L&T Technology Services

Details

Ver. Rel. No.	Release Date	Prepared By	Modules	To Be Approved	Remarks/Revision Details
1.0	19/02/2022	Vishnu Kumar.V 40020488	C Programming on Multiple Platforms		
1.0	19/02/2022	Vishnu Kumar.V 40020488	Essentials of Embedded System		
1.0	19/02/2022	Vishnu Kumar.V 40020488	Applied SDLC and Software Testing		
1.0	19/02/2022	Vishnu Kumar.V 40020488	OOPS with Python		
1.0	19/02/2022	Vishnu Kumar.V 40020488	Applied Model Based Design Module		
1.0	19/02/2022	Vishnu Kumar.V 40020488	Mastering Microcontrollers with Embedded Driver Development Module		
1.0	19/02/2022	Vishnu Kumar.V 40020488	Overview of Automotive Systems		
1.0	19/02/2022	Vishnu Kumar.V 40020488	Applied Control Systems and Vehicle Dynamics		
1.0	19/02/2022	Vishnu Kumar.V 40020488	Classic Autosar Basic to Intermediate		

Contents

List of Figures

Miniproject – 1: Phonebook Management System [Individual]

Modules:

1. C Programming
2. Git

Requirements

4W's and 1 H's

Why:

1. To store all information under a single contact number

Where:

1. This can be used in our daily lives to search contacts.

Who:

1. It can be used by each and every individual.

When:

1. One is in need to search their very important contact.

How:

1. By giving different functions one can find their desired output

High Level Requirements

ID	Description	Status
HLR_1	The user can add the contact	Implemented
HLR_2	The user can search the contact	Implemented
HLR_3	The user can display the contact	Implemented
HLR_4	The user can delete the contact	Implemented

Low Level Requirements

ID	Description	Status
LLR_1	List of operations displayed	Implemented
LLR_2	Input from the user	Implemented
LLR_3	Exit the program	Implemented

