# Details

| Ver. Rel. No. | Release Date | Prepared By | Reviewed By | To Be Approved | Remarks/Revision Details |
|---|---|---|---|---|---|
| 1.0 | 16/02/2022 | Milind Gautam Bagul 40021135 | | | |

# Contents

## List of Figures

# Miniproject – 1: Tik Tac Toe Game [Individual]

## Modules:

1. C Programming
2. Git

## Requirements

### 4W's and 1 H's

**What:**

1. This is a purely leisure game. Because there are so many different outcomes in this game, businesses can utilise it to design strategies.

**Where:**

1. A simple tic tac toe game is accessible on a number of websites. In addition, corporations and organisations use it.

**Who:**

1. This is a game anybody can play.

**When:**

1. This game can be played if you're bored or want to learn more about the game's methods, consequences, and scenarios. game.

**How:**

1. ***Blocks your opponent from winning as you try to win.***

## High Level Requirements

| ID | Description | Status |
|---|---|---|
| HLR_1 | Users can use a web browser to obtain the information | Implemented |
| HLR_2 | From the landing page, the user should choose the game's difficulty level and begin playing | Implemented |
| HLR_3 | When a user moves, the game page allows them to move | Implemented |
| HLR_4 | The user can see the opponent's movements in real time on the game page | Implemented |
| HLR_5 | The user can pick up where they left off in a game that isn't yet finished | Implemented |
| HLR_6 | When one player gets three symbols in a row, the game should be over | Implemented |
| HLR_7 | After the game, the user sees the results | Implemented |

## Low Level

## Requirements

| ID | Description | Status |
|---|---|---|
| LLR_1 | Name of the player | Implemented |
| LLR_2 | Players personal details like gender,contact number | Implemented |

## Design

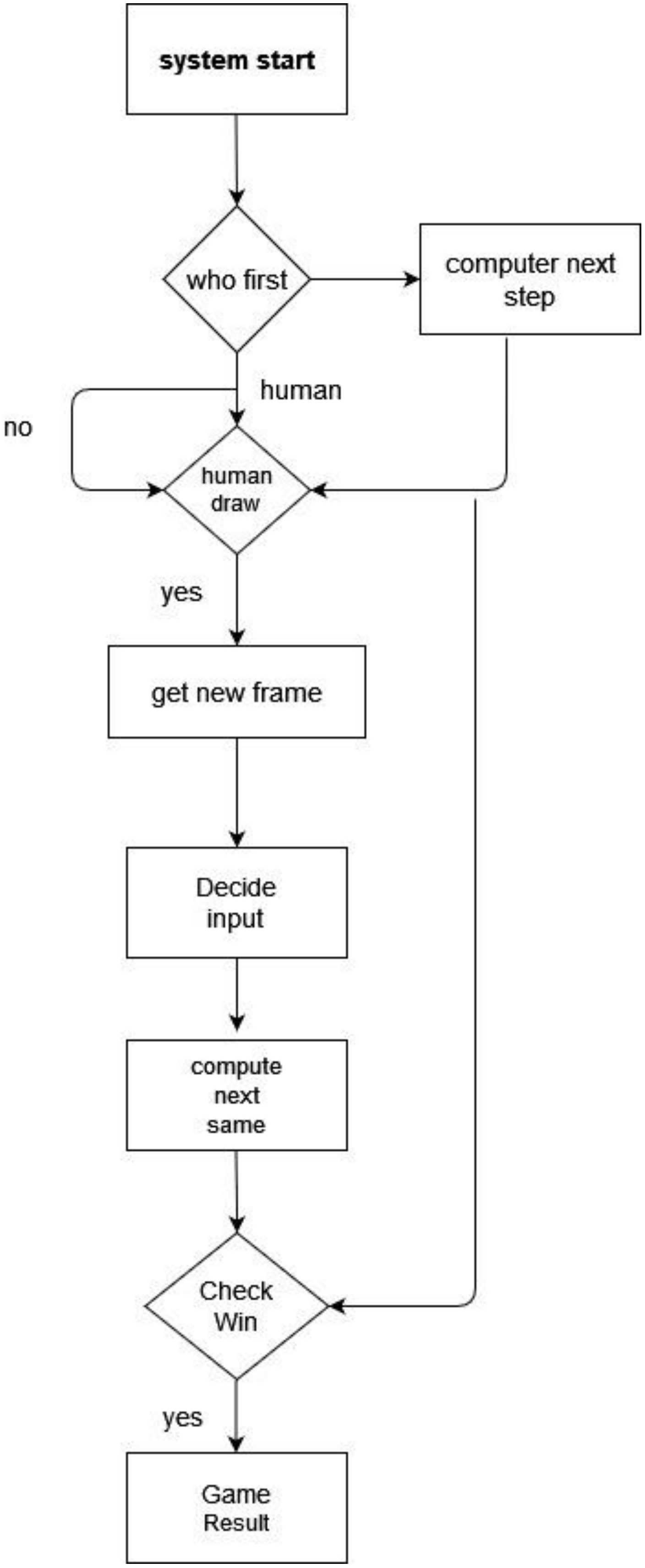
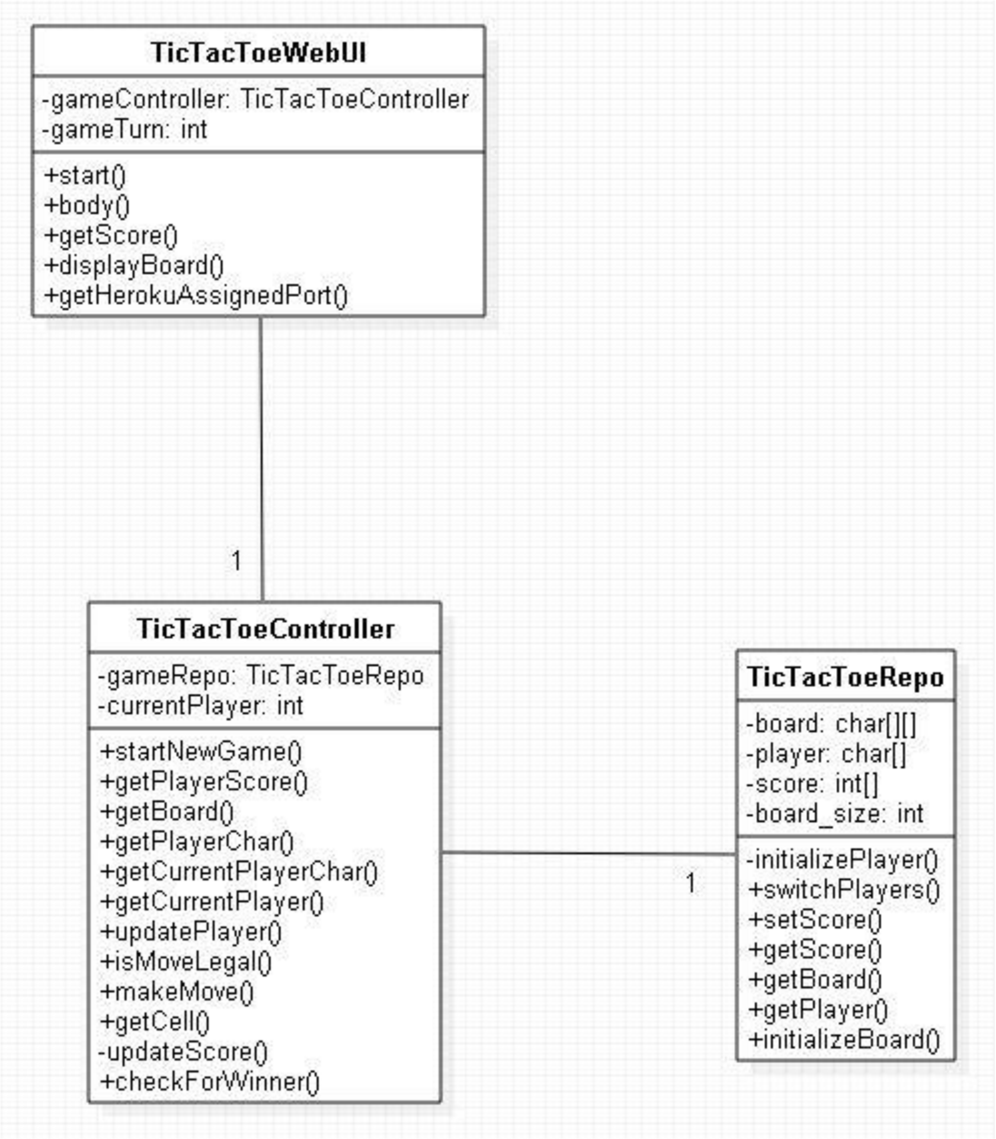


Figure 1 Behavior Diagram

## Test Plan

## Test Plan

We gathered all the elements needed for the game to work and created tests to test our implementation. Each addition to our project was then pushed to our Git repository to save files. Depending on the result of the build, we either fixed our code or moved to the next step. when the cryptography section, laptop programs square measure obtainable which will be dead for testing purpose.

