# GENESIS - Learning Outcome & Mini-project Summary Report

L&T Technology Services

## Details

| Ver. Rel. No. | Release Date | Prepared By | Reviewed By | To Be Approved | Remarks/Revision Details |
|---|---|---|---|---|---|
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | C Programming on Multiple Platforms | | |
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | Essentials of Embedded System | | |
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | Applied SDLC and Software Testing | | |
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | OOPS with Python | | |
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | Applied Model Based Design Module | | |
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | Mastering Microcontrollers with Embedded Driver Development Module | | |
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | Overview of Automotive Systems | | |
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | Applied Control Systems and Vehicle Dynamics | | |
| 1.0 | 18/02/2022 | Kavin Vignes J K 40020494 | Classic Autosar Basic to Intermediate | | |

# Contents

## List of Figures

# Miniproject – 1: Bank Management System [Individual]

## Modules:

1. C Programming
2. Git

## Requirements

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

**Why:**

1. The interface is user-friendly and simple to navigate.
2. Customers can perform most banking operations without or with minimal help.

**Where:**

1. It can be used by a person from anywhere.

**Who:**

1. An individual person only can get access to it.
2. Small/Medium size bank can use this application.

**When:**

1. It can be used at any time depend upon the person's preferences.
2. This problem emerges when the customer base of bank increased rapidly, and multiple transactions take place simultaneously.

**How:**

1. Entering the credentials, you can get access to the banking system.
2. It will help banks to keep their customer account updated and will also enable customer to make a withdrawal.

## High Level Requirements

| ID | Description | Category | Status |
|---|---|---|---|
| HLR_1 | Bank Management System | Technical | Implemented |

## Low Level Requirements

| ID | Description | HLR ID | Status |
|---|---|---|---|
| LLR_1 | Login System Admin | HLR_1 | Implemented |
| LLR_1.1 | Login System Customer | HLR_1 | Implemented |
| LLR_2 | Account Management System | HLR_1 | Implemented |
| LLR_2.1 | Withdrawal mechanism | HLR_1 | Implemented |
| LLR_2.2 | Calculator | HLR_1 | Implemented |

## Design


Figure 1 Behaviour Diagram

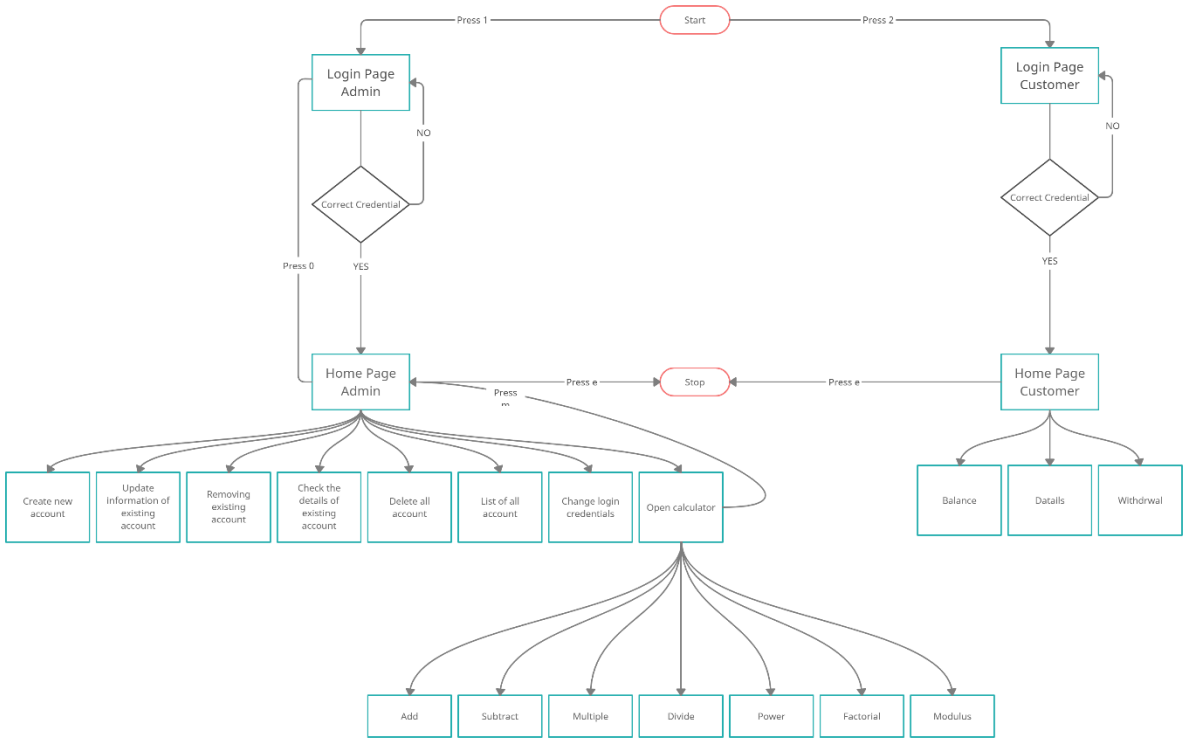



Figure 2 Structure Diagram

# Test Plan

## High Level Test Plan

| Test ID | Description | Exp I/P | Exp O/P | Actual Out | Type of Test |
|---|---|---|---|---|---|
| HLTP_1 | Check the calculator | Two number | One Result | One Result | Requirement Based |
| HLTP_2 | Check the account operations | nil | nil | nil | Requirement Based |
| HLTP_3 | Check the Login credentials | Username & Password | True/False | True/False | Scenario Based |

## Low Level Test Plan

| Test ID | Description | Exp I/P | Exp O/P | Actual Out | Type of Test |
|---|---|---|---|---|---|
| LLTP_1 | Addition | 1,2 | 3 | 3 | Requirement Based |
| LLTP_2 | Subtraction | 5,1 | 4 | 4 | Requirement Based |
| LLTP_3 | Multiplication | 6,3 | 18 | 18 | Requirement Based |
| LLTP_4 | Division | 4,2 | 2 | 2 | Requirement Based |
| LLTP_5 | Power | 2,3 | 8 | 8 | Requirement Based |
| LLTP_6 | Factorial | 5 | 120 | 120 | Requirement Based |
| LLTP_7 | Create account | Data | create a new account | Success | Requirement Based |
| LLTP_8 | Update account | Data | update the selected data | Success | Requirement Based |

| Test ID | Description | Exp I/P | Exp O/P | Actual Out | Type of Test |
|---|---|---|---|---|---|
| LLTP_9 | Delete account | Data | Delete the account | Success | Requirement Based |
| LLTP_10 | Withdrawal Money | Data | Update Balance | Success | Requirement Based |
| LLTP_11 | Login Admin | Username & Password | True/False | True/False | Scenario Based |
| LLTP_10 | Login Customer | Account number | True/False | True/False | Scenario Based |