Design

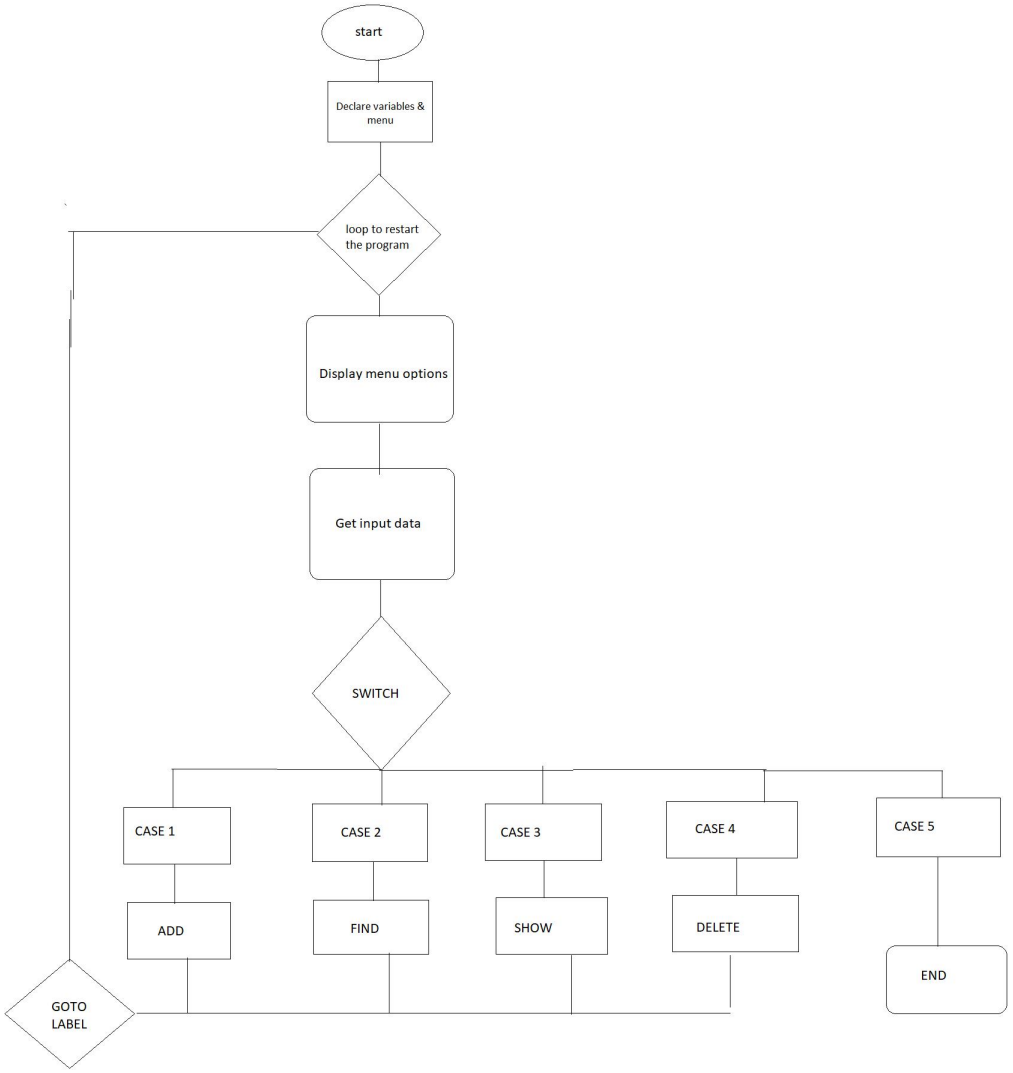

Figure 1 Behavior Diagram

Test Plan

High Level Test Plan

ID	Description	Status
HLR_1	The user can add the contact	Implemented
HLR_2	The user can search the contact	Implemented
HLR_3	The user can display the contact	Implemented
HLR_4	The user can delete the contact	Implemented

Low Level Test Plan

ID	Description	Status
LLR_1	List of operations displayed	Implemented
LLR_2	Input from the user	Implemented
LLR_3	Exit the program	Implemented

Implementation and Summary

Git Link:

Link: https://github.com/vishnukumar25/M1_application_Phonebook-management-system

Git Dashboard

Figure 2 Git Dashboard

Summary

Git Inspector Summary

```
Author                    Commits    Insertions      Deletions    % of changes
vishnukumar25                  47          4974            509          100.00

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
vishnukumar25             4497           90.4         0.1                8.41
```

Figure 3 Git Inspector Summary

Miniproject 2 – Functioning LED [Individual]

Modules

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

Requirements

4W's and 1 H's

Why:

1.To Function two LED's using a switch in ATmega328.

2.To understand basic concepts in ATmega328.

Where:

1. It can be used anywhere.

2. It can be used for understanding purposes in schools and colleges.

Who:

1.It can be used by students and trainees.

2.It can be used by anyone who are new to embedded programming language.

When:

1.People when they are in need to learn embedded programming language.

2.Mostly it can be done as projects in schools and colleges

How:

1. By using softwares (C programming) to exceute the program.

2. By uploading the program in ATmega328.

High Level Requirements

ID	Description	Status
HLR_1	Microcontroller	Implemented
HLR_2	Switch	Implemented
HLR_3	LED	Implemented
HLR_4	Software	Implemented

Low Level Requirements

ID	Description	Status
LLR_1	ATmega328	Implemented
LLR_2	Switch	Implemented
LLR_3	LED	Implemented
LLR_4	Visual studio & SimulIDE	Implemented