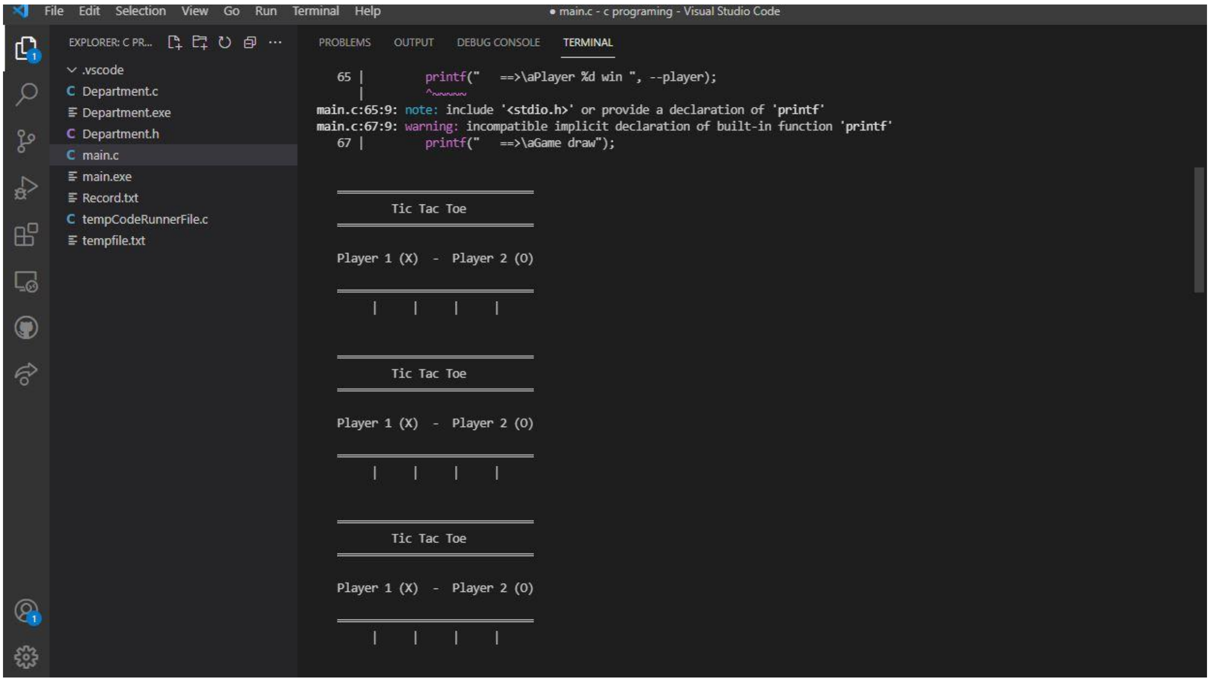


## Implementation and Summary

## Git Link:

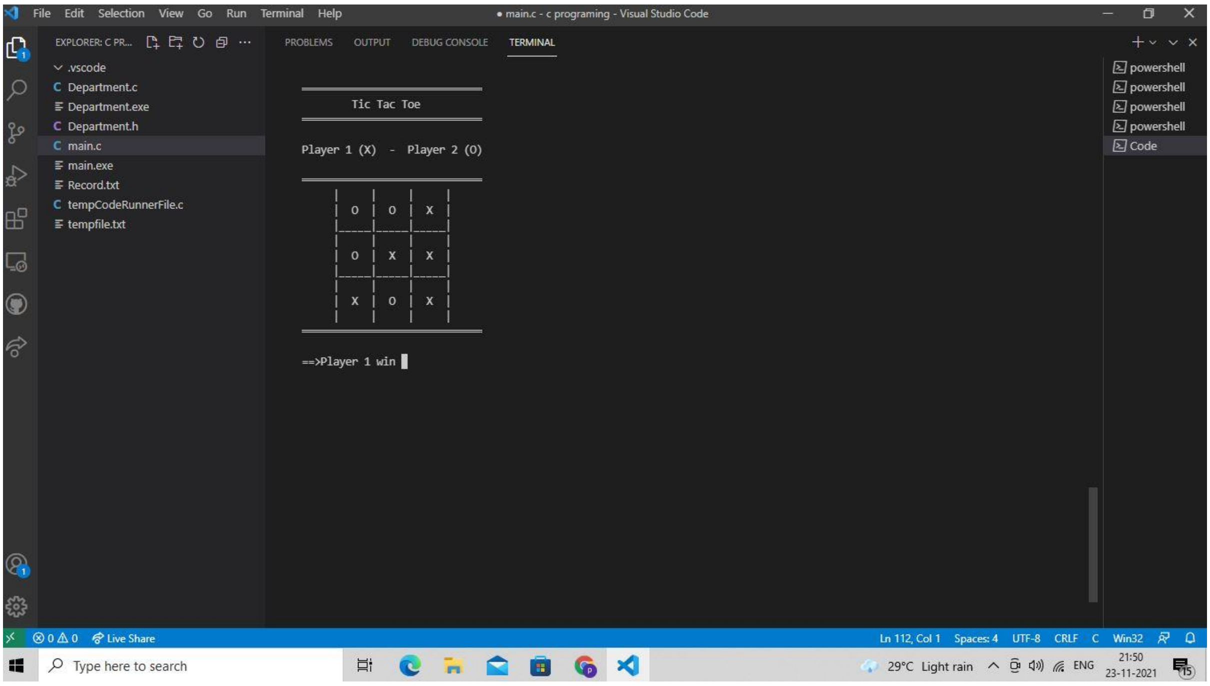
Link: milixx21/M1_game_tic-tac-toe- (github.com)

## TEST CASE



```
   65 |         printf("   ==>\aPlayer %d win ", --player);
      |         ^~~~~~
main.c:65:9: note: include '<stdio.h>' or provide a declaration of 'printf'
main.c:67:9: warning: incompatible implicit declaration of built-in function 'printf'
   67 |         printf("   ==>\aGame draw");


   ═══════════════════════════

         Tic Tac Toe

   ═══════════════════════════


   Player 1 (X)  -  Player 2 (O)


   ═══════════════════════════

        |     |     |     |


   ═══════════════════════════

         Tic Tac Toe

   ═══════════════════════════


   Player 1 (X)  -  Player 2 (O)


   ═══════════════════════════

        |     |     |     |


   ═══════════════════════════

         Tic Tac Toe

   ═══════════════════════════


   Player 1 (X)  -  Player 2 (O)


   ═══════════════════════════

        |     |     |     |
```

Output 1



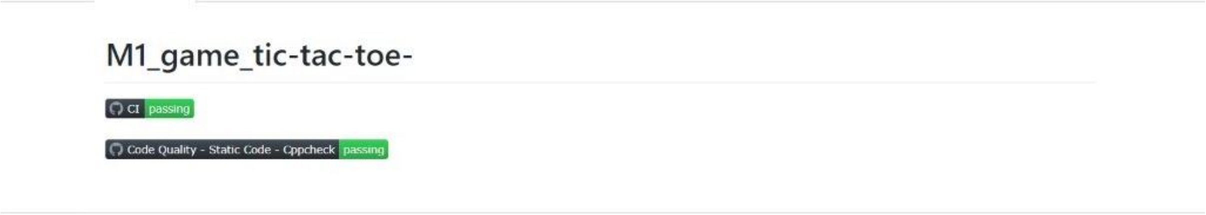
```
   ═══════════════════════════

         Tic Tac Toe

   ═══════════════════════════


   Player 1 (X)  -  Player 2 (O)


   ═══════════════════════════

        |     |     |     |
        |  O  |  O  |  X  |
        |_____|_____|_____|
        |     |     |     |
        |  O  |  X  |  X  |
        |_____|_____|_____|
        |     |     |     |
        |  X  |  O  |  X  |
        |     |     |     |

   ═══════════════════════════


   ==>Player 1 win █
```

Output 2

## CERTIFICATION DONE IN MODULE

- SOLO-Learn Certification
- Linux Certification
- Github Learning Certification

## Git Dashboard



M1_game_tic-tac-toe-

CI passing

Code Quality - Static Code - Cppcheck passing

Figure 2 Git Dashboard

**Mini project 2 – Ultrasonic Sound Sensor with Atmega328 Microprocessor [Individual]**

**Module: - Essentials of Embedded System**

**Topic: - ULTRASONIC SOUND SENSOR WITH ATmega328 MICROPROCESSOR**

**Requirements**

## Introduction

The project as the name suggests is based on Ultrasonic sensors. Ultrasonic sensors work by sending out a sound wave at a frequency above the range of human hearing. Our ultrasonic sensors, like many others, use a single transducer to send a pulse and to receive the echo. The sensor determines the distance to a target by measuring time lapses between the sending and receiving of the ultrasonic pulse.

**Features, Hardware and Software:-**

## a) HARDWARE:-

1] SimulIDE:

- SimulIDE provides AVR, Arduino and PIC microcontrollers that can be accessed just like other components.
- Features like gypsum and simavr allow you to use PIC and AVR microcontrollers, respectively.

2] AVR:

- An automatic voltage regulator (AVR) is an electronic device that maintains a constant voltage level to electrical equipment on the same load.
- The AVR regulates voltage variations to deliver constant, reliable power supply.

## b) SOFTWARE:-

1] ATmega328:

- ATmega328 is commonly used in many projects and autonomous systems where a simple, low-powered, low-cost micro-controller is needed

- Perhaps the most common implementation of this chip is on the popular Arduino development platform.

2] Sound:

- A sound sensor is defined as a module that detects sound waves through its intensity and converting it to electrical signals.

3] Display:

- A display device is an output device for presentation of information in visual or tactile form.

## SWOT ANALYSIS:-

**d) Strength:**

The distance to an obstacle can be measured with the low cost ultrasonic sensor. The sensors can measure distances from 2 to 400cm with an accuracy of 3mm. This sensors module includes ultrasonic transmitter, ultrasonic receiver and control circuit.

**b) Weakness:**

Although we fully believe in the capability of our sensors, we understand that ultrasonic are not suited for every application. Focuses of low thickness, similar to froth and fabric, have a tendency to assimilate sound vitality; these materials may be hard to sense at long range.

**c) Opportunity:**

This project can be used as parking assistance systems in vehicles with high power ultrasonic transmitter. This Project Can be used as burglar alarm with suitable additional software for homes and offices.

**d) Threats:**

Ultrasonic sensors must view a surface (particularly a hard, level surface) unequivocally (oppositely) to get adequate sound reverberation. Additionally, solid detecting requires a base target surface range, which is indicated for every sensor sort. If connection is wrong there might be chances of short-circuit.

**4W's a 1H:-**

- **What:**

We have made a setup based on a microcontroller in which real time distance is sensed by an ultrasonic sensor and displays measured distance on an LCD display.

- **Where:**

It measures accurate distance using a non-contact technology - A technology that involves no physical contact between sensor and object.

-3 When: In 1959, Satomura created an ultrasonic flow meter that used Doppler technology.

-# Why: I am Developing this project for easily measure the distance between objects

- **How:**

By using Atmega328 and display an ultrasonic sensor mainly used to determine the distance of the target object.

**High Level Requirements**

| **ID** | **Description** |
|---|---|
| HLR1 | Used to avoid and detect obstacles with robots like biped robot, obstacle avoider robot, path finding robot etc. |
| HLR2 | Used to measure the distance within a wide range of 2cm to 400cm |
| HLR3 | Depth of certain places like wells, pits etc can be measured since the waves can penetrate through water |

**Low Level Requirements**

| **ID** | **Description** |
|---|---|
| LLR_1 | • Power Supply: +5V DC. |

| ID | Description |
|---|---|
| LLR_2 | • Measuring Angle: 30 degree. |
| LLR_3 | • Trigger Input Pulse width: 10uS TTL pulse. |
| LLR_4 | • Depth of certain places like wells, pits etc can be measured since the waves can penetrate through water. |

## Design

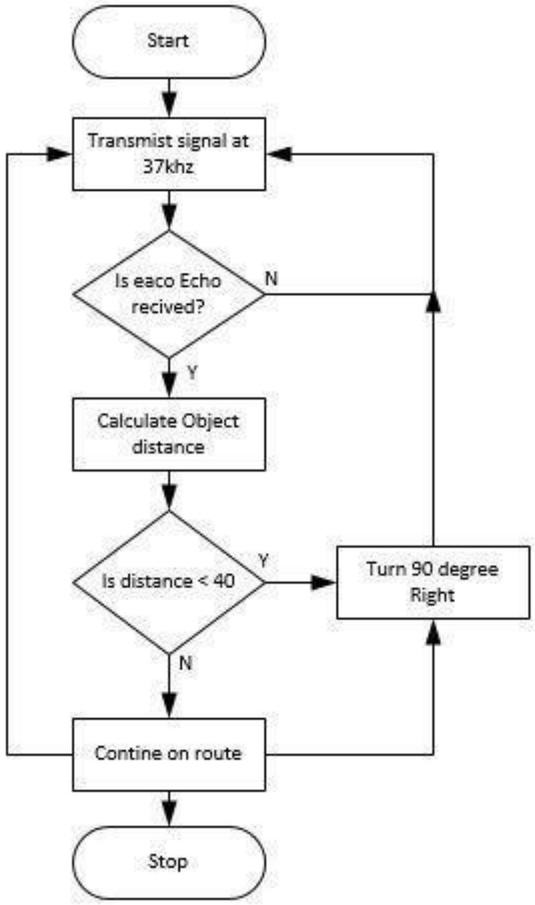


Figure 3 Behaviour Diagram

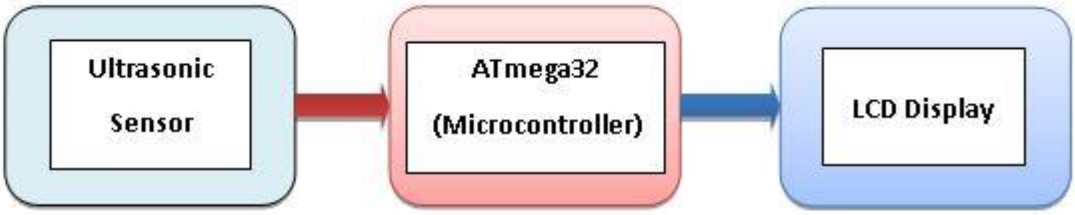
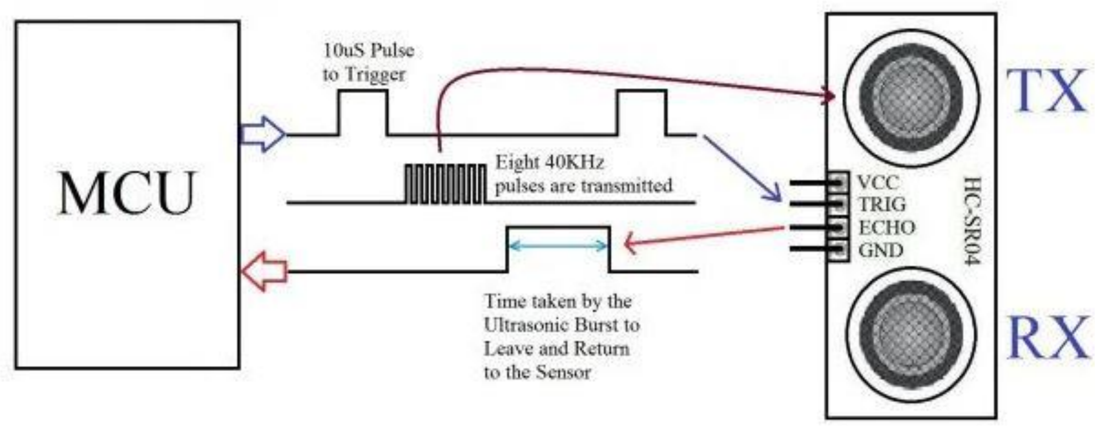