## Implementation and Summary

## Git Link:

Link: https://github.com/kavinvignes/M1_Application_Bank_Management_System

## Git Dashboard

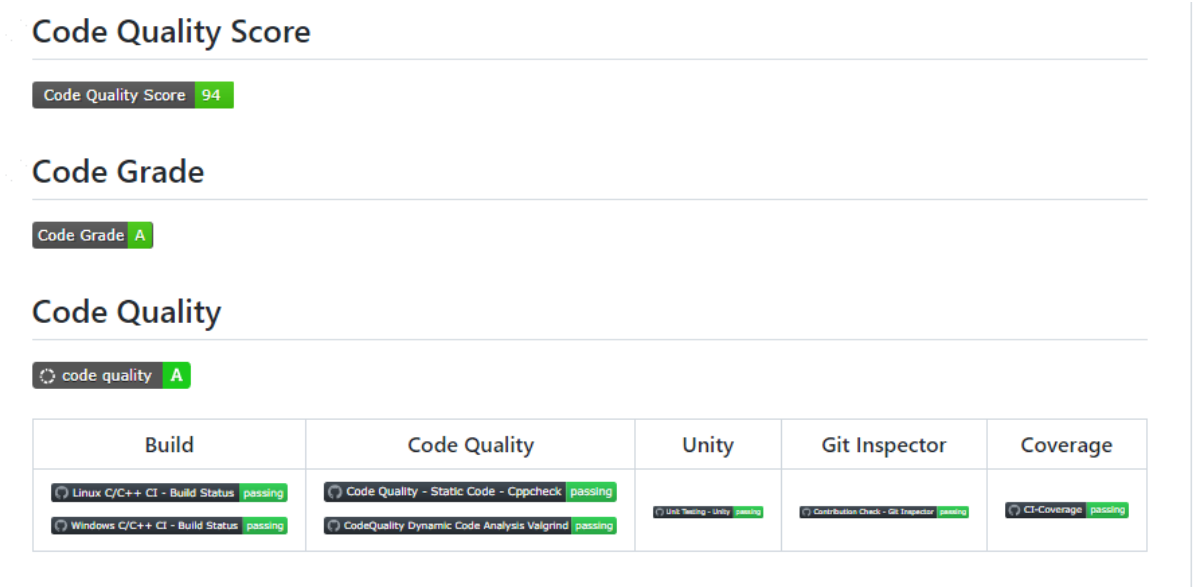



Figure 3 Git Dashboard

## Summary

## Git Inspector Summary

```
23
24  Author                  Commits    Insertions      Deletions    % of changes
25  Kavin Vignes J K            188          7472           1649           99.65
26  kavinvignes                   2            16             16            0.35
27
28  Below are the number of rows from each author that have survived and are still
29  intact in the current revision:
30
31  Author                  Rows      Stability          Age       % in comments
32  Kavin Vignes J K        5217           69.8         12.4               17.23
33  kavinvignes              560         3500.0         11.9                0.00
34
```

Figure 4 Git Inspector Summary

# Miniproject 2 – Embedded Basic Calculator [Individual]

## Modules

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

## Requirements

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

**Why:**

1. To perform the basic numerical operations.
2. To reduce the manpower.

**Where:**

1. The calculator can be operated from anywhere.
2. It can be used in our daily lives.

**Who:**

1. Can be used by anyone is required to solve a numerical operation.
2. Some provisional stores are using calculators for getting the accurate total amount.

**When:**

1. Anyone who have to solve a basic calculation.
2. A person needs do a basic calculation in a fraction of second.

**How:**

1. Giving different inputs we can find their desired output.
2. The results of the calculators will be accurate than the manual calculations.

## High Level Requirements

| ID | Description | Status |
|---|---|---|
| HLR_1 | Control Unit | Implemented |
| HLR_2 | Input Unit | Implemented |
| HLR_3 | Output Unit | Implemented |
| HLR_4 | Software Design | Implemented |

## Low Level Requirements

| ID | Description | HLR ID | Status |
|---|---|---|---|
| LLR_1 | AVR Atmega 328 Microcontroller | HLR_1 | Implemented |
| LLR_2 | 4*4 Keypad Interface | HLR_2 | Implemented |
| LLR_3 | 16*2 LCD Interface | HLR_3 | Implemented |
| LLR_4 | Visual Studio Code & Simulide | HLR_4 | Implemented |