Design

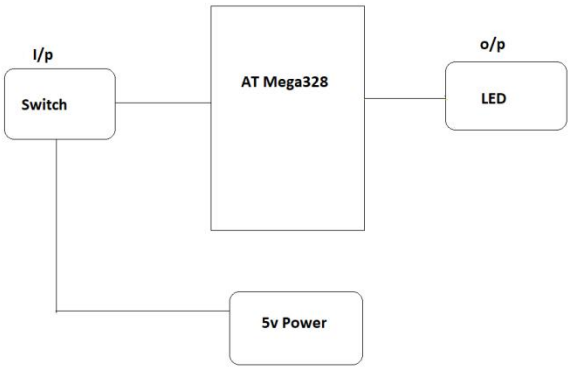

Figure 4 Block Diagram

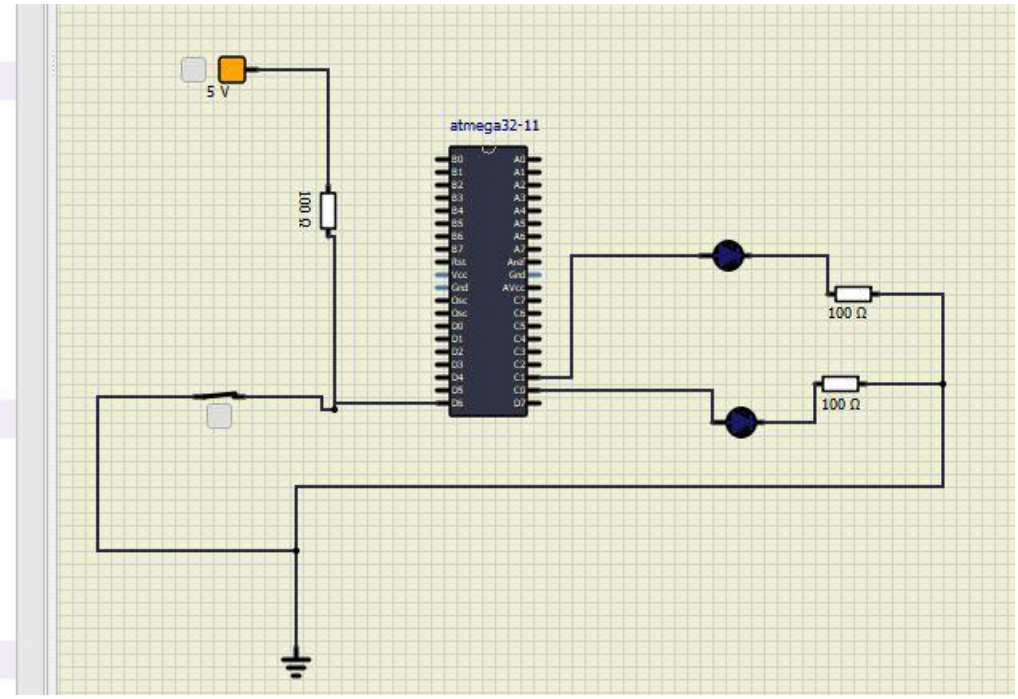

Figure 5 Simulation

Test Plan

High Level Test Plan

Id	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
HLR_1	Switch on	High power	LED On	LED On	Requirement
HLR_2	Switch Off	No power	LED Off	LED Off	Requirement

Low Level Test Plan

Id	Description	Expected I/P	Expected O/P	Actual O/P	Type Of Test
LLR_1	Switch on	value 1	LED On	LED On	Requirement
LLR_2	Switch Off	value 0	LED Off	LED Off	Requirement

Implementation and Summary

Git Link:

Link : https://github.com/vishnukumar25/M2-EMBEDDED_Functioning_LED.git

Git Dashboard

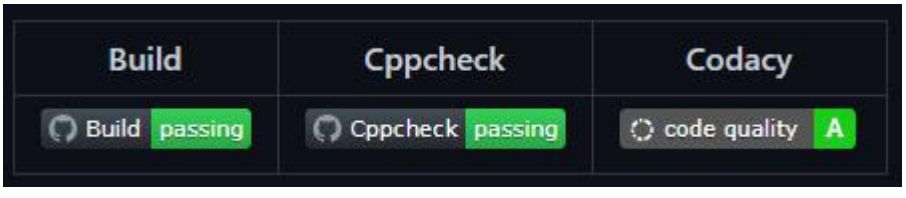

Figure 6 Git Dashboard

Miniproject 3 – AI ChatBot [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