Figure 4 Block Diagram



Working of HC-SR04 Ultrasonic Sensor

Figure 7 Structural Diagram

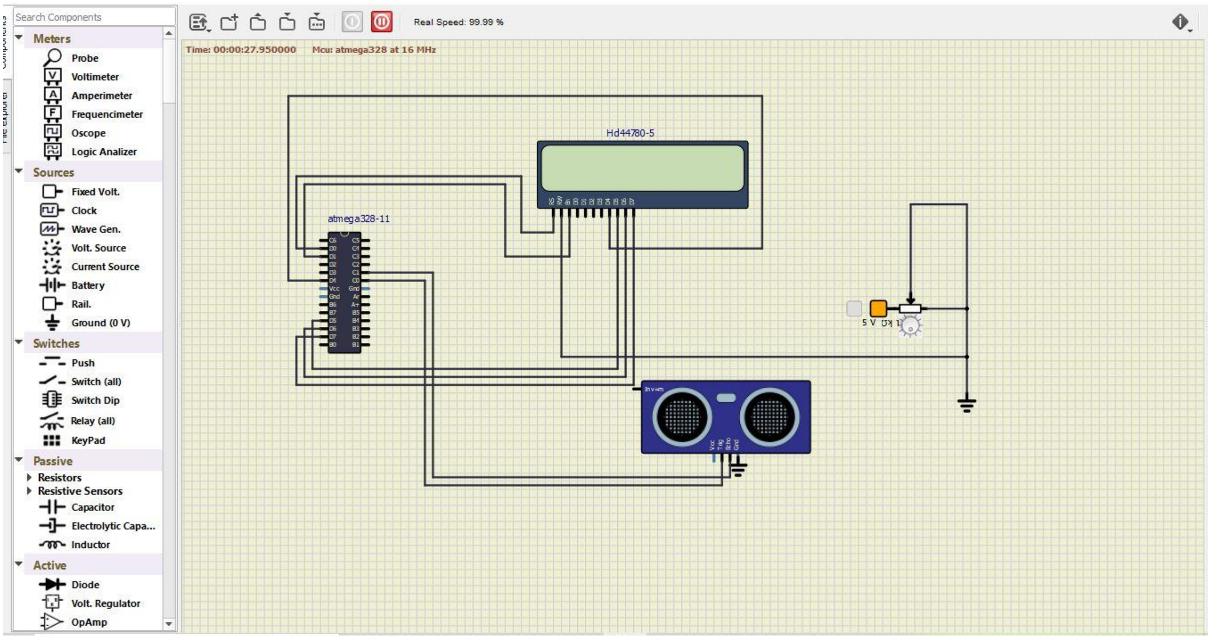


Figure 8 Simulation

**Test Plan**

**Obstacle Detection:**

**How: Our implementation for this step requires multiple steps:**

Step 1: Find a distance value between each pair of sensors. To test the distance

Value, we may use the numbers we see for the height and length, as well as the Pythagorean Theorem. ####Step 2: Check the angle found between each pair of sensors using the distance value initially found. ####Step 3: Using these values, determine what each angle should approximately be to detect different types of obstacles. ####Step 4: Detect the obstacles.

**Output: As we had steps for each test, we will again make steps for the expected outputs:**

Step 1: Compare the outputted (through serial) value for the hypotenuse to the

Pythagorean calculated value. We expect them to be the same.

Step 2: Using the same technique as step 1 except calculating the angle, we should

See the same value for this calculation as well.

Step 3: The values and outputs for the “obstacle detected” will be constantly

Checked and rechecked to make sure the angles determine the correct obstacle.

Step4: Adding Audio to the Ultrasonic Sensors.

**Testing cases**

| **Average Speed(m/s)** | **0.8** | **1.5** | **2.0** |
|---|---|---|---|
| Mean RMS error (cm)* | 19.4 | 12.7 | 10.2 |
| SD** | 11.2 | 14.3 | 13.4 |
| Sensing error (%) | 5.0 | 1.6 | 1.0 |

| | | | |
|---|---|---|---|
| **Average Speed(m/s)** | **0.8** | **1.5** | **2.0** |

------------------------------------------------------

RMS error: Root mean square error between actual and sensing distance.

SD**: Standard deviation of the RMS errors.

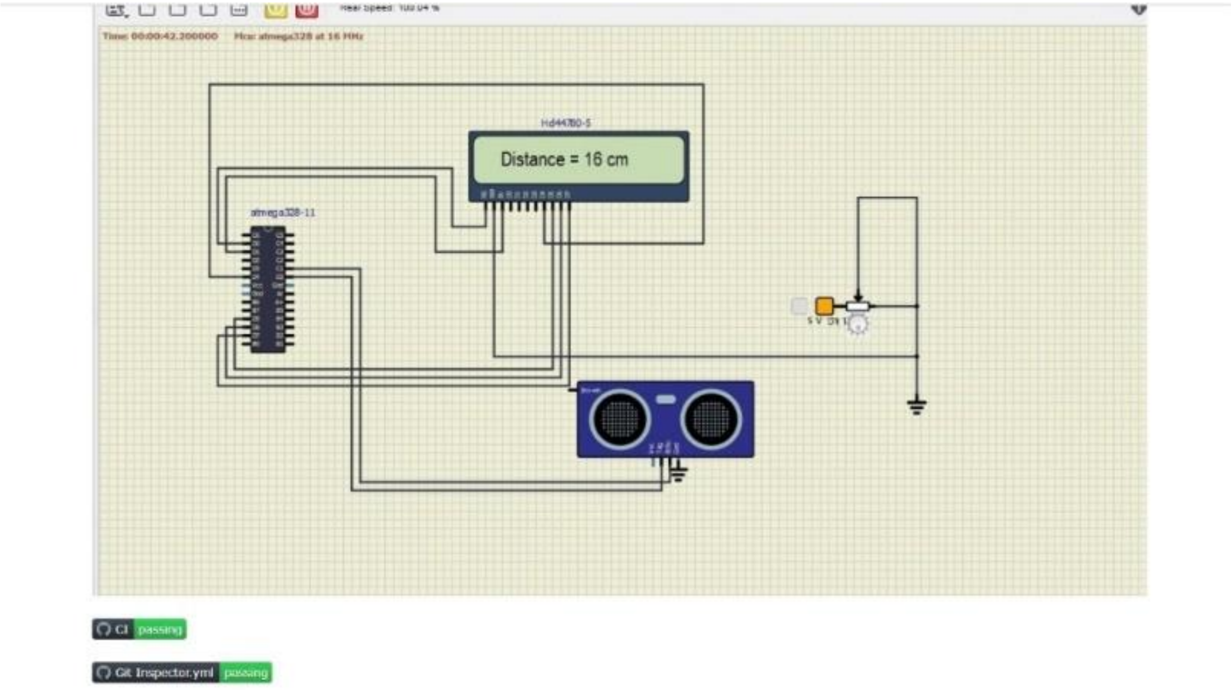
## Summary

The objective of the project was to design and implement an ultrasonic distance meter. The device described here can detect the target and calculate the distance of the target. The ultrasonic distance meter is a low cost, low a simple device for distance measurement. The device calculates the distance with suitable accuracy and resolution. It is a handy system for non-contact measurement of distance. The device has its application in many fields. It can be used in car backing system, automation and robotics, detecting the depth of the snow, water level of the tank, production line. This device will also have its application in civil and mechanical field for precise and small measurements. For calculating the distance using this device, the target whose distance is to be measured should always be perpendicular to the plane of propagation of the ultrasonic waves. Hence the orientation of the target is a limitation of this system. The ultrasonic detection range also depends on the size and position of the target. The bigger is the target, stronger will be the reflected signal and more accurate will be the distance calculated. Hence the ultrasonic distance meter is an extremely useful device.

## Git Link:

Link: milixx21/M2-Embedded_ultrasonic-Sound-Sensor (github.com)

### Git Dashboard :



# Miniproject 3 – Diary Entry System [Team]

## Modules

1. SDLC
2. Git

## Requirements

## 4W's and 1 H's

### Wh0 :

Anybody can use it.

### Where:

It can be used by Travel Specialists or doctors infact anybody can utilize it to keep their records safe.

### What:

Makes a difference the client to effectively include their imperative meetings,presentation records, additionally can be altered.

## When:

At whatever point the client needs to keep his individual records secure at a place.

## How:

It will keep your all individual records securely at one place.

## High Level Requirements

| ID | Description | Status |
|---|---|---|
| HLR_1 | Add the inputs to add records,edit,view and delete. | Implemented |
| HLR_2 | Users can add the record in the system. | Implemented |
| HLR_3 | Can view that record for further. | Implemented |
| HLR_4 | User can Edit the added record and can make some changes in it | Implemented |
| HLR_5 | Can delete the record permanently if not needed. | Implemented |
| HLR_6 | User can edit the Password for security purpose. | Implemented |

## Low Level Requirements

| ID | Description | Status |
|---|---|---|
| LLR_1 | Login page of Diary Entry system. | Implemented |
| LLR_2 | The system will ask password to view and Edit the records. | Implemented |
| LLR_3 | Edit data | Implemented |
| LLR_4 | Enter username and password | Implemented |
| LLR_5 | Newly addd details should be recorded | Implemented |

