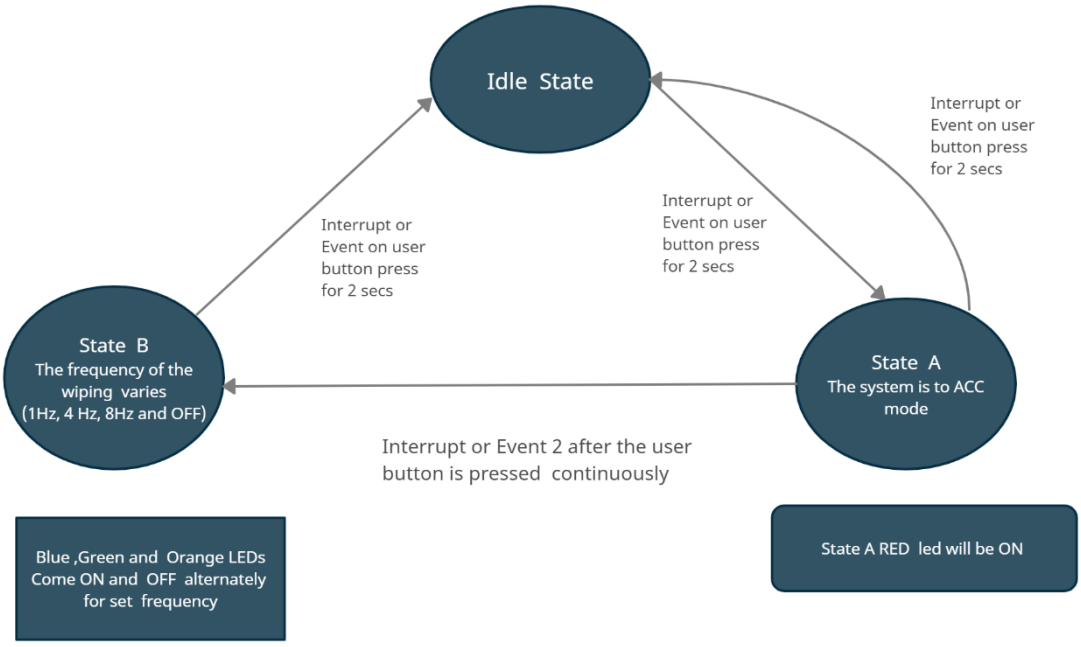

Figure 13 Structure Diagram

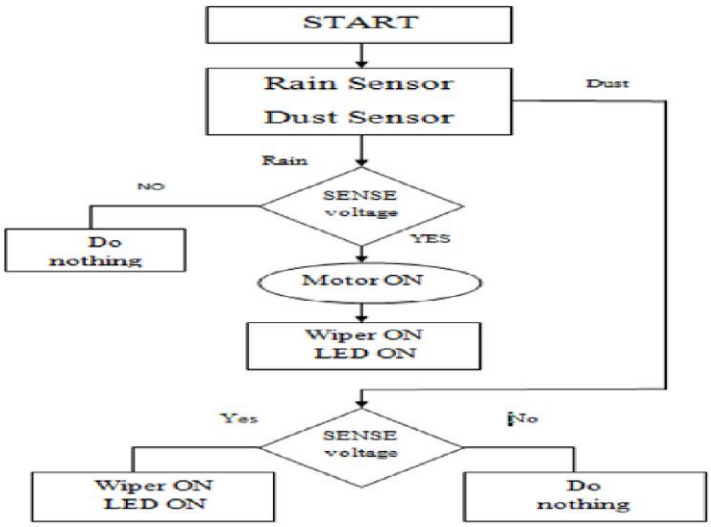

Figure 14 Behaviour Diagram

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

ID	Description	Output	HLTP ID	Type of Test
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

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 – Tata Altroz [Team]

Modules

1. Automotive Systems
2. Git

INTRODUCTION

Our integrated circuits and reference designs for automotive exterior rearview mirror modules help you accelerate design through accurate light sensors and LED lighting solutions with high precision in controlling and driving electrochromic mirrors and high integration for human interface (HMI) control. Modern rearview mirror modules require: Accurate, stable control of large capacitive load for electrochromic auto dimming. Precise sensing of ambient light to allow effective auto dimming.

FEATURES

BLIND SPOT DETECTION

Blind Spot Detection tracks traffic just behind you. The alert stays active until the car in the adjacent lane is in front of you, or at least directly alongside and you'd have to be blind not to see it. Blind Spot Detection uses ultrasonic or radar sensors on the side and rear of the car.

DEFOGGERS

Defoggers are used to defog the rear window, and to remove raindrops, dew and frost from the outside rear view mirrors. The operation time changes according to the ambient temperature and vehicle speed, It may be 10 minutes to 25 minutes.