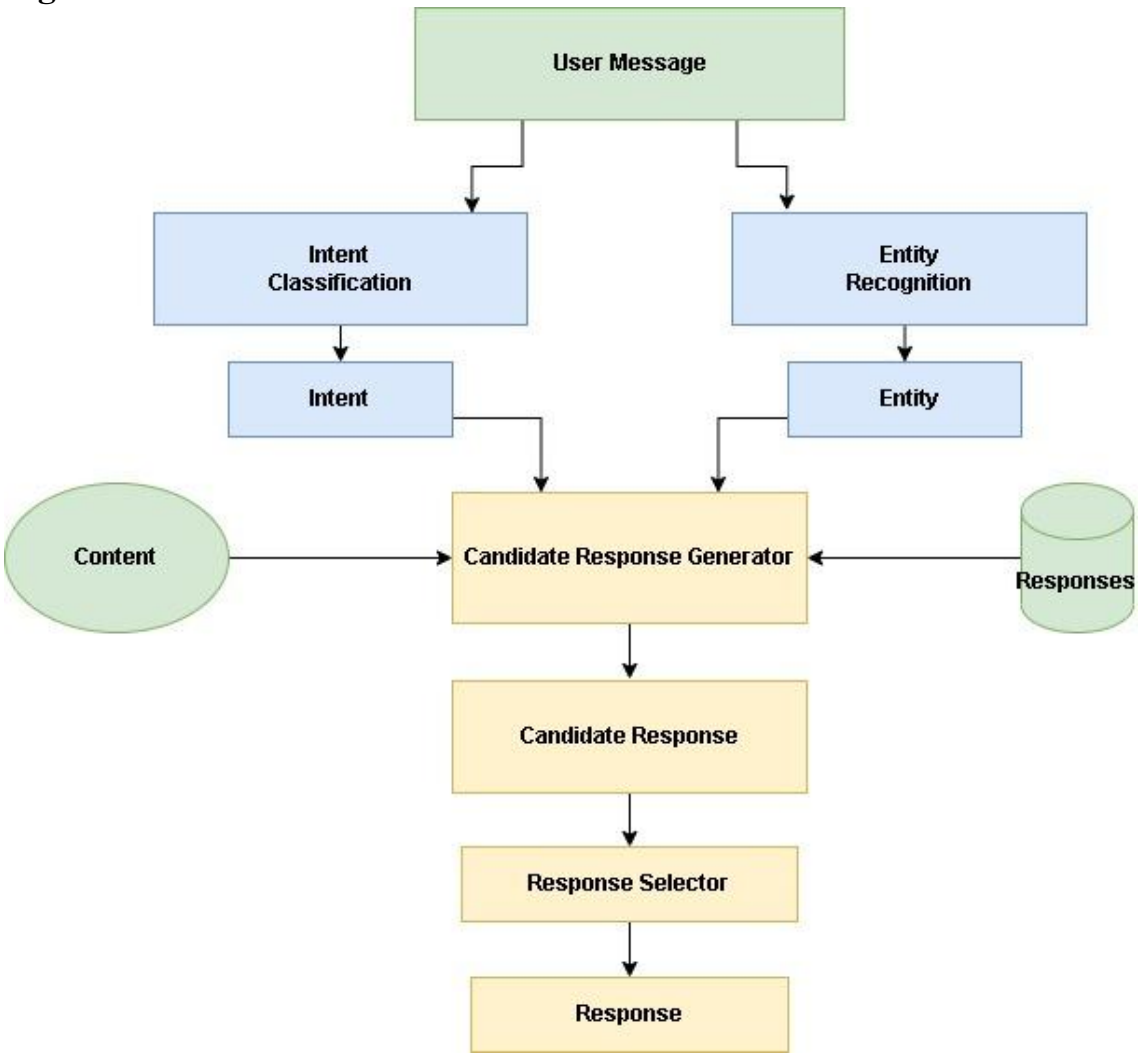

Figure 10 Behaviour Diagram

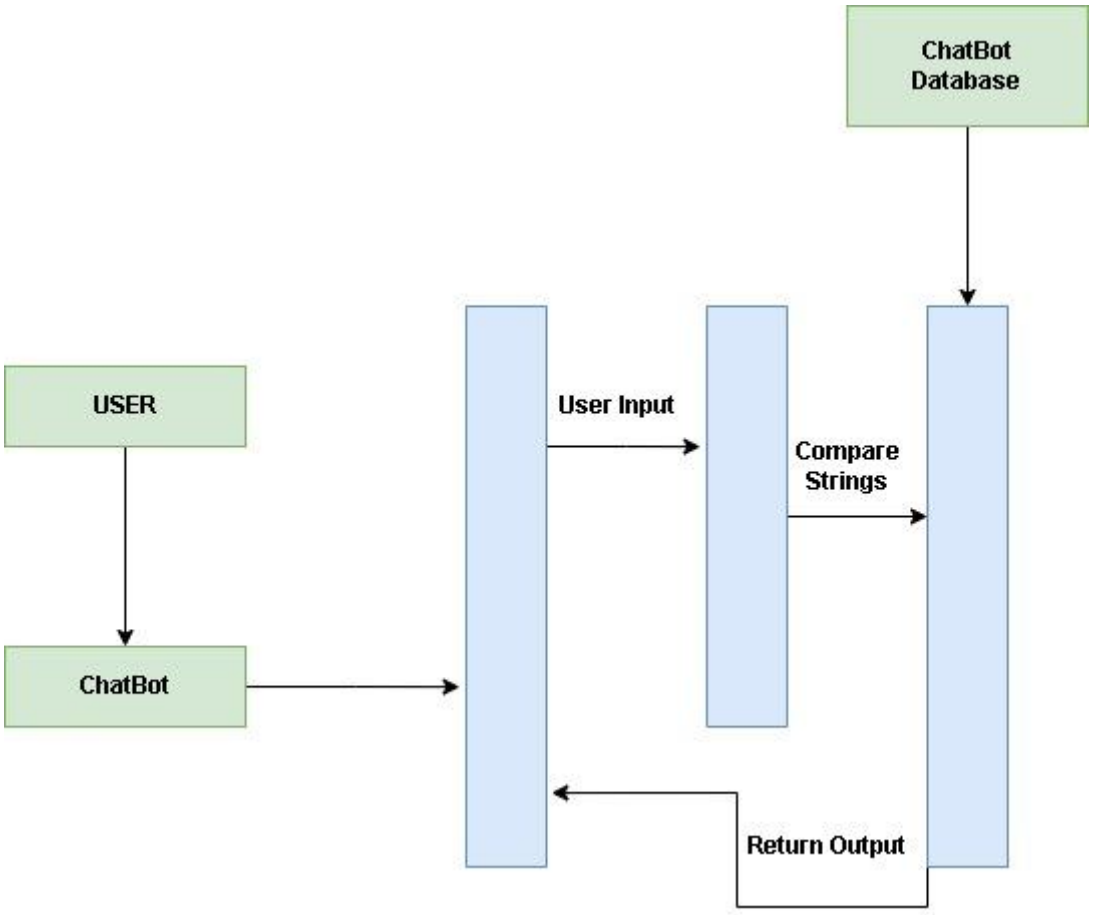

Figure 11 User Flow Diagram

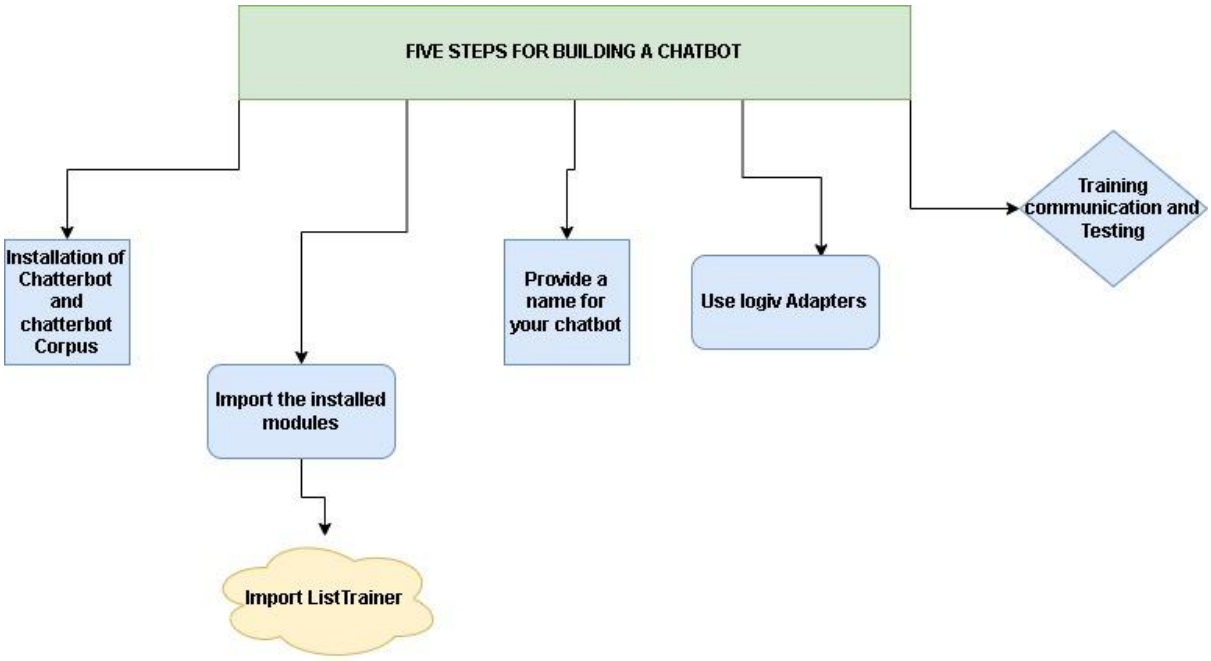

Figure 12 Structure Diagram

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
LLTP_2.	HLTP_2	Return a	Process	SUCCESS	SUCCESS	Requirement

Test ID	HLTP ID	Description	Exp IN	Exp OUT	Actual Out	Type Of Test
1		known response to the selected match and a confidence value based on the matching	input			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

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
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

Build	Code Quality	Pytest	Git Inspector
Python CI - Build Status passing	Code Quality passing	PyTest passing	Git inspector passing