## Design

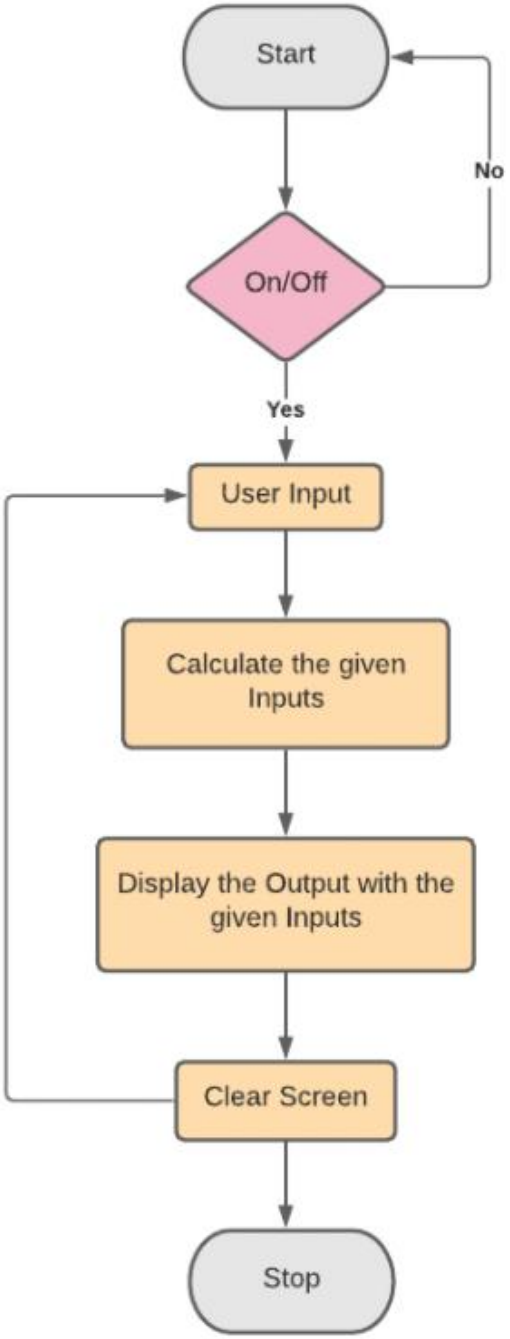



Figure 5 Behaviour Diagram

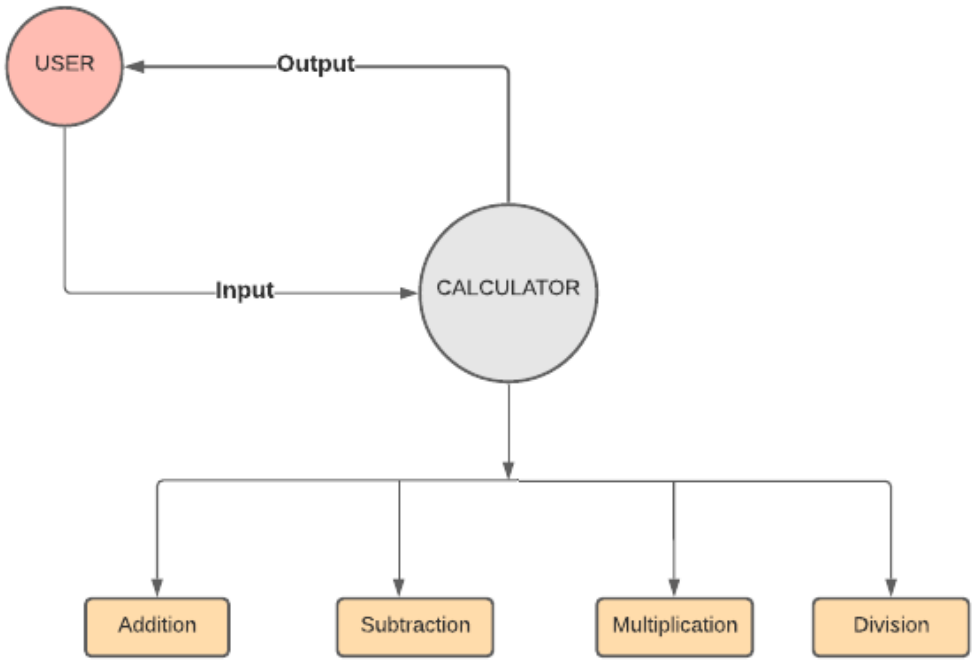



Figure 6 Structure Diagram

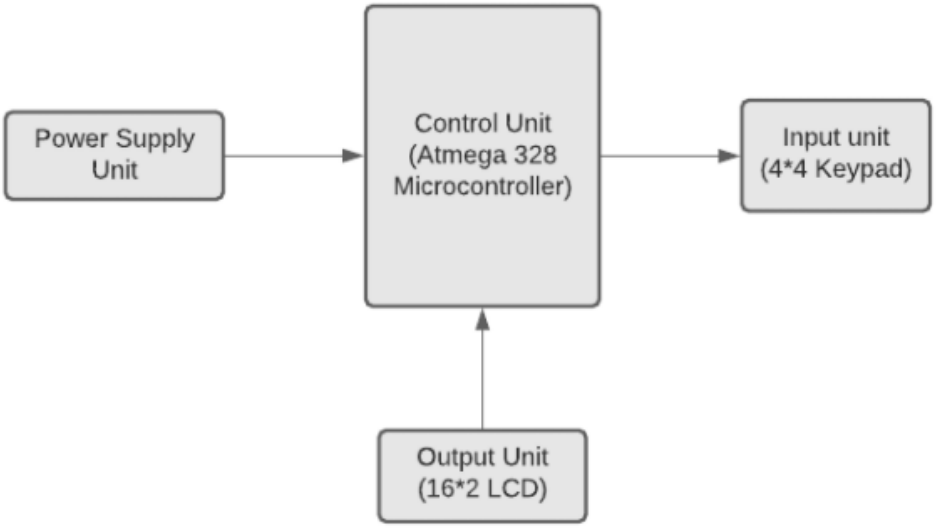



Figure 7 Block Diagram

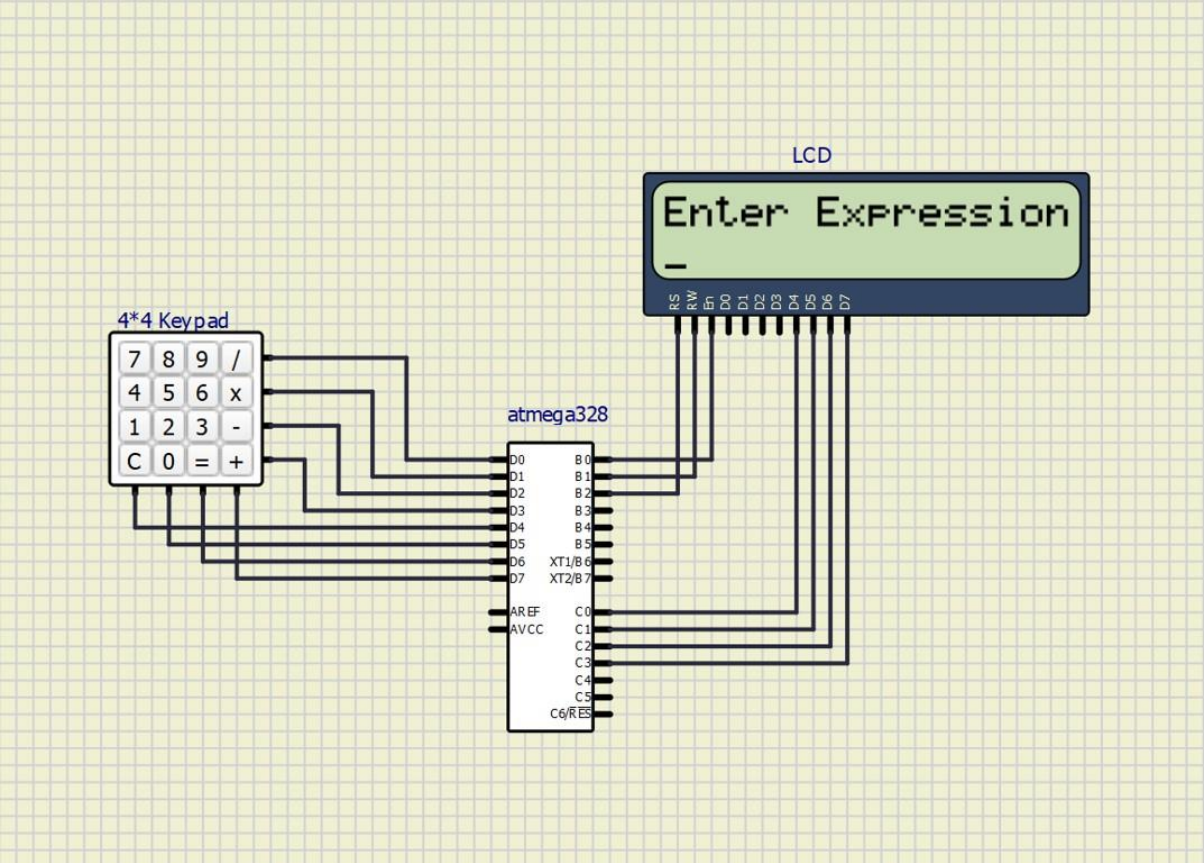



Figure 8 Simulation

## Test Plan

## High Level Test Plan

| Test ID | Description | Exp I/P | Exp O/P | Actual Out | Type of Test |
|---|---|---|---|---|---|
| HLTP_1 | Power ON | Power | Display ON | SUCCESS | Requirement Based |
| HLTP_2 | User Input | Input Value | Return Output to the User | SUCCESS | Scenario Based |
| HLTP_3 | Return Output from Input | Inputted Value by User | Shows Output in Display | SUCCESS | Requirement Based |

## Low Level Test Plan

| Test ID | Description | Exp I/P | Exp O/P | Actual Out | Type of Test |
|---|---|---|---|---|---|
| LLTP_01 | '+' operation | (40, 50) | 90 | 90 | Requirement Based |
| LLTP_02 | '-' operation | (70, 20) | 50 | 50 | Requirement Based |
| LLTP_03 | '*' operation | (5, 5) | 25 | 25 | Requirement Based |
| LLTP_04 | '/' operation | (24, 2) | 12 | 12 | Requirement Based |