COMPONENTS

DC MOTOR

Each rear-view mirror has two DC motors. One DC motor operates the up/down function while the other DC motor operates the left/right function. Both switches inside the power mirror switch are constantly connected to the vehicle's electrical ground circuit with the switch at rest.

INFRARED SENSOR

An infrared sensor (IR sensor) is a radiation-sensitive optoelectronic component with a spectral sensitivity in the infrared wavelength range 780 nm … 50 µm. IR sensors are now widely used in motion detectors, which are used in building services to switch on lamps or in alarm system.

RADAR

Radar is a electromagnetic sensor used for detecting, locating, tracking, and recognizing objects of various kinds at considerable distances.

High Level Requirements

	TITLE	MODULES	DESCRIPTION
SYS_1	Requirements	DC MOTOR	The Side Rear View Mirror should Open when the Engine Started.
SYS_2	Requirements	DIGITAL CAMERA	The Main Camera Present in the Mirror gives the precise view of the mirror in an LCD Display inside the car when the weather outside is not good.
SYS_3	Requirements	DC MOTOR	When the Engine Turns off, the Mirror Should Close Automatically.
SYS_1	Requirements	DC MOTOR	The Side Rear View Mirror should Open when the Engine Started.

Low Level Requirements

	TITLE	MODULE	DESCRIPTION
SYS_1	Requirements	H-MOD	The Defogger System should connect with the Aircon System to generate

	TITLE	MODULE	DESCRIPTION
			the heat and make the mirror crystal clear for the driver.
SYS_2	Requirements	INFRARED SENSOR	The Sensor in the side mirror which can detect the chasing vehicle near it and gives a signal to the driver to be cautious.
SYS_3	Requirements	RADAR	To detect the Blind spots of the DRIVER, up to 6 cameras have been placed in the rear mirror to cover the entire blind spot.

Design

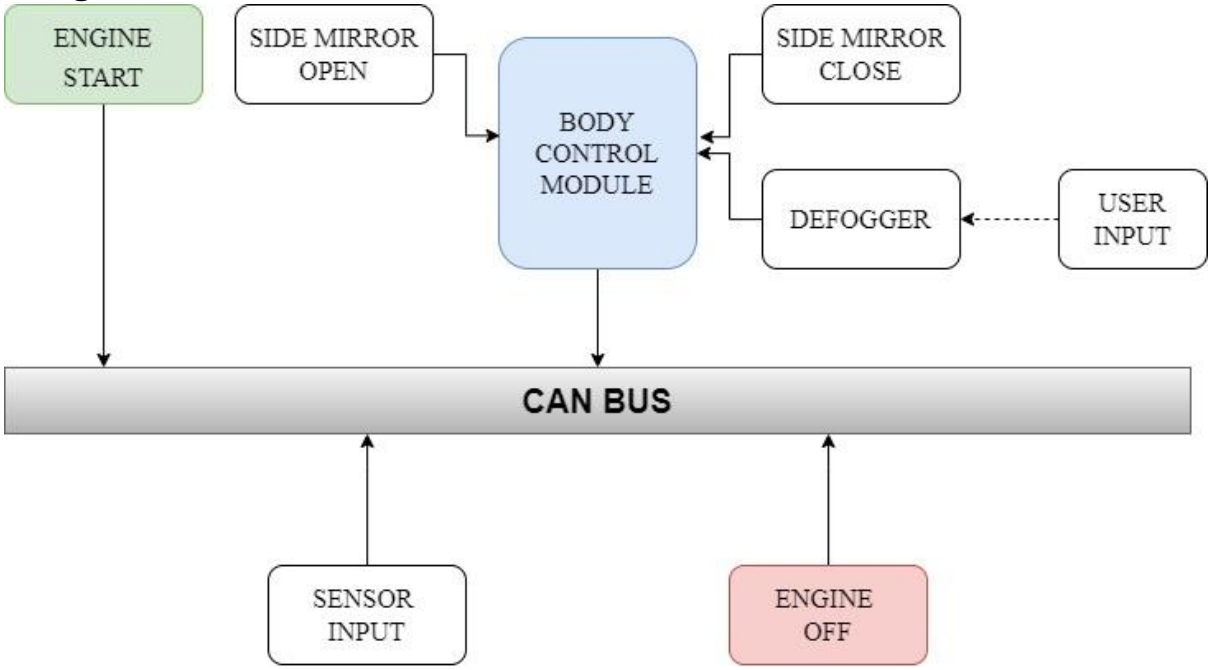

Figure 15 Structure Diagram

Implementation and Summary

Git Link:

Link: https://github.com/vettri1827/Automotive_Tata_Altroz

Individual Contribution and Highlights

1. Power Window Case Study
2. Source code management using GitHub

Role in Project Team

1. Designer: Done Designing for Project
2. Researcher: Done case study for Power Window

Miniproject 8 – EV Bike[Team]

Modules

1. Matlab
2. Matlab Script

Requirements

Motor Performance:

1. Our Arrow M1 has a Mid Drive IPM motor which can produce 7.2 kW power and 40 Nm torque. We find these figures to be a nice balance of drivability and efficiency.
2. Arrow M1 has an acceleration time from 0 to 60 km/hr of 6.5 seconds.
3. Top speed of our Arrow M1 is 100 km/hr

Battery Performance:

1. We are using a Lithium polymer battery to reduce weight and thereby increase fuel efficiency, performance and handling.
2. A range of 220 km is class leading due to our battery being the biggest at 4.6 kWh.
3. Charging times of our Arrow M1 is higher than the competition at 7.15 hours but we make up for it in the range section.
4. We also offer fast charging.

Braking Performance:

1. Our Arrow M1 also uses combi braking system and use disc brakes for both front and back wheels.
2. Braking performance is on par with the competition.

Wheel Performance:

1. Our Arrow M1 uses Alloy wheels at 12 inches diameter.
2. We use a 90 section, 90 profile tire for a balance between grip, efficiency and ride quality.

Suspension Performance:

1. We use Mono shocks for rear and single fork for front.

Dimensions:

1. Our kerb weight is 110 kg which is just 2 kg heavier than the Ather 450X while having a substantially bigger battery and more powerful motor.
2. Length, Height and Weight are all comparable to the competition.
3. Wheelbase is 1370 mm is the longest in the segment.
4. With a seat height of 782 mm it is accessible for a wide range of people in terms of height.

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 – Side Mirror [Individual]

Modules

1. Autosar
2. Git

Requirements

The window switches on the driver's door control all the door windows. The passenger door windows can also be operated by using the single window switches on the passenger door trim panel. The window switches will operate only when the ignition is in the ON/RUN position.

High Level Requirements:

S.No.	Feature	Description
HLR_1	Side Mirror System	Mirrors can be accessed by switching on the Vehicle
HLR_2	Defogger	The Fog in the mirror is removed using this system.

Low Level Requirements:

S.No.	Feature	Description
LLR_1	Side mirror System	The Side Mirror will be open and close When the Vehicle stops and engine is off.

Design

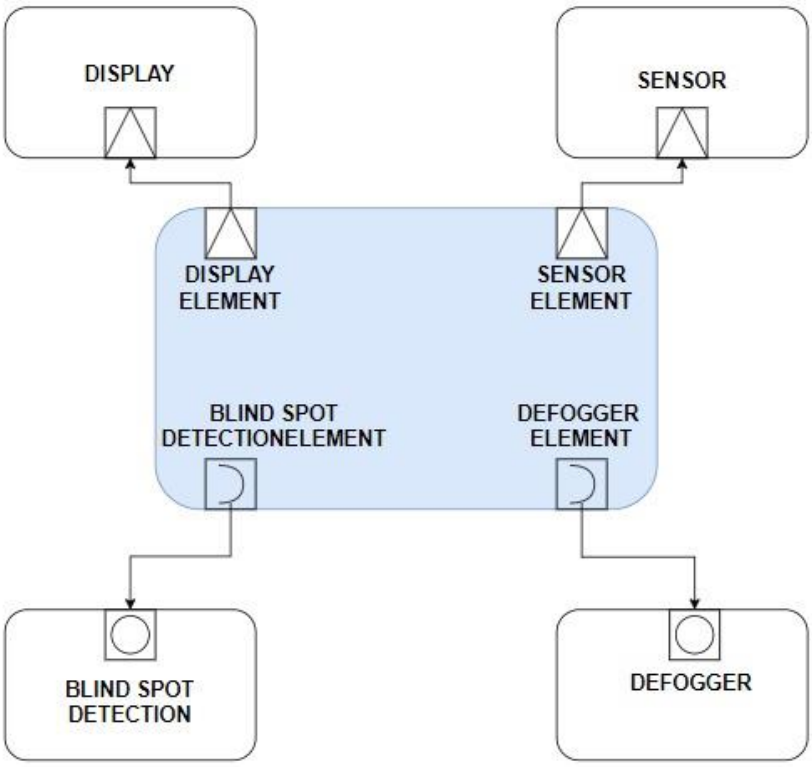

Figure 16 VFB Diagram

Implementation and Summary

Git Link:

Link: https://github.com/vettri1827/RearViewMirror_40020491_DPS

Individual Contribution and Highlights

1. Power Window Case Study
2. Source code management using GitHub
3. SWC Internal Behaviour
4. SWC Implementation