Figure 13 Git Dashboard

Git Inspector Summary

```
Author                    Commits    Insertions      Deletions    % of changes
KARTHIKEYAN SELVARAJ            6            34             10            5.80
MidhunChakravarthi-0            3             8              2            1.32
kavinvignes                    43           381            228           80.34
praba2499                       2            17             10            3.56
robin6119                       2            65              3            8.97

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
KARTHIKEYAN SELVARAJ         2            5.9         7.1                0.00
MidhunChakravarthi-0         6           75.0         7.1                0.00
kavinvignes                215           56.4         5.3                0.00
praba2499                   11           64.7         7.1                0.00
robin6119                   18           27.7         7.1                0.00
```

Figure 14 Git Inspector Summary

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. Automatic Transmission Control System

4. Door Locking system
5. Climate Control system
6. Lane Assist System
7. Power Window

Design

This project was implemented using Matlab.

Implemenation and Summary

https://github.com/RAshwin990/Team_Toyota

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.

Why:

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

2. To ensure the driver's safety.

Where:

1. Used in four wheelers .

2. Used in heavy vehicles.

How:

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

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,similerly the dust particals detected and wiped off.

Design

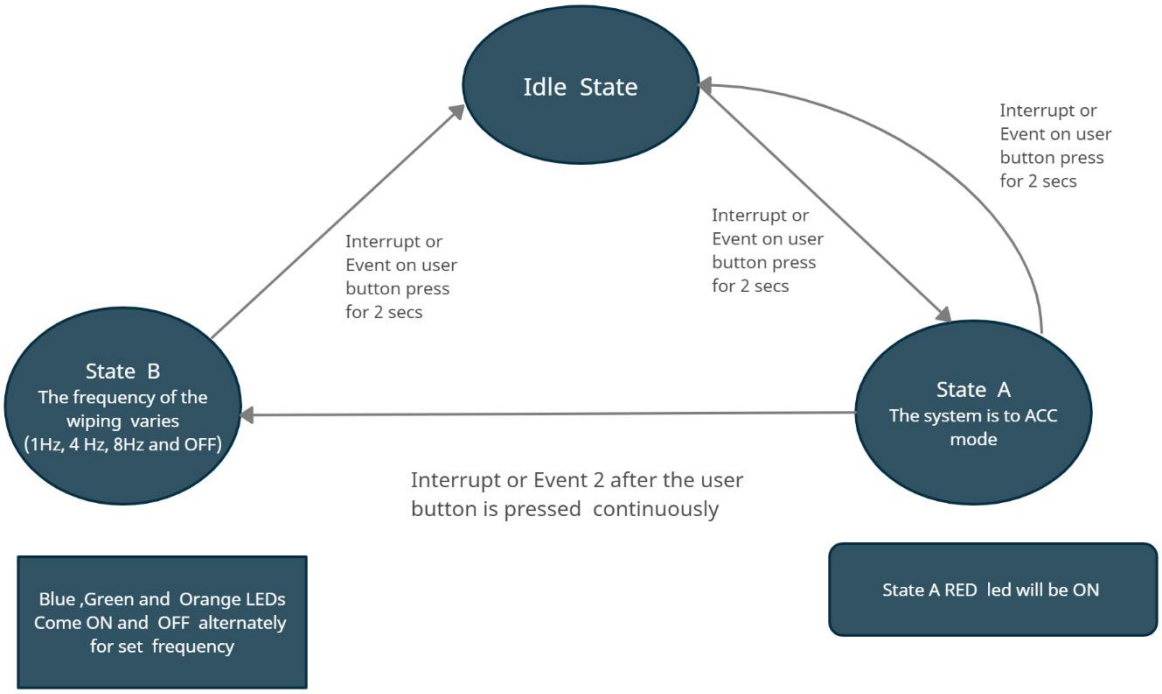

Figure 12 Structure Diagram

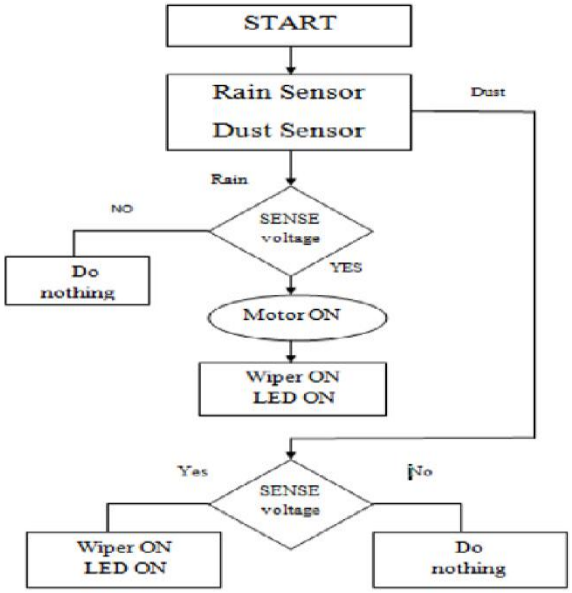

Figure 13 Behavior Diagram

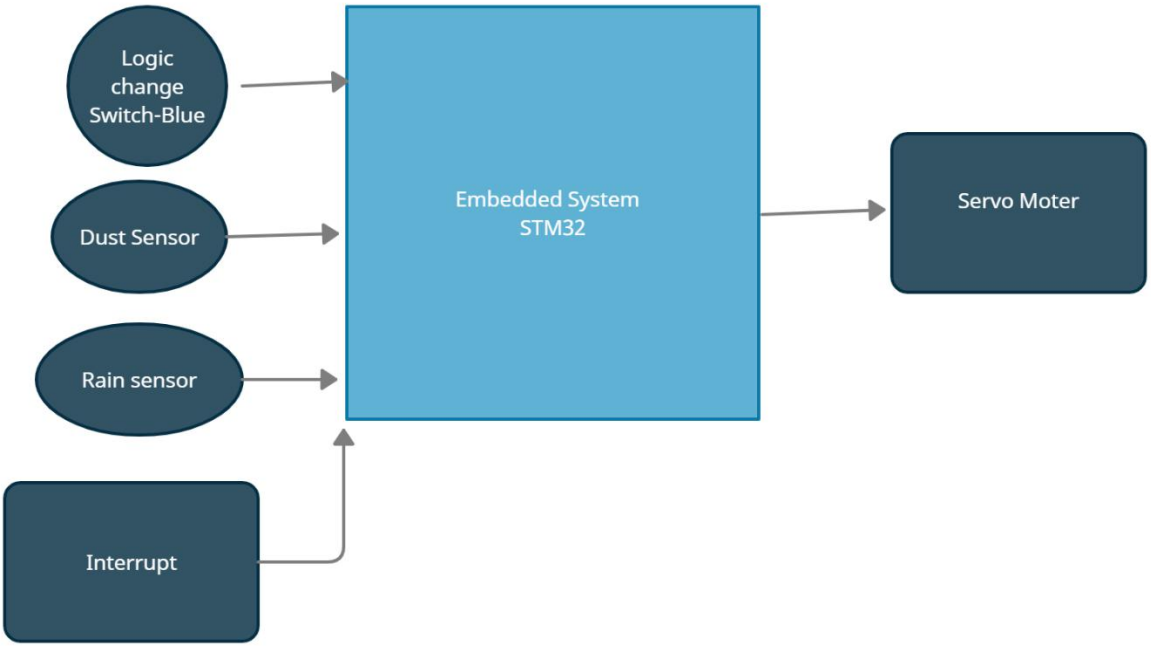

Figure 14 Block Diagram

Test Plan

High Level Test Plan

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 Test Plan

ID	Description

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,similerly the dust particals detected and wiped off.

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. Tester: Writing Testcases and testing the integrated code

Miniproject 7 – Hyundai Project[Team]

Modules

1. Automotive Systems
2. Git

Requirements

In this Jaguar project we have taken following features and I have contributed to Parking System Feature

1. Power Window.
2. Cabin Lights
3. Central Door Lock System.
4. Keyless Entry System.

Design: Cabin Lights (Individual contribution):

- This feature turns on the cabin lights when the doors are unlocked and turns off the cabin lights when the doors are locked.
- A door lock reminder flashes when the doors are not properly shut.