## Design

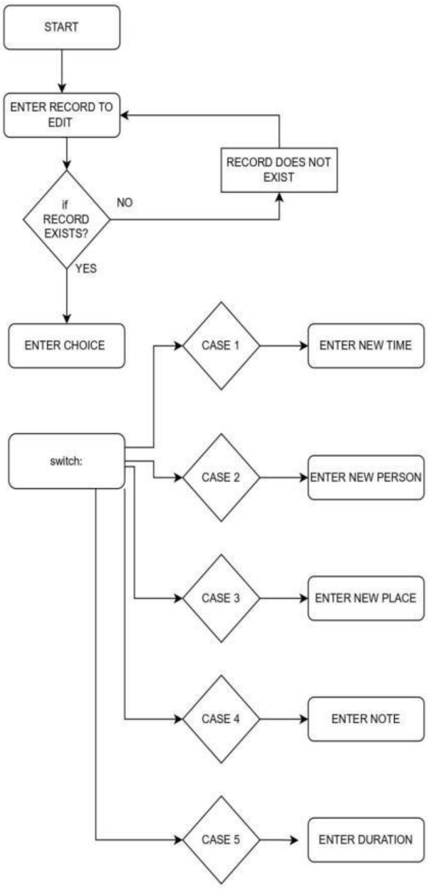
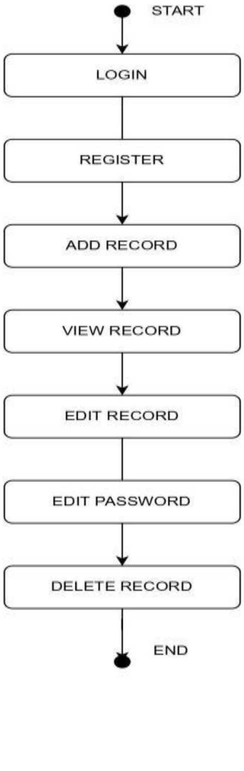
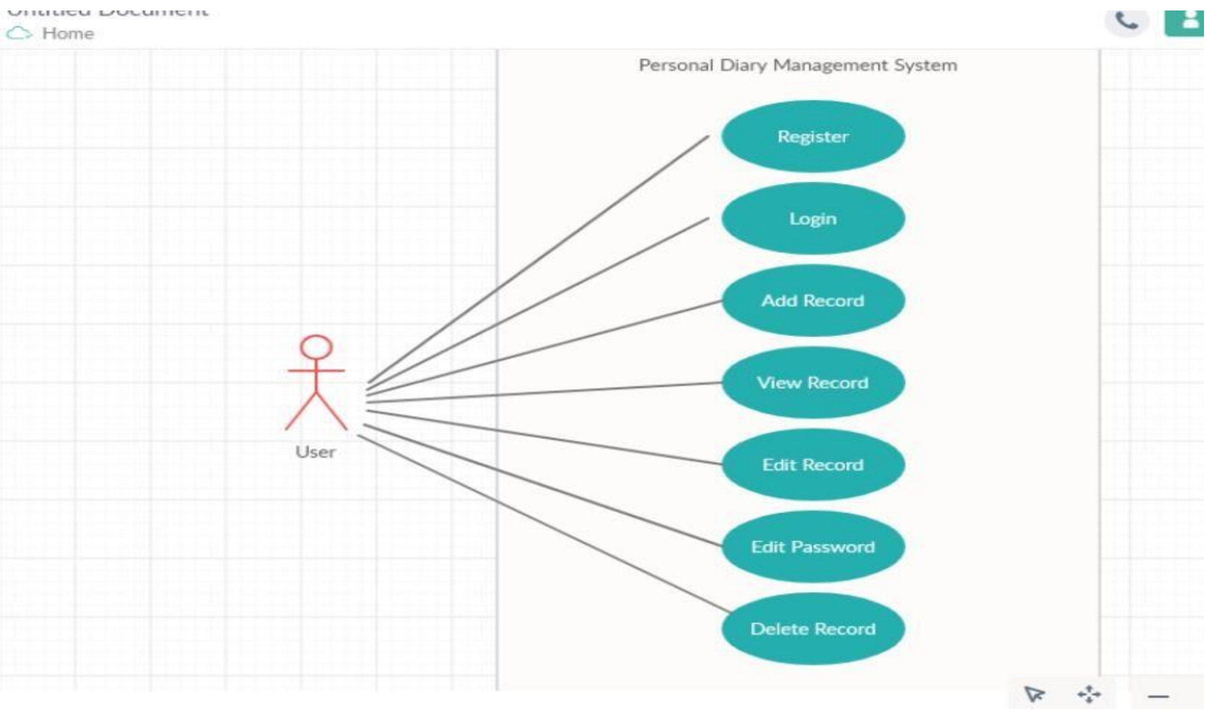


Figure 10 :Behauvior diagram

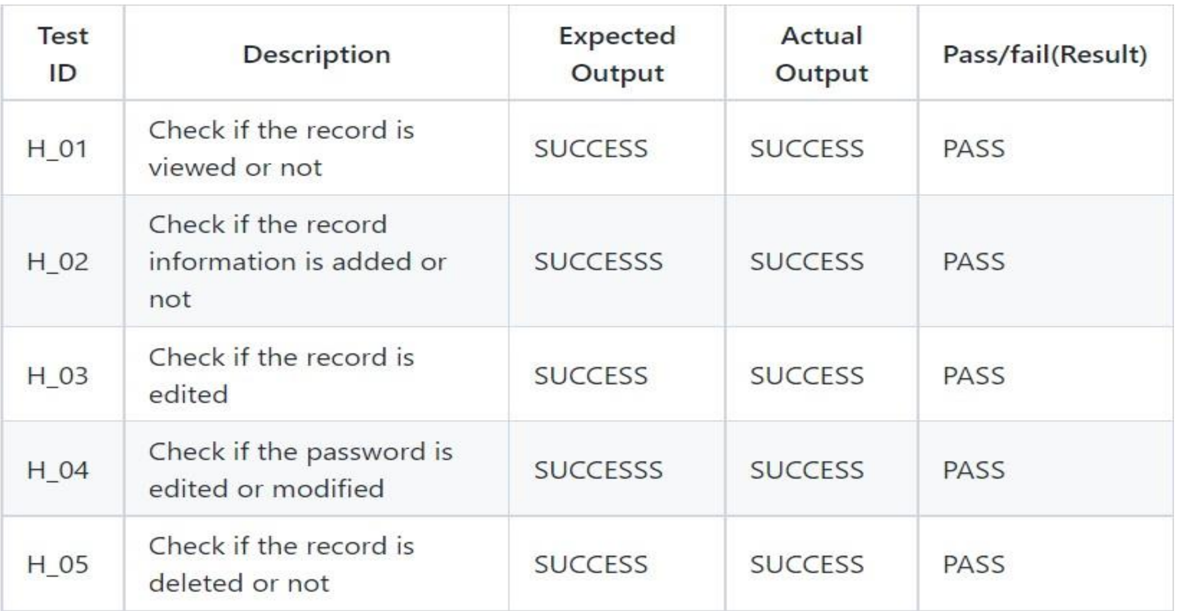




Structure Diagram

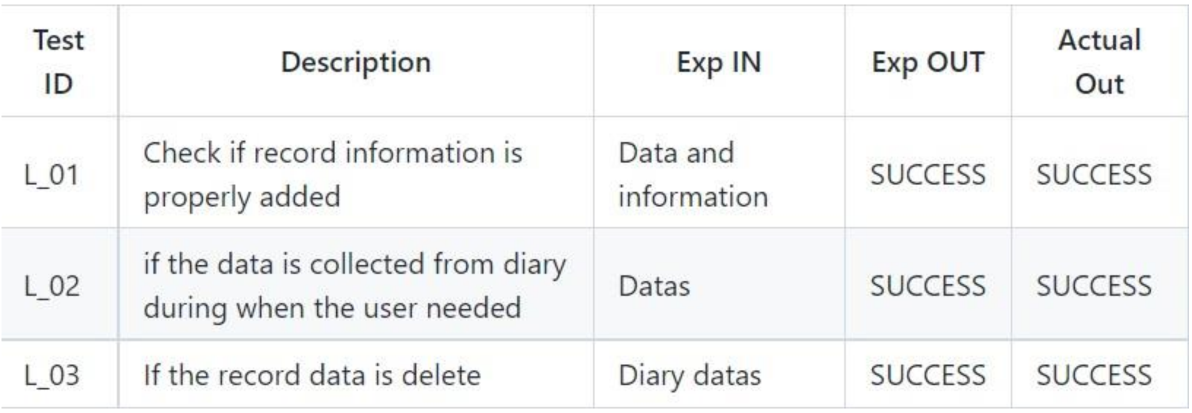
# Test Plan

## High Level Test Plan



| Test ID | Description | Expected Output | Actual Output | Pass/fail(Result) |
|---|---|---|---|---|
| H_01 | Check if the record is viewed or not | SUCCESS | SUCCESS | PASS |
| H_02 | Check if the record information is added or not | SUCCESSS | SUCCESS | PASS |
| H_03 | Check if the record is edited | SUCCESS | SUCCESS | PASS |
| H_04 | Check if the password is edited or modified | SUCCESSS | SUCCESS | PASS |
| H_05 | Check if the record is deleted or not | SUCCESS | SUCCESS | PASS |

## Low Level Test Plan



| Test ID | Description | Exp IN | Exp OUT | Actual Out |
|---|---|---|---|---|
| L_01 | Check if record information is properly added | Data and information | SUCCESS | SUCCESS |
| L_02 | if the data is collected from diary during when the user needed | Datas | SUCCESS | SUCCESS |
| L_03 | If the record data is delete | Diary datas | SUCCESS | SUCCESS |

## Implementation and SummaryGit

## Link:

Link : milixx21/M1_App_Personal-Diary-Management- (github.com)

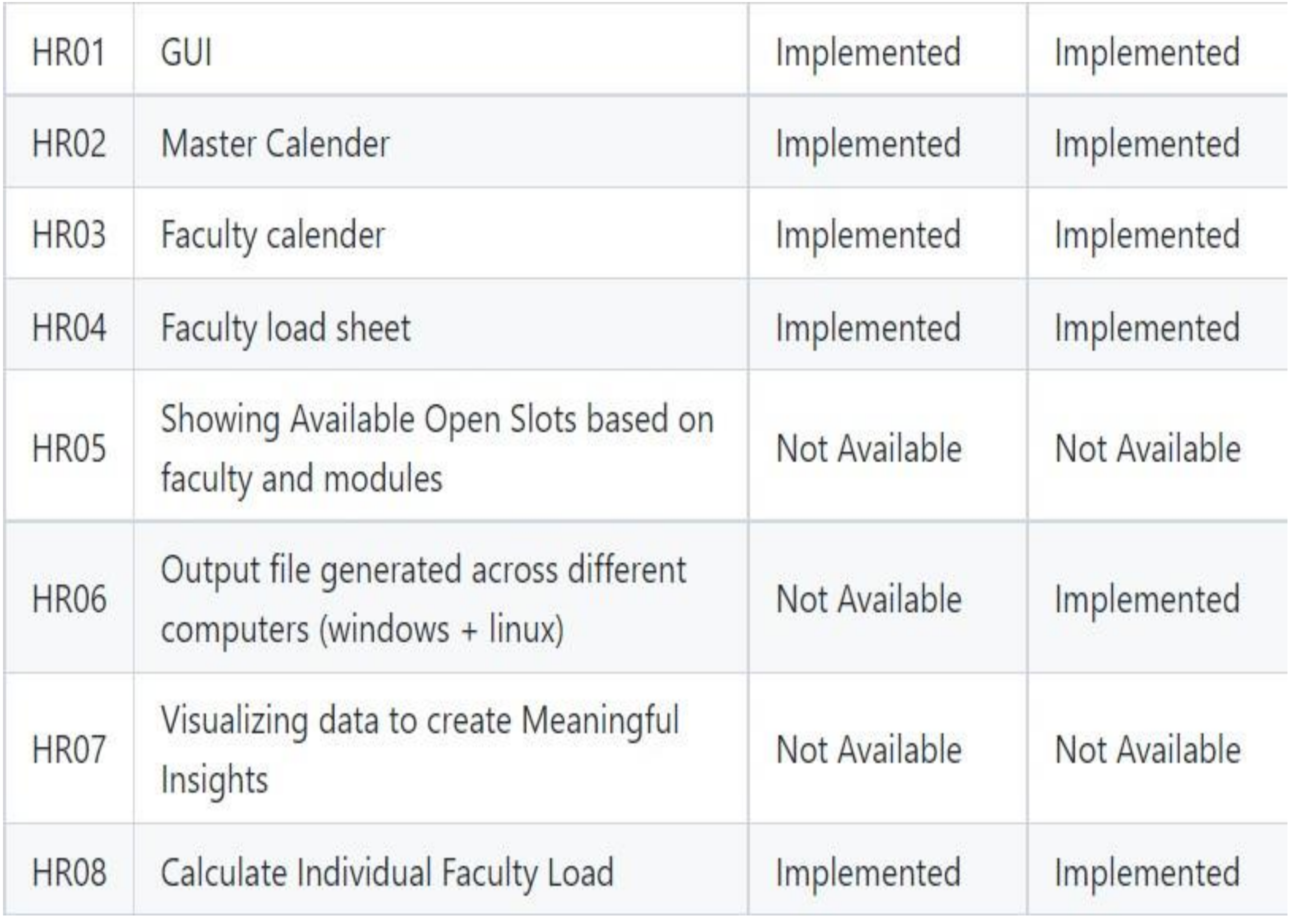
# Miniproject 4 – Calendar Automation[Team]