## Implementation and Summary

## Git Link:

Link: https://github.com/kavinvignes/M2-Embedded-BasicCalculator

## Git Dashboard

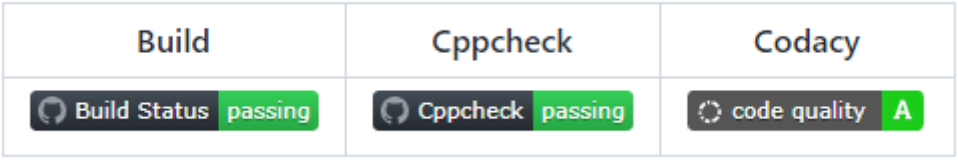



Figure 9 Git Dashboard

# 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

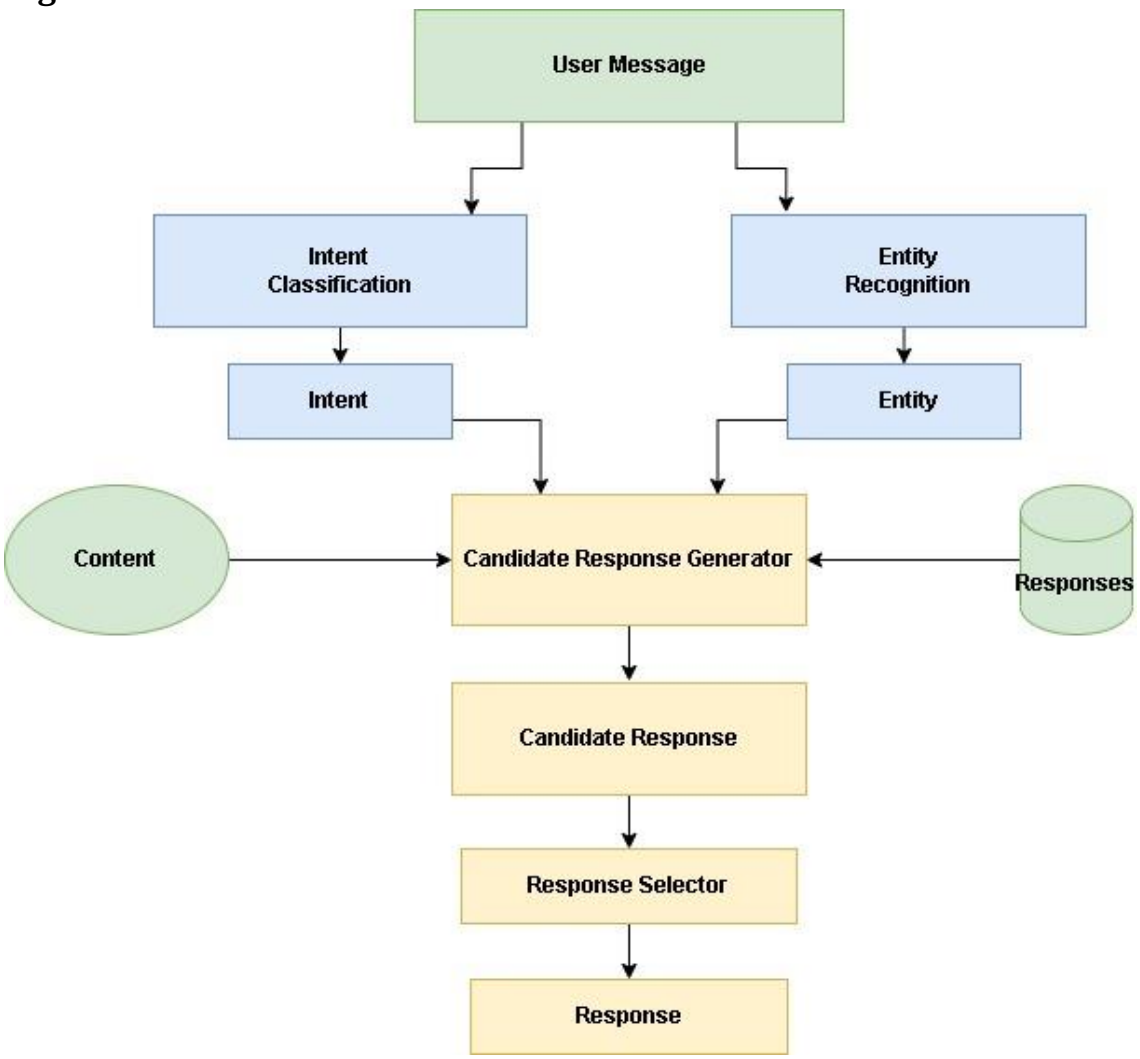



Figure 10 Behaviour Diagram

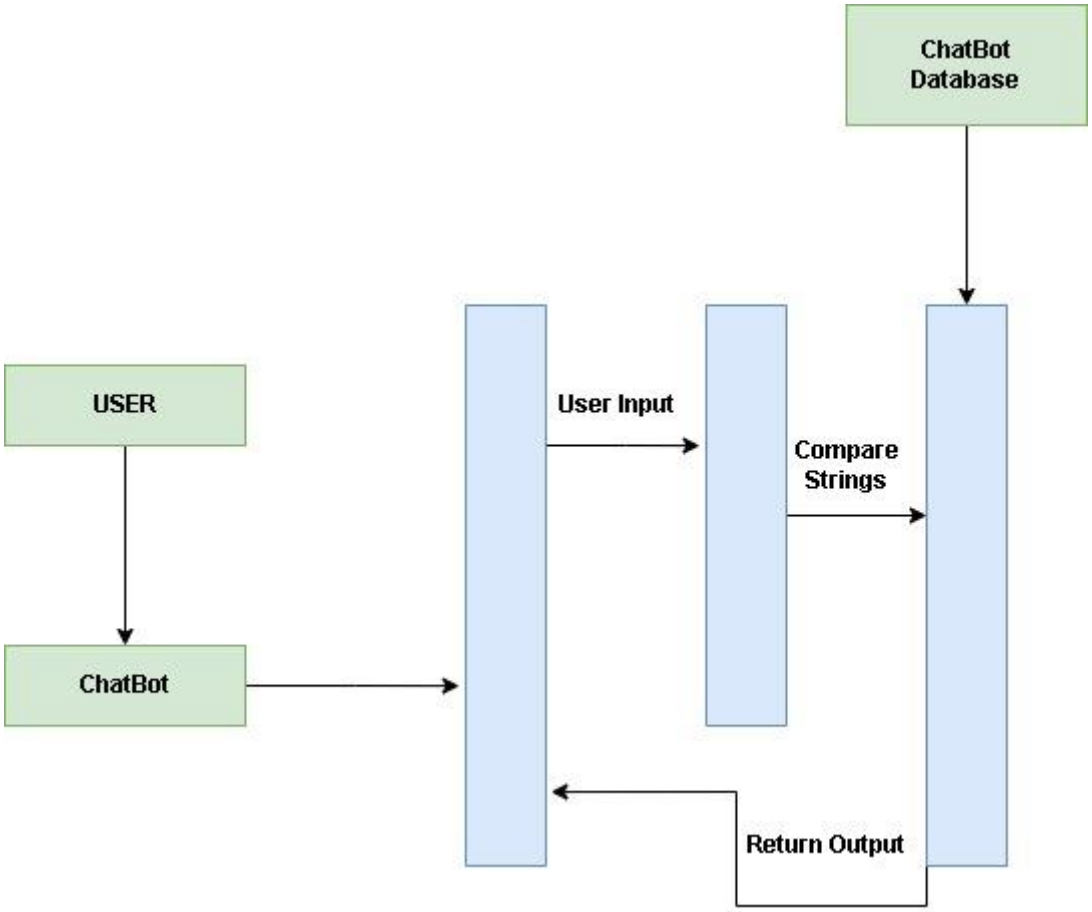



Figure 11 User Flow Diagram

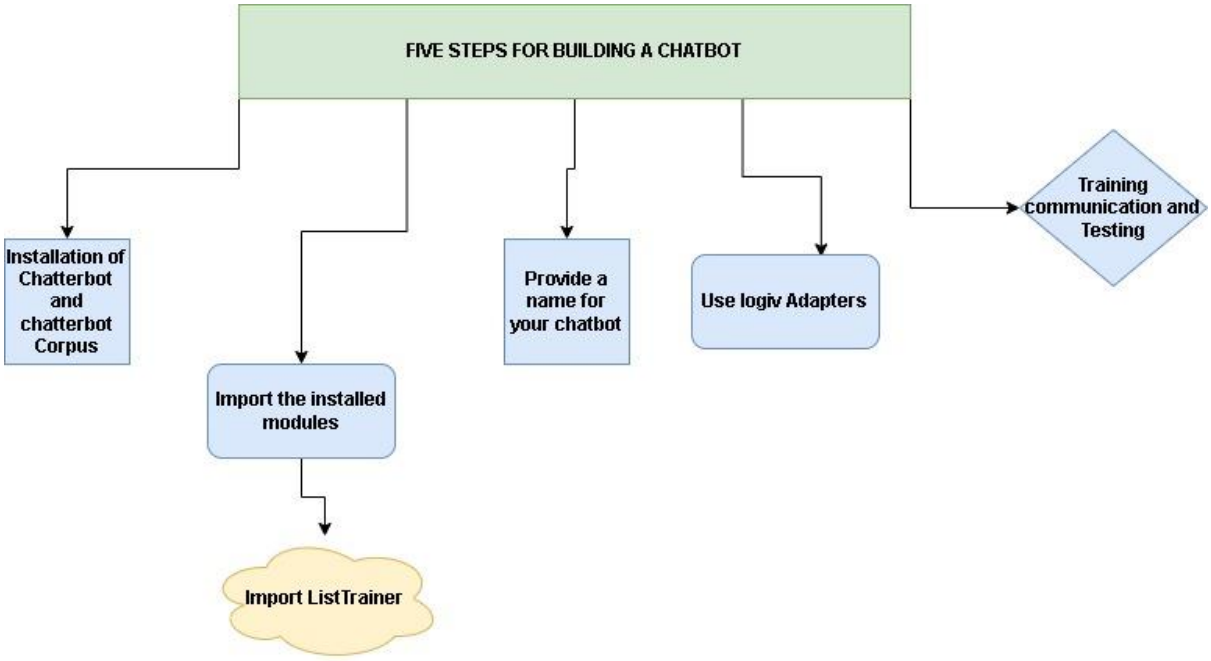



Figure 12 Structure Diagram

## 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_2.1 | HLTP_2 | Return a known response to | Process input | SUCCESS | SUCCESS | Requirement Based |

| Test ID | HLTP ID | Description | Exp IN | Exp OUT | Actual Out | Type Of Test |
|---|---|---|---|---|---|---|
| | | the selected match and a confidence value based on the matching | | | | |
| 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