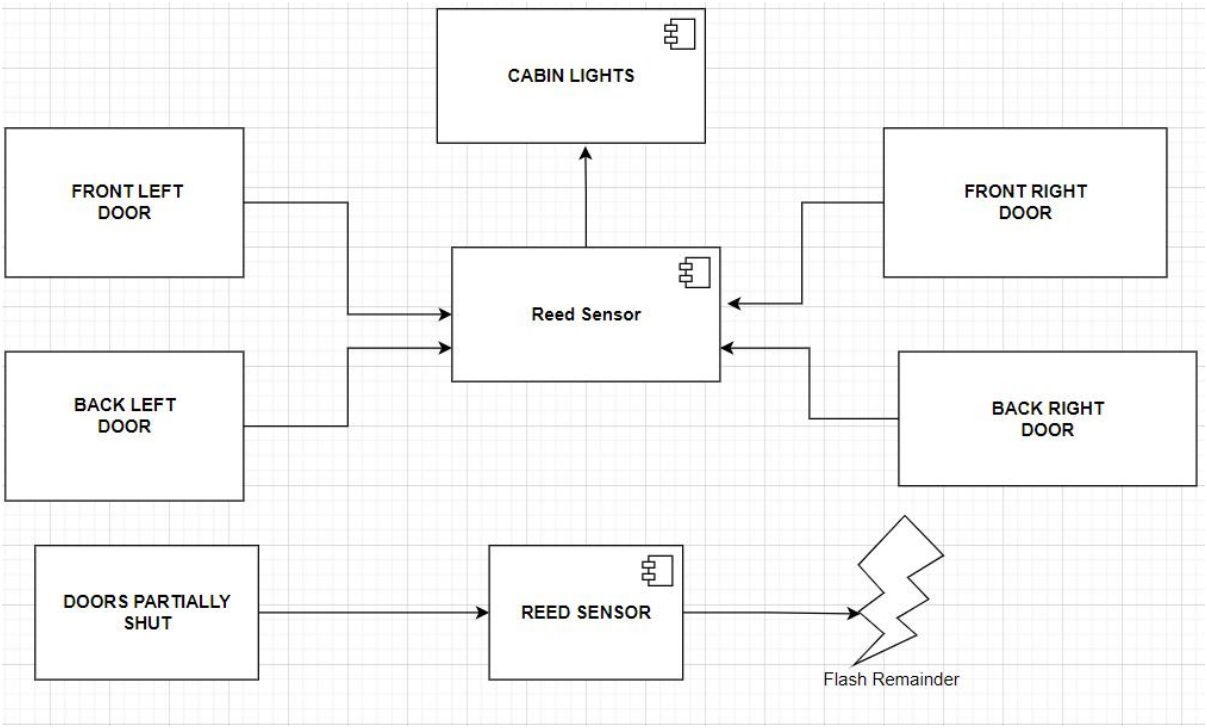

Figure 17 Structure Diagram

Implementation and Summary

Git Link:

Link: https://github.com/SHANMUGAAPRIYANM/TEAM_Hyundai-.git

Individual Contribution and Highlights

1. Door System Case Study
2. Source code management using GitHub

Role in Project Team

1. Designer: Done Designing for Project
2. Researcher: Done case study for Keyless Entry

Miniproject 8 – EV Golf Cart[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 GEALearn

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 – Cabin Lights [Individual]

Modules

1. Autosar
2. Git

Requirements

- This feature turns on the cabin lights when the doors are unlocked and turns off the cabin lights when the doors are locked.
- A door lock reminder flashes when the doors are not properly shut.

Design

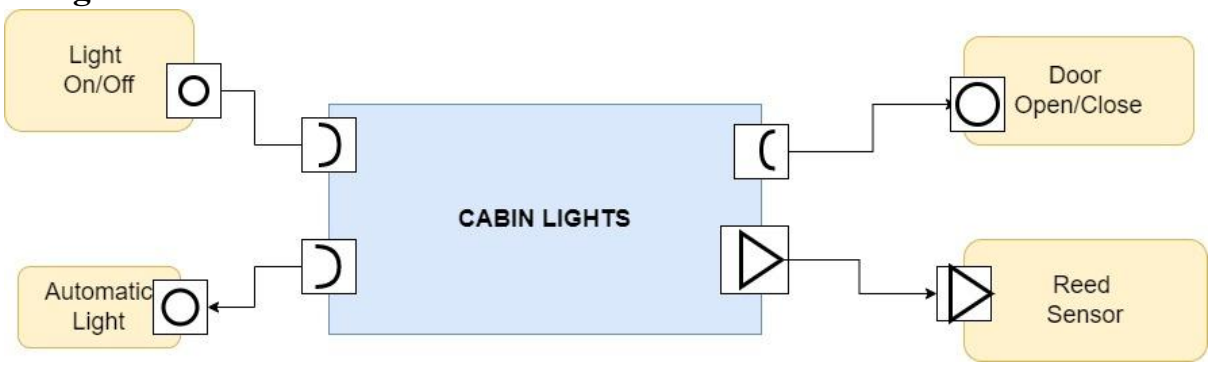

Figure 16 VFB Diagram

Implementation and Summary

Git Link:

Link: https://github.com/vishnukumar25/DoorSystem_40020488_DPS

Individual Contribution and Highlights

1. Studied cabin lights Case Study
2. Source code management using GitHub
3. AtomicSwComponent
4. SWCInternalBehavior
5. SWCImplementation
6.