## Modules

1. Python
2. Git

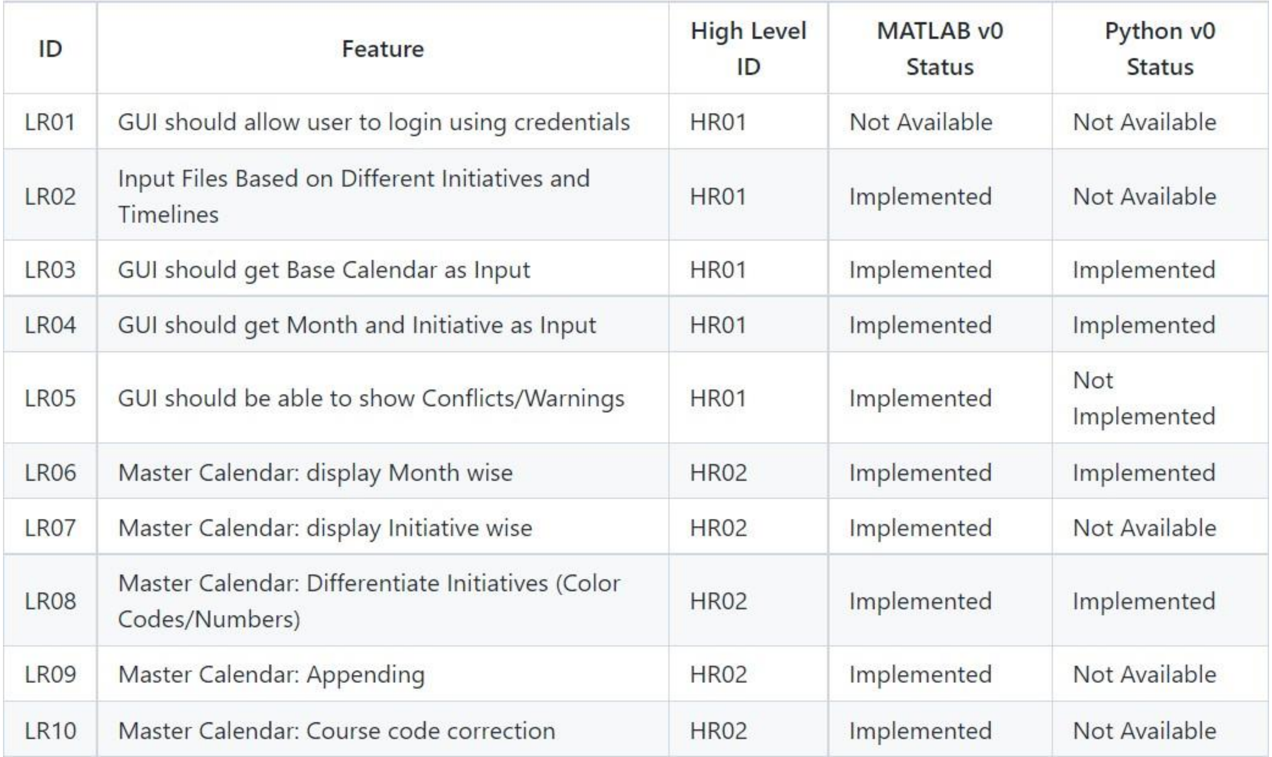
## Requirements

### High Level Requirements



| | | | |
|---|---|---|---|
| HR01 | GUI | Implemented | Implemented |
| HR02 | Master Calender | Implemented | Implemented |
| HR03 | Faculty calender | Implemented | Implemented |
| HR04 | Faculty load sheet | Implemented | Implemented |
| HR05 | Showing Available Open Slots based on faculty and modules | Not Available | Not Available |
| HR06 | Output file generated across different computers (windows + linux) | Not Available | Implemented |
| HR07 | Visualizing data to create Meaningful Insights | Not Available | Not Available |
| HR08 | Calculate Individual Faculty Load | Implemented | Implemented |

# Low Level Requirements



| ID | Feature | High Level ID | MATLAB v0 Status | Python v0 Status |
|---|---|---|---|---|
| LR01 | GUI should allow user to login using credentials | HR01 | Not Available | Not Available |
| LR02 | Input Files Based on Different Initiatives and Timelines | HR01 | Implemented | Not Available |
| LR03 | GUI should get Base Calendar as Input | HR01 | Implemented | Implemented |
| LR04 | GUI should get Month and Initiative as Input | HR01 | Implemented | Implemented |
| LR05 | GUI should be able to show Conflicts/Warnings | HR01 | Implemented | Not Implemented |
| LR06 | Master Calendar: display Month wise | HR02 | Implemented | Implemented |
| LR07 | Master Calendar: display Initiative wise | HR02 | Implemented | Not Available |
| LR08 | Master Calendar: Differentiate Initiatives (Color Codes/Numbers) | HR02 | Implemented | Implemented |
| LR09 | Master Calendar: Appending | HR02 | Implemented | Not Available |
| LR10 | Master Calendar: Course code correction | HR02 | Implemented | Not Available |

## Implementation and Summary

## Git Link:

Link: tlnsnani/OopsWithPython_Calendar_Automation_Team-48 (github.com)

# 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 calendar Automation
2. Integrator: Integrated all the codes

**Mini project 5 –Team BMW [Team]**

**Module: - Applied Model Based Design Module**

**Requirements**

## INTRODUCTION

Anti-lock brake systems (ABS) prevent brakes from locking during braking. Under normal braking conditions the driver controls the brakes. However, during severe braking or on slippery roadways, when the driver causes the wheels to approach lockup, the antilock system takes over. ABS modulates the brake line pressure independent of the pedal force, to bring the wheel speed back to the slip level range that is necessary for optimal braking performance. An antilock system consists of wheel speed sensors, a hydraulic modulator, and an electronic control unit. The ABS has a feedback control system that modulates the brake pressure in response to wheel deceleration and wheel angular velocity to prevent the controlled wheel from locking. The system shuts down when the vehicle speed is below a pre-set threshold.

## OVERVIEW

- ABS was first invented and applied in the aircraft industry and then was introduced to automobile industry in the early 1970's. However, it had not been used popularly until the middle of the 1980's due to technical difficulties and high cost.
- ABS functions in place of the traditional brake system at times of wheel lock-up. A quick test sequence checks all the components of the system.
- If ever the test sequence fails, the normal brake system is in control.
- Although the normal brake system can give instant and efficient braking, it can cause the wheels to be lock up, therefore, the driver cannot steer and would lose control of the car.
- If any of the wheels happen to be skidding, the driver must recognize wheel-skid and manually 'pump the brakes' to avoid a skid. The advantage of ABS lies in its ability to allow the driver retain steering control in order to keep the car moving in the direction that the wheels are turned towards, rather than skidding in the direction of the car's forward momentum.
- ABS has the classic design of an embedded system.
    1. controller Sensor
    2. Wheel speed.
    3. Actuators (valve and ABS reservoir) at each wheel.

## REQUIREMENTS

### High Level Requirements:-

| ID | Description |
|---|---|
| **HLR1** | Receive on/off signals from the brakes, and use them for engaging and disengaging the ABS system |
| **HLR2** | Receive rotational speed data from four wheel speed sensors |
| **HLR3** | The same signal that is used to turn on the brake lights is read by the ABS system in order to determine whether or not the brake has been pressed engaged. The ABS will then only be able to become engaged if this signal shows the brakes are being used. |
| **HLR5** | The ABS will receive information from a wheel speed sensor for each wheel. Each wheel speed sensor will send the speed of the wheel it is monitoring in meters/second. |
| **HLR6** | Run a system diagnostic test sequence at ignition and determine if any errors are present in the system. |

### Low Level Requirements:-

| ID | Description |
|---|---|
| **LLR1** | The wheel rotates with an initial angular speed that corresponds to the vehicle speed before the brakes are applied. |
| **LLR2** | The system test will engage when the car is turned on. |
| **LLR3** | Calculate rotational deceleration from the wheel speed data, and determine if wheel lock-up is imminent. |
| **LLR4** | The same signal that is used to turn on the brake lights is read by the ABS system in order to determine whether or not the brake has been pressed engaged. The ABS will then only be able to become engaged if this signal shows the brakes are being used. |
| **LLR5** | Terminate system execution if any failure occurs form either test. The termination shall not affect normal braking behaviour of the vehicle |

## Analysis and Physics

The wheel rotates with an initial angular speed that corresponds to the vehicle speed before the brakes are applied. We used separate integrators to compute wheel angular speed and vehicle speed. We use two speeds to calculate slip, which is determined by Equation 1. Note that we introduce vehicle speed expressed as an angular velocity (see below).

$$\omega_v = \frac{V}{R} \text{ (equals the wheel angular speed if there is no slip)}$$