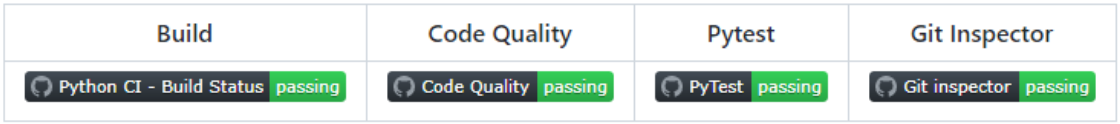
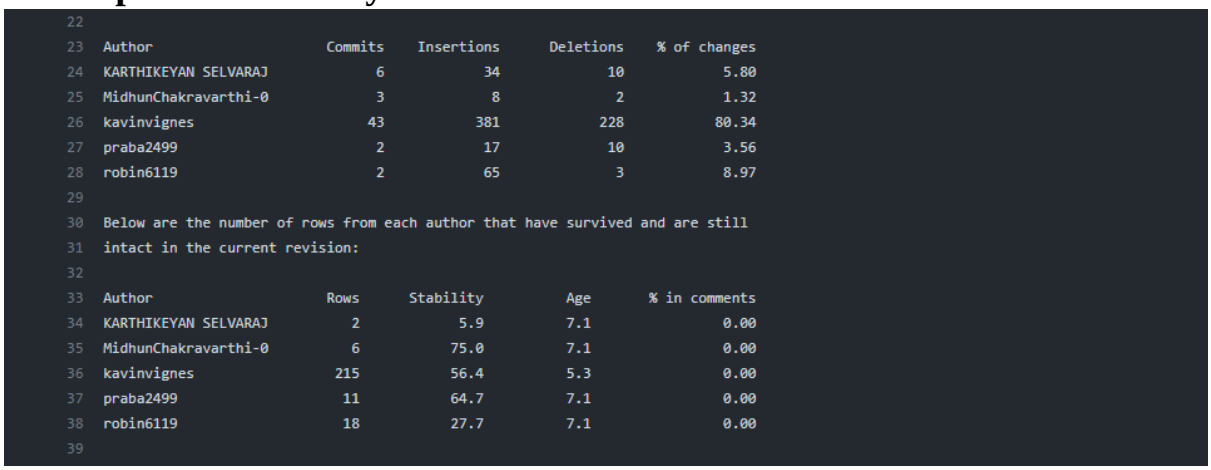



Figure 13 Git Dashboard

## 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

## 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 – Kia 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. Engine Braking System
6. Lane Assist System
7. Power Window

**Anti Lock Braking System:**

Antilock Braking System is employed to prevent slipping and locking of wheels when brake applied. It is a vehicle brake control system, which uses ABS controller to monitor and control the necessary torque to maintain slip ration. Antilock brakes have been introduced to enhance test track braking performance but their effect on crash risk in actual driving stays less. ABS generally offers advanced vehicle control and minimizes the stopping distance in slippery and dry surface.

## Design

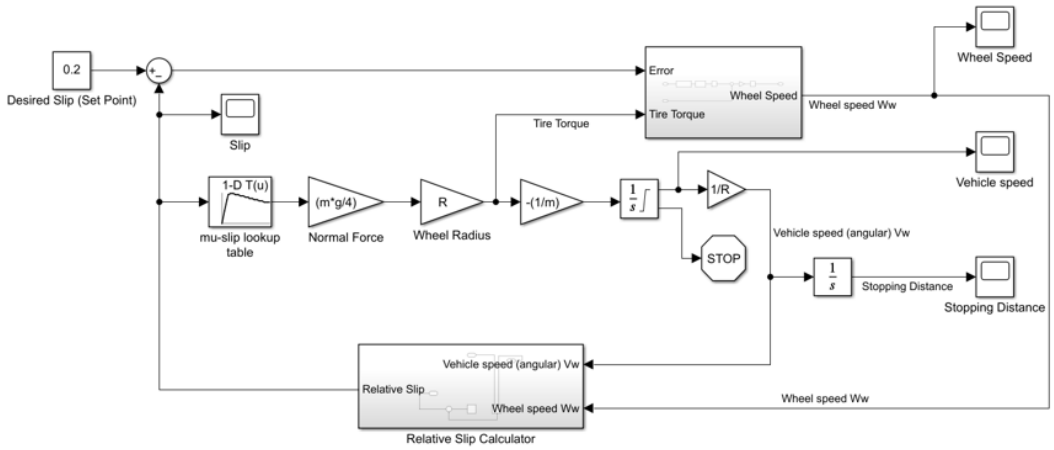



Figure 15 Simulation

## Implementation and Summary

## Git Link:

Link: https://github.com/karthikeyans99/Genesis2021_Applied_Mbd-Kia_Project_Team

## Individual Contribution and Highlights

1. Anti Lock Braking System Case Study
2. Designed and implemented using matlab and the file is in github.

Role in Project Team

1. Designer: Done Designing for Anti lock braking system using matlab.

# Miniproject 6 – Wiper Control [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 | Status |
|---|---|---|
| HLR_1 | Ignition at ACC - Red LED ON | Pass |
| HLR_2 | Wiper ON - LED'S ON in Blue, Green and Orange | Pass |
| HLR_3 | Wiper OFF - LED'S OFF in Blue, Green and Orange | Pass |
| HLR_4 | Ignition at lock - Red LED OFF | Pass |

## Low Level Requirements

| ID | Description | Status |
|---|---|---|
| LLR_1 | Pressing button for two seconds - Red LED ON | Pass |
| LLR_2 | Wiping at 1Hz - Blue LED ON | Pass |
| LLR_3 | Wiping at 4Hz - Green LED ON | Pass |
| LLR_4 | Wiping at 8Hz - Orange LED ON | Pass |
| LLR_5 | Pressing button for two seconds - Red LED OFF | Pass |