## Equation 1

$$\omega_v = \frac{V_v}{R_r}$$

$$slip = 1 - \frac{\omega_w}{\omega_v}$$

$\omega_v =$ vehicle speed divided by wheel radius

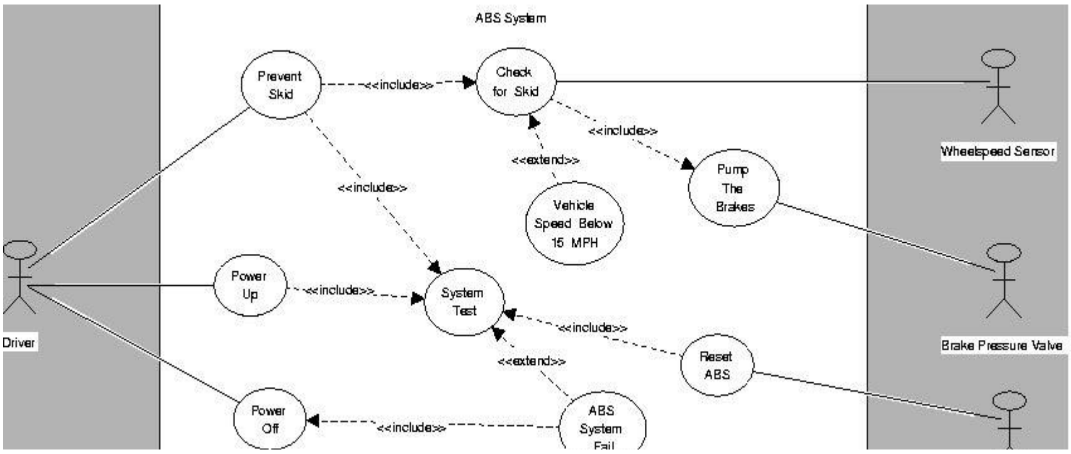
$V_v =$ vehicle linear velocity

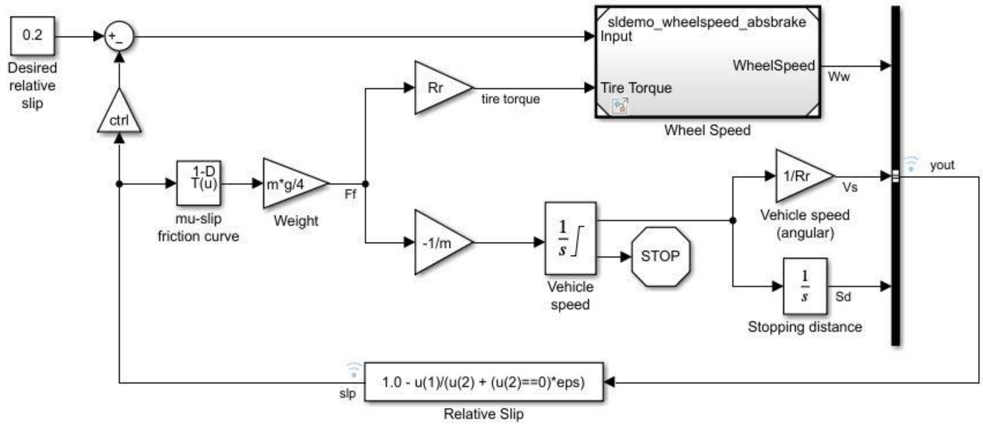
$R_r =$ wheel radius

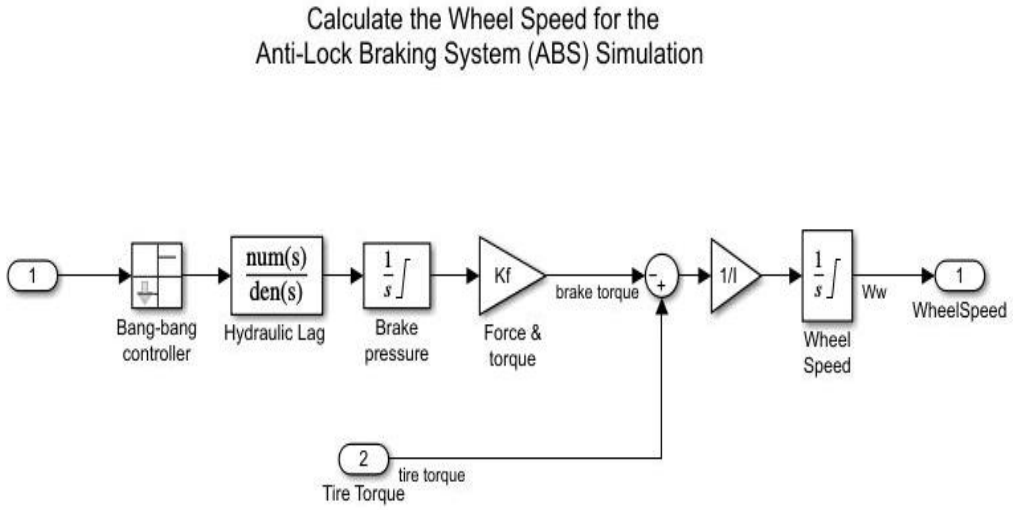
$\omega_w =$ wheel angular velocity

From these expressions, we see that slip is zero when wheel speed and vehicle speed are equal, and slip equals one when the wheel is locked. A desirable slip value is `0.2`, which means that the number of wheel revolutions equals `0.8` times the number of revolutions under non-braking conditions with the same vehicle velocity. This maximizes the adhesion between the tire and road and minimizes the stopping distance with the available friction.

## DESIGN



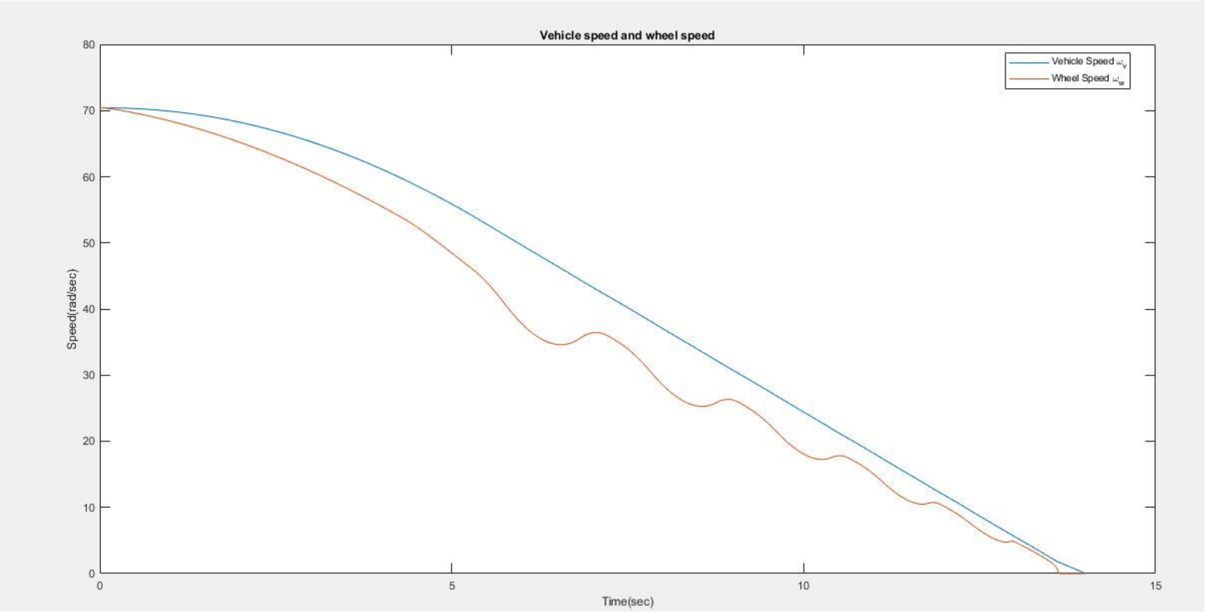




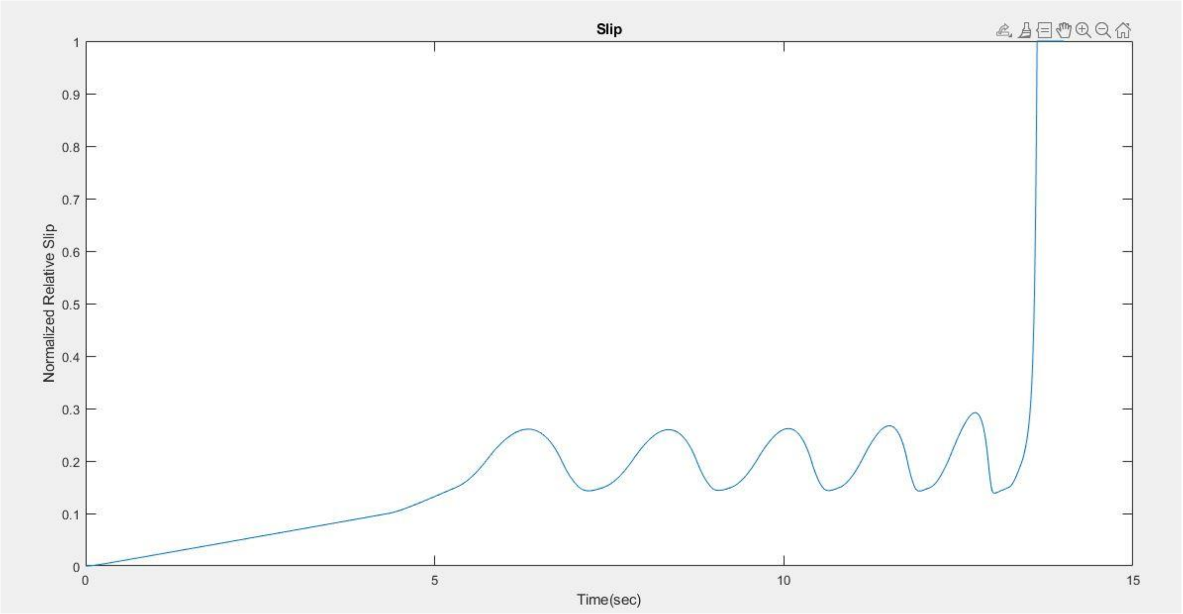
## OUTPUT

## Running the Simulation in ABS Mode

- <u>**Vehicle Speed and Wheel Speed**</u>



- <u>**Slip**</u>

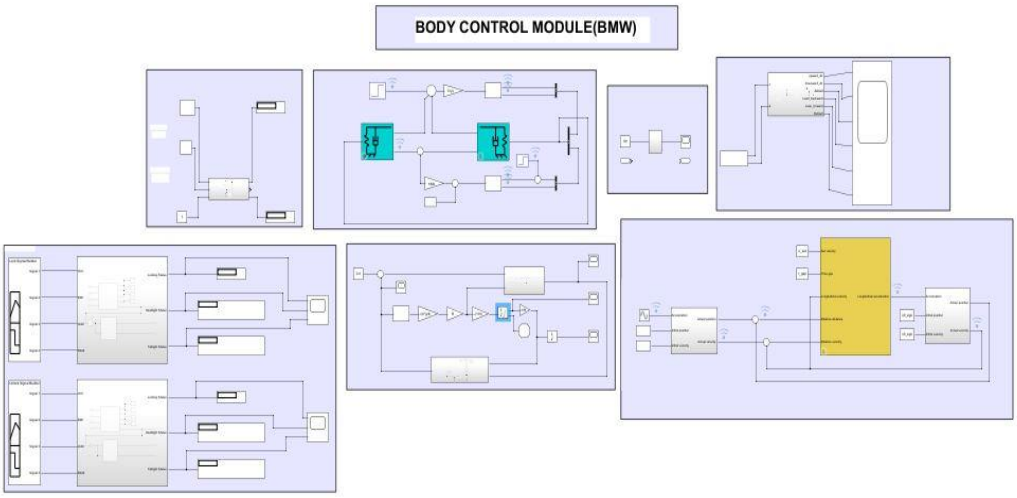


## Conclusion

This model shows how you can use Simulink to simulate a braking system under the action of an ABS controller. The controller in this example is idealized, but you can use any proposed control algorithm in its place to evaluate the system's performance. You can also use the Simulink® Coder™ with Simulink as a valuable tool for rapid prototyping of the proposed algorithm. C code is generated and compiled for the controller hardware to test the concept in a vehicle. This significantly reduces the time needed to prove new ideas by enabling actual testing early in the development cycle.

For a hardware-in-the-loop braking system simulation, you can remove the 'bang-bang' controller and run the equations of motion on real-time hardware to emulate the wheel and vehicle dynamics. You can do this by generating real-time C code for this model using the Simulink Coder. You can then test an actual ABS controller by interfacing it to the real-time hardware, which runs the generated code. In this scenario, the real-time model would send the wheel speed to the controller, and the controller would send brake action to the model.

# Merging with BMW (BCM MODULE)



# Miniproject 6 – AUTOMATIC RAIN OPERATED WIPER [Team]

**Modules**
**Mastering Microcontrollers with Embedded Driver Development Module**
**Requirements**

## Requirements

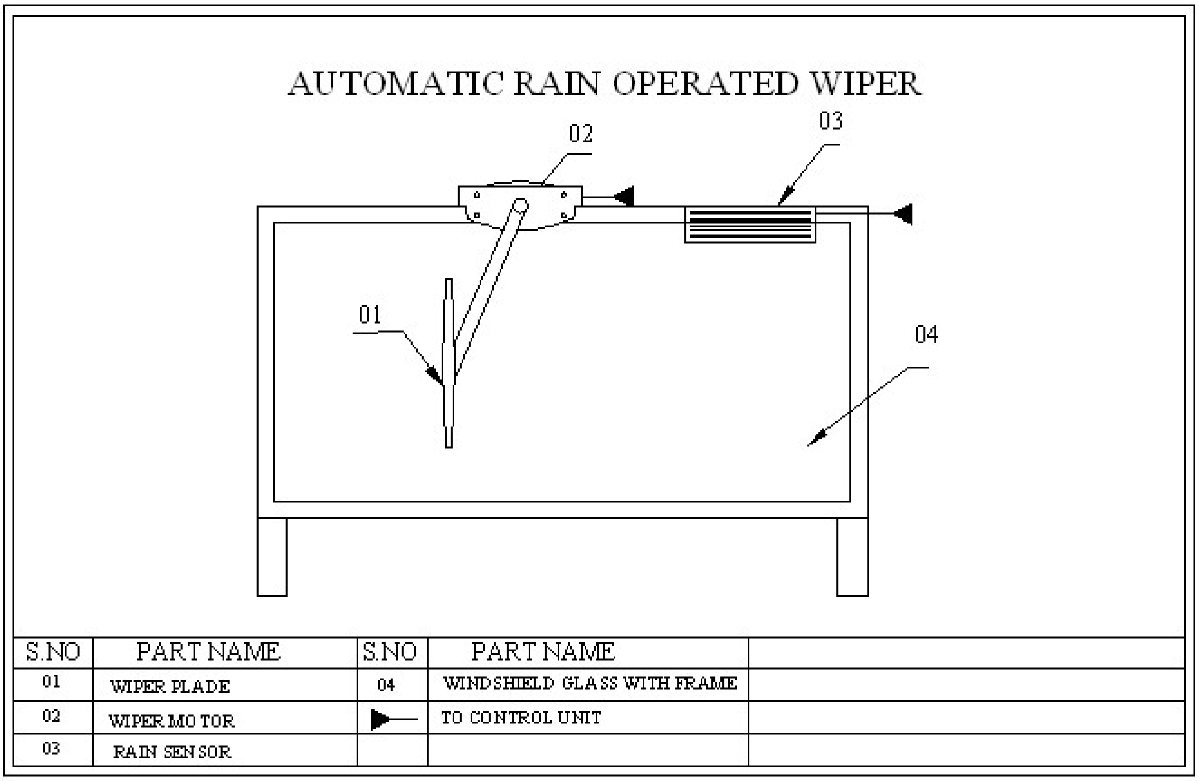
### Introduction

Automotive wipers form an essential part for any vehicle. They perform to remove water, ice, snow, and dust from a windshield of a vehicle. An automotive wiper is either powered by an electric motor or pneumatic power. Almost all motor vehicle including cars, trucks, buses, train locomotives and watercraft with a cabin are equipped with one or more such wipers. The automotive wiper market is multiplying as there is an exponentially increased production of automobiles globally.

### Features

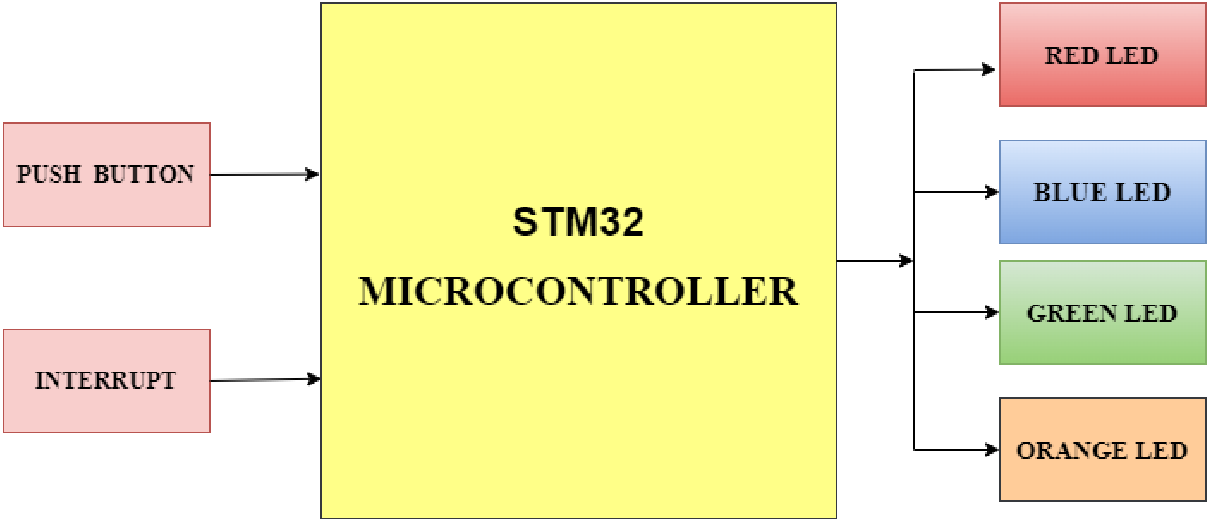
- To achieve high safety
- To reduce man power
- To increase the efficiency of the vehicle
- To reduce the work load
- To reduce the vehicle accident
- To reduce the fatigue of workers
- To high responsibility
- Less Maintenance cost

## State of Art



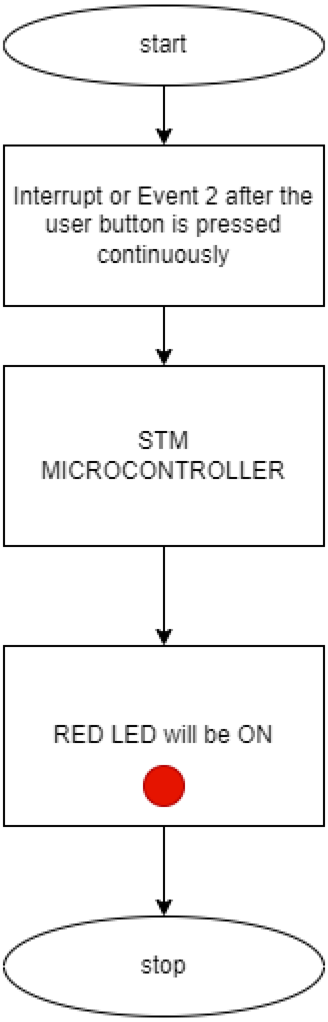
| S.NO | PART NAME | S.NO | PART NAME | |
|---|---|---|---|---|
| 01 | WIPER PLADE | 04 | WINDSHIELD GLASS WITH FRAME | |
| 02 | WIPER MOTOR | ▶— | TO CONTROL UNIT | |
| 03 | RAIN SENSOR | | | |

## Design



Block Diagram

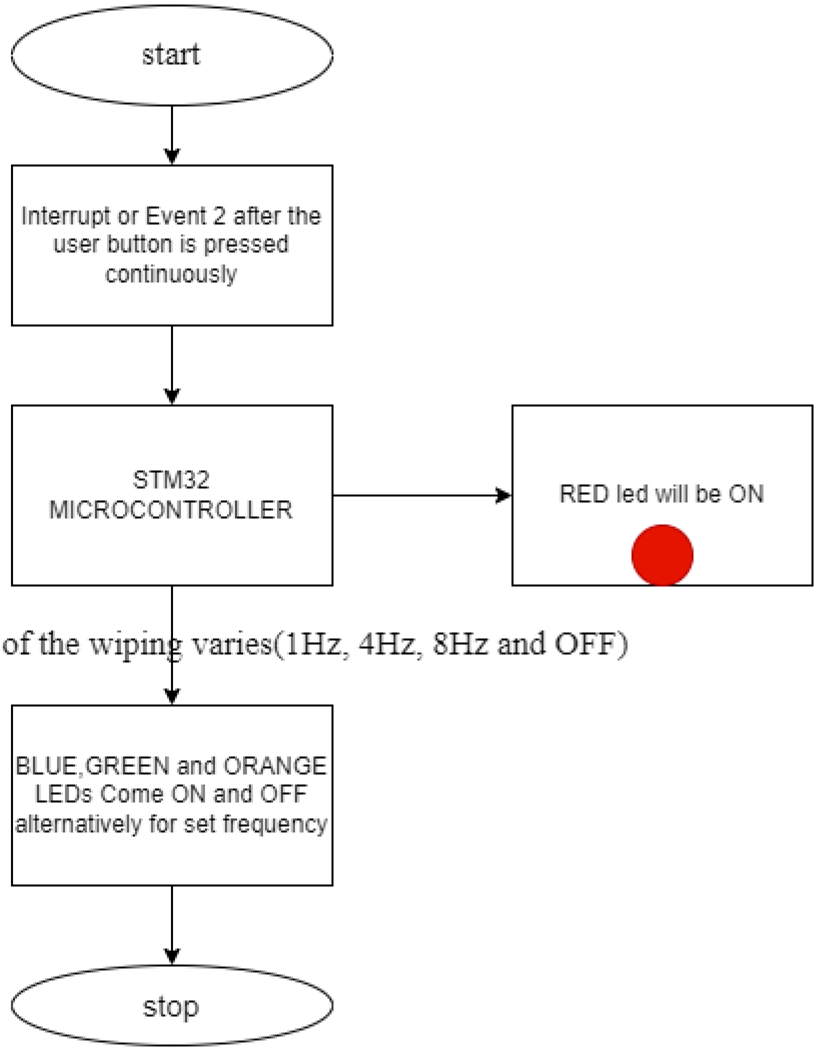
## STATE A


start
Interrupt or Event 2 after the user button is pressed continuously
STM MICROCONTROLLER
RED LED will be ON
stop

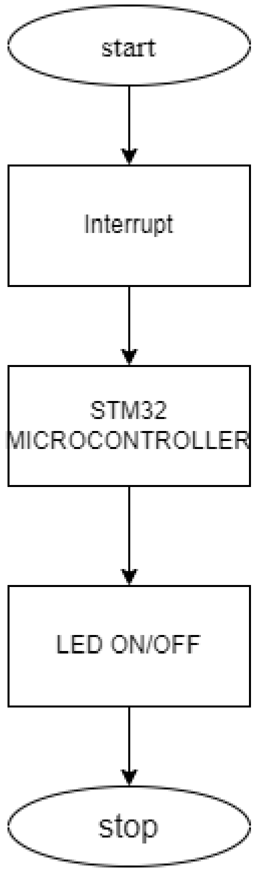
## WIPER IS ON

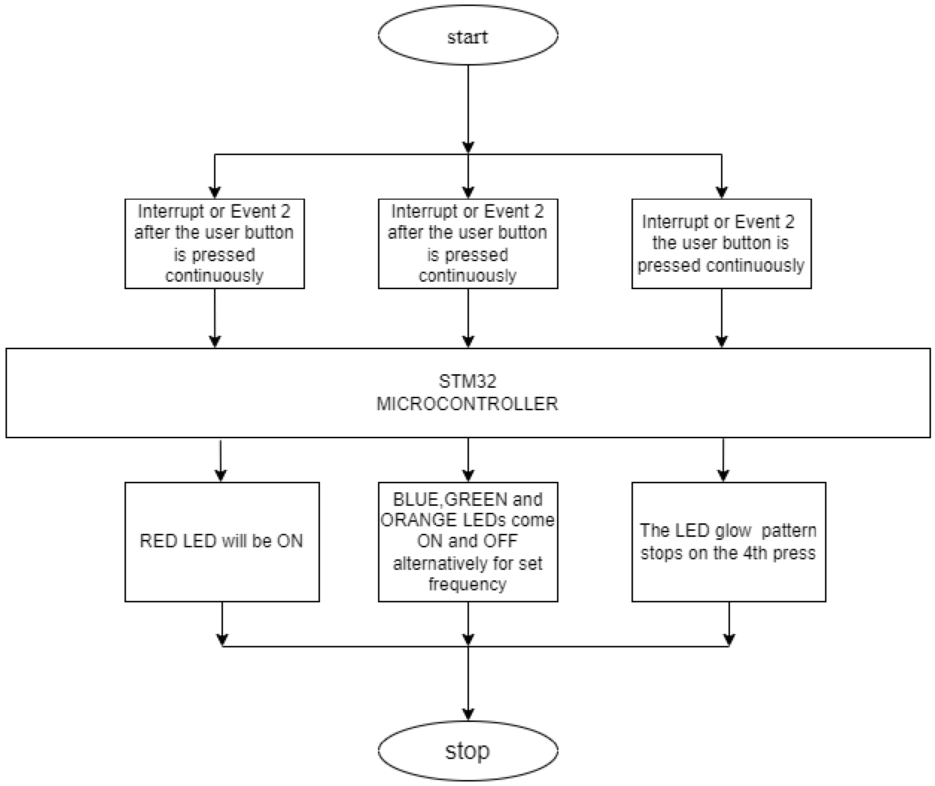
Diagram

## STATE B


start
Interrupt or Event 2 after the user button is pressed continuously
STM32 MICROCONTROLLER
RED led will be ON
BLUE,GREEN and ORANGE LEDs Come ON and OFF alternatively for set frequency
stop

The frequency of the wiping varies(1Hz, 4Hz, 8Hz and OFF)


start
Interrupt
STM32 MICROCONTROLLER
LED ON/OFF
stop


start
Interrupt or Event 2 after the user button is pressed continuously
Interrupt or Event 2 after the user button is pressed continuously
Interrupt or Event 2 the user button is pressed continuously
STM32 MICROCONTROLLER
RED LED will be ON
BLUE,GREEN and ORANGE LEDs come ON and OFF alternatively for set frequency
The LED glow pattern stops on the 4th press
stop

## Test Plan

## High Level Requirements

| ID | Description | Category | Status |
|---|---|---|---|
| HR01 | User shall be able to switch to other control condition | Techincal | - |
| HR02 | User shall be able to switch to other control when button is pressed | Techincal | - |
| HR03 | User shall be able to switch from intial state when is pressed for 2 secs | Techincal | - |

### Low Level Requirements

| ID | Description | Category | Status |
|---|---|---|---|
| LR01 | User shall be able to switch back to intial state when is pressed for 2 secs | Techincal | - |
| LR02 | User shall not be able to switch back to previous Condition when button is pressed | Techincal | - |

## Implementation and Summary

## Git Link:

Link: GENESIS-2022/MasteringMCU-Team30: Details (github.com)

**Mini project 7 – TATA [INDIVISUAL]**

**Modules: - Automotive Systems**
**Requirements**

## Ford Aspire

The Ford Aspire nameplate has been used by the American automobile manufacturer Ford for the following cars, in the following markets: Ford Festiva, in North America from 1993 to 1997. The sedan version of the Ford Figo, a rebadged third generation Ford Ka in India since 2015.