## Design

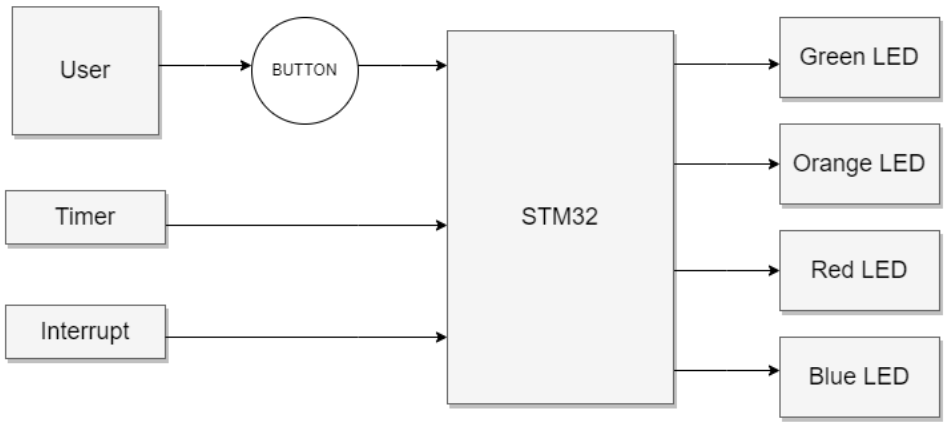



Figure 16 Structure Diagram

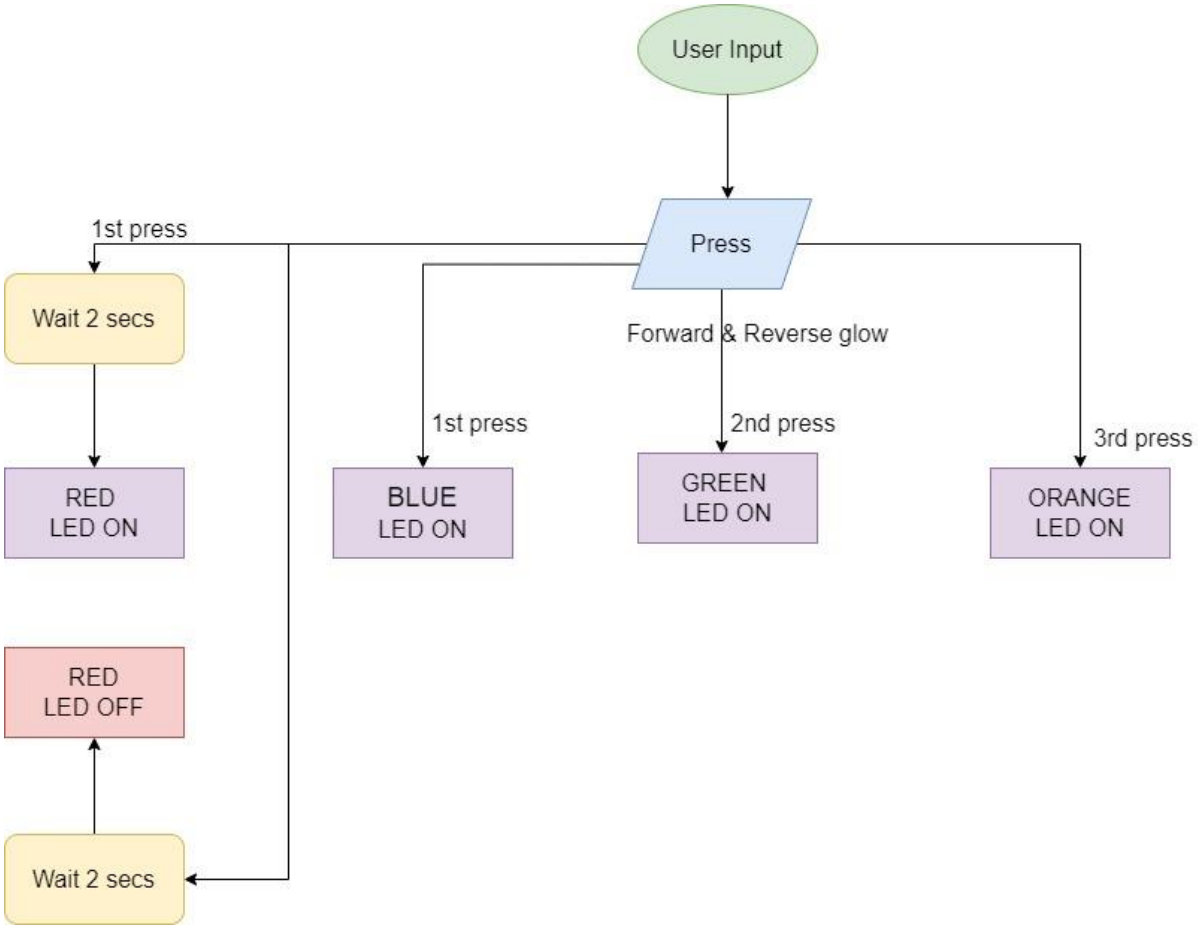



Figure 17 Behaviour Diagram

## 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 |
| 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-Team23

## 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 – Jeep Compass [Team]

## Modules

1. Automotive Systems
2. Git

## Requirements

Door System is a type of door opening, typically hinged on its front edge, but sometimes attached by other mechanisms such as tracks, for entering and exiting a vehicle. Doors most often integrate side windows for visibility from inside the car and can be locked to secure the vehicle. The door system available in this car are,

1. Power Door Locks
2. Passive Keyless Entry
3. Automatic Unlock Doors on Exit
4. Power Window System

## High Level Requirements

| S.No. | Feature | Description |
|---|---|---|
| HLR_1 | Power Door Lock | Driver or front passenger can lock or unlock the doors. |
| HLR_2 | Passive Keyless Entry | Allows you to unlock and lock the doors to a vehicle without using a key. |
| HLR_3 | Automatic Unlock Door on Exit | The door locks will unlock automatically. |
| HLR_4 | Power Window System | All windows can be accessed by switching on the passenger door trim panel. |

## Low Level Requirements

| S.No. | Feature | Description |
|---|---|---|
| LLR_1 | Power Door Lock | It automatically locks doors at certain speeds. |
| LLR_1.1 | Power Door Lock | To secure all the doors at the same time. |
| LLR_2 | Passive Keyless Entry | Doors will be lock or unlock by using a radio frequency remote keyless system. |
| LLR_2.1 | Passive Keyless Entry | If the vehicle is unlocked by Passive Entry and no door is opened within 60 seconds, the vehicle will re-lock. |
| LLR_3 | Automatic Unlock Door on Exit | If the vehicle is unlocked by Passive Entry and no door is opened within 60 seconds, the vehicle will re-lock. |
| LLR_4 | Power Window System | The windows will be open and close within 4 seconds. |

## Design

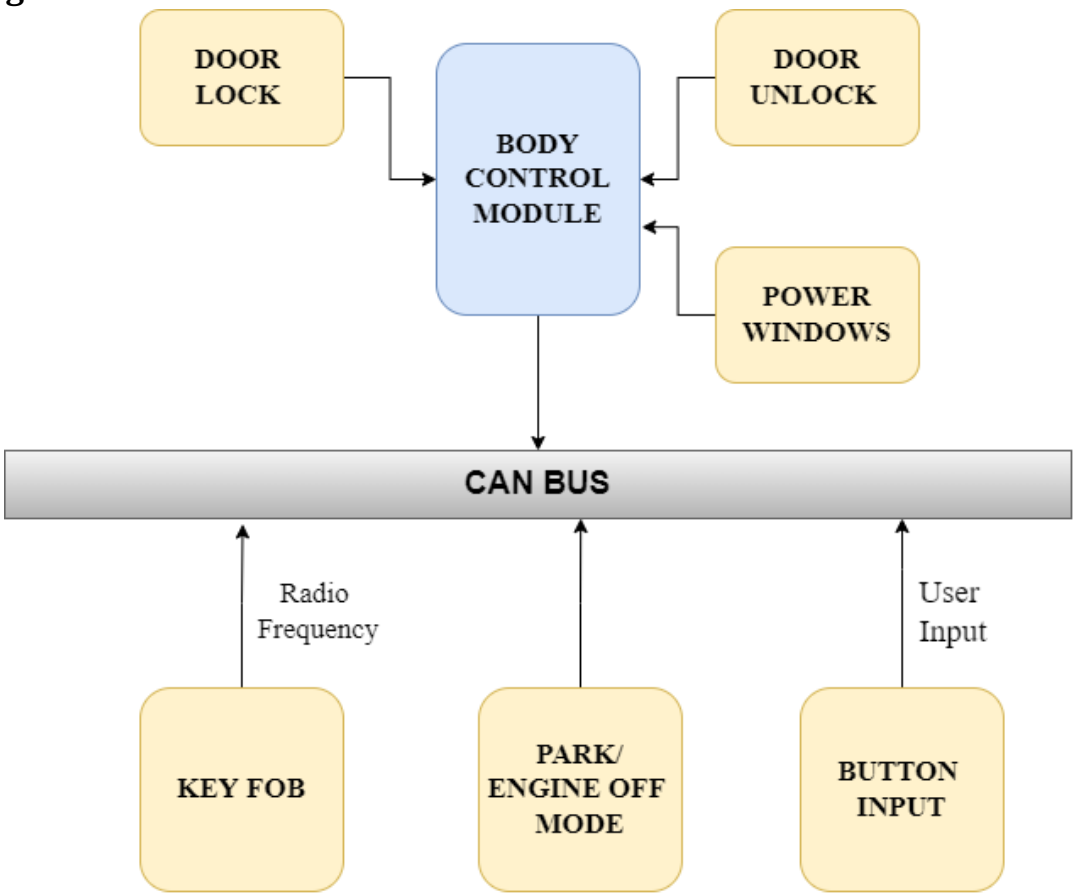



Figure 18 Structure Diagram

## Implementation and Summary

## Git Link:

Link: https://github.com/karthikeyans99/Jeep_Compass_Team

## 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 Car [Team]

## Modules

1. Matlab
2. Matlab Script

## Requirements

## Motor Specifications:

| Component | Tata Nexon | Hyundai Kona |
|---|---|---|
| Top speed | 120kmh | 165kmh |
| Acceleration(0-100kmph) | 9.14s | 7.9s |
| Engine type | PMSM | PMSM |
| Max motor performance | 127bhp 245Nm | 134 bhp 395 Nm |
| Driving Range | 312 kms | 450kms |
| Battery | 30.2 kWh, Lithium-Ion Polymer, 320V Battery Placed Under Floor Pan | 39.2 kWh, Lithium-Ion Polymer, 327V Battery Placed Under Floor Pan |

## Safety Specifications:

| Component | Tata Nexon | Hyundai Kona |
|---|---|---|
| Overspeed Warning | 1 beep over 80kmph, Continuous beeps over 120kmph | 1 beep over 80kmph, Continuous beeps over 120kmph |
| Emergency Brake Light Flashing | No | Yes |
| NCAP Rating | 5 Star (Global NCAP) | 5 Star (Euro NCAP) |

| Component | Tata Nexon | Hyundai Kona |
|---|---|---|
| Airbags | 2 Airbags (Driver, Passenger) | 6 Airbags (Driver, Passenger, 2 Curtain, Driver Side, Front Passenger Side) |
| Middle Rear Head Rest | No | Yes |

## Battery Performance:

1. Both cars use the same Lithium-ion battery type as it is the industry standard right now.
2. Hyundai Kona has a massive 452 km lead in terms of range which is more than double of what the Tata Nexon.
3. Battery charging times are longer in the Hyundai Kona due to its larger 39.1 kWh battery compared to the 37.4 kWh.
4. Hyundai Kona offer fast charging.

## Braking Performance:

1. Hyundai Kona has ESP, not in Tata Nexon.
2. Huyndai Kona has TCS, not in Tata Nexon.

## Suspension Performance:

1. Hyundai kona has Independent MacPherson strut with coil spring and Tata nexon has McPherson Strut Type front suspension.
2. Hyundai kona has Twist beam with dual path Strut and Tata nexon has multi-Link rear suspension.

## Implementation and Summary
## Git Link:

Link: https://github.com/kavinvignes/Team4_EV_Car

Matlab file Submission: Submitted in GEALearn

## Individual Contribution and Highlights

1. Done in Matlab Scripting

Role in Project Team

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

# Miniproject 9 – Power Window [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 | Power Window System | All windows can be accessed by switching on the passenger door trim panel. |

## Low Level Requirements:

| S.No. | Feature | Description |
|---|---|---|
| LLR_1 | Power Window System | The windows will be open and close within 4 seconds. |

## Design

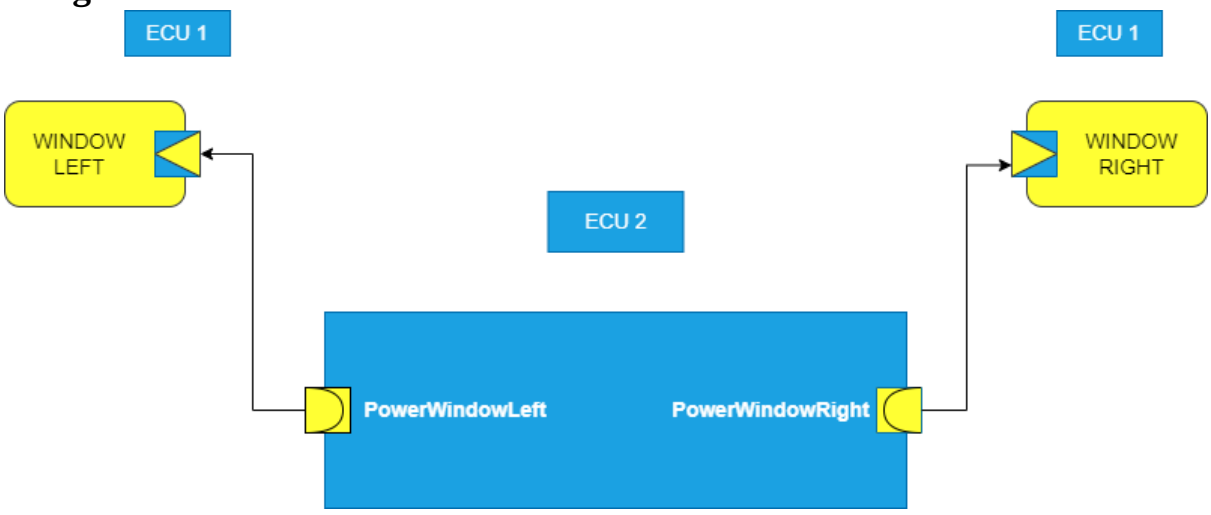



Figure 19 VFB Diagram

## Implementation and Summary

## Git Link:

Link: https://github.com/kavinvignes/PowerWindow_40020494_DPS

## Individual Contribution and Highlights

1. Power Window Case Study
2. Source code management using GitHub
3. AtomicSwComponent
4. SWCInternalBehavior
5. SWCImplementation