## Body Control Module

**Features:**

- Door Lock System
- Interior Light Control
- Power Mirror
- Power Window

**Individual Feature**: - **Interior Light Control**

## Introduction

The interior lighting in cars consisted of a few incandescent lights that turned on or off in response to microswitches in various doors or simple switches near the light fixture.

## 4W's 1H

### WHAT:

The lighting normally stays lit until the vehicle is turned on so the passengers can safely fasten their seat belts. In addition, interior lights can aid in reading maps or finding lost items in the dark and The following are a few things you should know about your car’s interior lights :( 1) Dim Light (2) Flickering Light (3)Light stays on , Etc.

### WHERE:

The interior light is used in dashboard for indication, in safety indicator and headlight of door and foot step, bonnet, boot etc.

### WHEN:

The system is standard or available on many of Ford's models, including the Fiesta, Focus, Fusion, Taurus, and Mustang cars and the Escape, Edge, Flex, and F-150 trucks, depending on the selected trim level.

### WHY:

Most vehicles have interior lights that are also called dome lights or courtesy lights. These can be located on the ceiling of the vehicle and illuminate when people enter or exit the car.

**How:-**

When the interior lights in a car are working correctly, they will usually come on when you open your door and then shut off some time after you close the door. This process relies on a switch in the door jamb that opens when you open the door and closes when you close the door.

**Requirement**

**High Level Requirement:-**

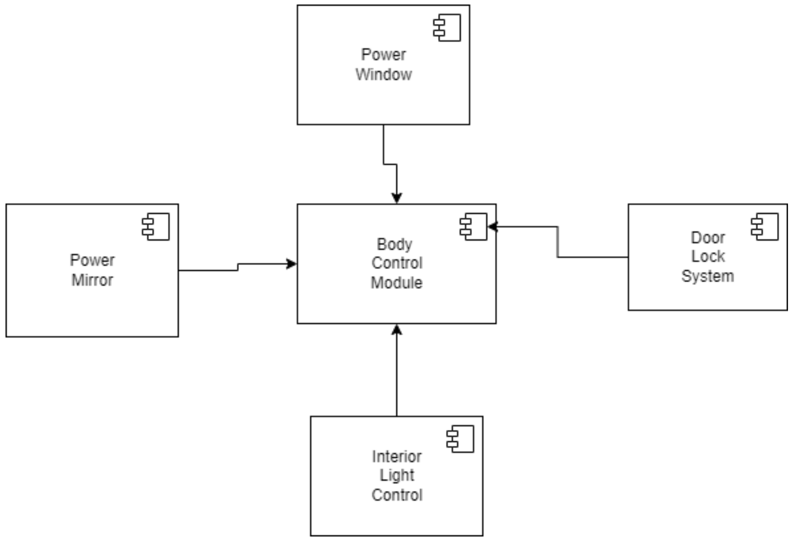
| High level Requirement | Description |
|---|---|
| HLR_1 | Light will be on when Door is open |
| HLR_2 | Light will be in on state until all the door is correctly closed |
| HLR_3 | Dashboard Light will ON when the car is unlocked |
| HLR_4 | Lights wiil be OFF after 10 sec when the lock button pressed on the key |
| HLR_5 | In Night Foot step light is automatically ON |
| HLR_6 | Lights can be turn off and on manually with the switch |

**Low Level Requirement:-**

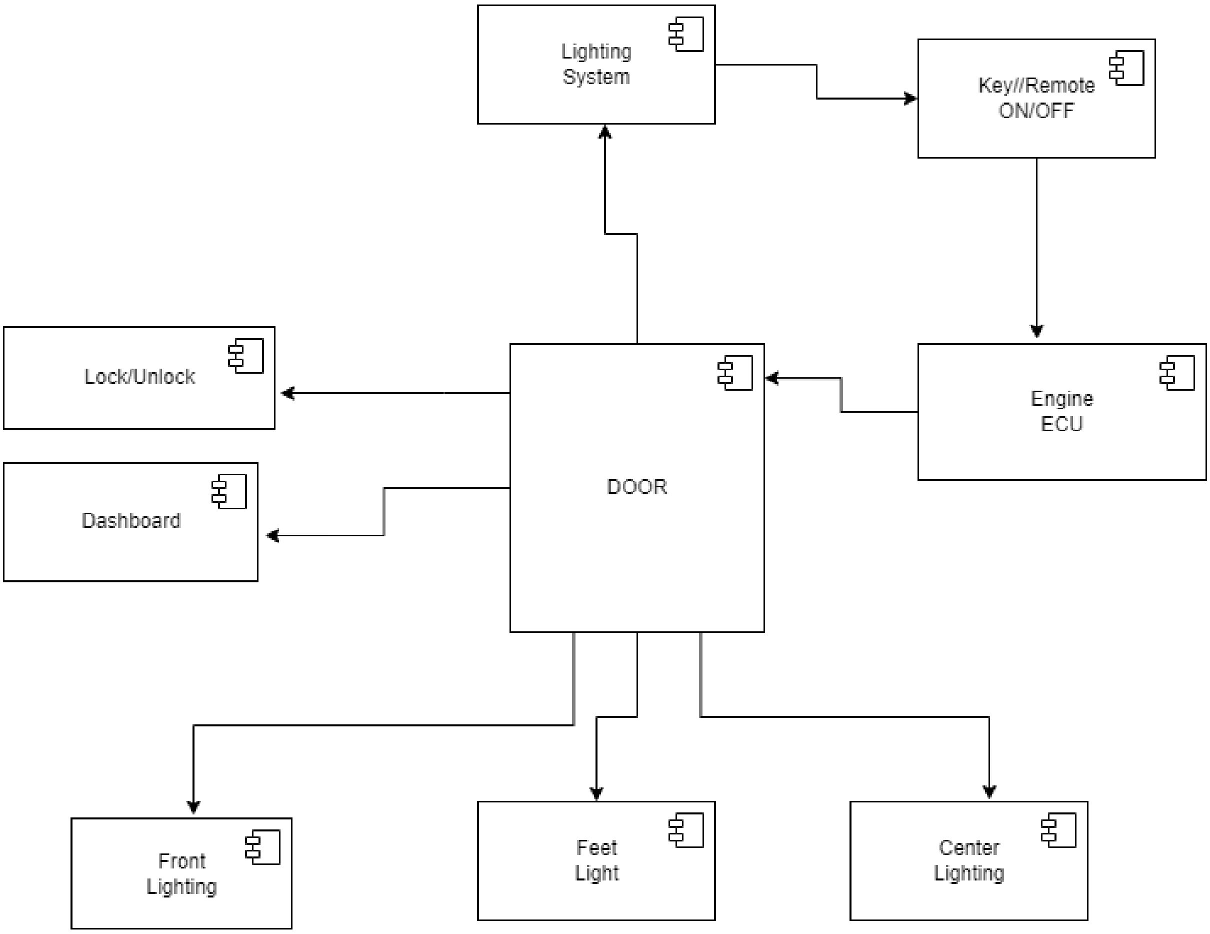
| Low level Requirement | Description |
|---|---|
| LLR_1 | Voice command to turn on and off the light |
| LLR_2 | Lights will be turned on when unlockes the car with the remote key from outside |
| LLR_3 | Add Multicolour LED |
| LLR_4 | The light will turn-off after 5 seconds when we lock the car through the key |
| LLR_5 | Safety Indicator |

## Design

### Body Control Module



## Interior Light Control



### Implementation and Summary

**Git Link:** milixx21/Automotive-system_Interior-light-control (github.com)

### Mini project 8 – EV Truck [Team]

### Module: - Applied Control Systems and Vehicle Dynamics

### EV TRUCK

An electric truck is an electric vehicle powered by batteries designed to transport cargo, carry specialized payloads, or perform other utilitarian work. Electric trucks have serviced niche applications like milk floats, pushback tugs and forklifts for over a hundred years, typically using lead-acid batteries, but the rapid development of lighter and more energy-dense battery chemistries in the twenty-first century has broadened the range of applicability of electric propulsion to trucks in many more roles. Electric trucks reduce noise and pollution, relative to internal-combustion trucks. Due to the high efficiency and low component-counts of electric power trains, no fuel burning while idle, and silent and efficient acceleration, the costs of owning and operating electric trucks are dramatically lower than their predecessors.

THE FUTURE OF HEAVY - DUTY VEHICLES = ELECTRIC TRUCKS

Electric trucks are becoming more popular due to its cost efficiency, better performance and lower emissions. Global sales of electric vehicles increased by 43% in 2020 and is expected to keep growing in upcoming years. The same will slowly apply on EV trucks and companies will adapt its business models accordingly.

## ARE ELECTRIC TRUCKS BETTER?

In urban areas, delivery routes with heavy traffic and lots of stops can become very costly. Electric trucks are roughly 50% more effective than diesel trucks, which makes them roughly 20% less expensive than diesel trucks. However, it largely depends on how the trucks will be used. Speeding, braking, exceeding RPM and many other aspects influence the consumption of the EV battery, which leads to a lower battery health and therefore, increased need for charging. Researchers analysed driving behaviour, number of stops, speeding and overall usage of the truck, and electric trucks clearly outperformed diesel trucks. By driving an EV truck, you use around 30% less total energy, reduce greenhouse gas emissions by roughly 50%, and lower down the noise level of the vehicle. The noise disturbance is a real issue, especially with older diesel trucks that produce a lot of noise and emissions. This challenge is typically solved with a right driver management system, which is a set of measurements to improve driving experience and manage fleet drivers safely and effectively.

## BENEFITS OF ELECTRIC TRUCKS

- Lower emissions
- Lower maintenance
- Lower noise disturbance
- Better performance
- Increased efficiency

It is always cheaper to charge your electric truck than spending money on gas. Electric trucks provide businesses with many benefits that primarily aim for the long run. EV trucks do not require fuel, which is already one of the biggest advantages, due to fuel cost and effect on nature. Driving electric trucks reduces CO2 emissions and actually offers better performance for drivers.EV trucks also have less parts, which should lead to less damage and lower maintenance. However, this depends on a truck model and its usage. While driving within urban areas with frequent stops and speeding, it is way more efficient to drive electric truck than diesel truck.

## Argument for Electric Trucks

WEIGHT - Commercial battery electric vehicle (CBEV) weight is not an issue

TECHNOLOGY - CBEV technology is proven and here now it will last beyond 10 years, Maintenance will be less costly

COST - it will be competitively priced, less expensive to operate and command a premium at resale

CHARGING - Trust the market to provide Commercial battery electric vehicle charging solutions, The grid and market will evolve with CBEVs

## Argument against Electric Truck's

WEIGHT - Vehicle tare weight is too high to support my freight needs

TECHNOLOGY - Technology is not ready, Maintenance may not be less costly and Vehicle life is too short

COST - Vehicle purchase price is too high for a positive ROI, Operating costs are too great for positive ROI and residual value is questionable

CHARGING - Charging infrastructure is not ready, Charging Infrastructure is not fast enough, The electric grid cannot support growth in electric vehicles

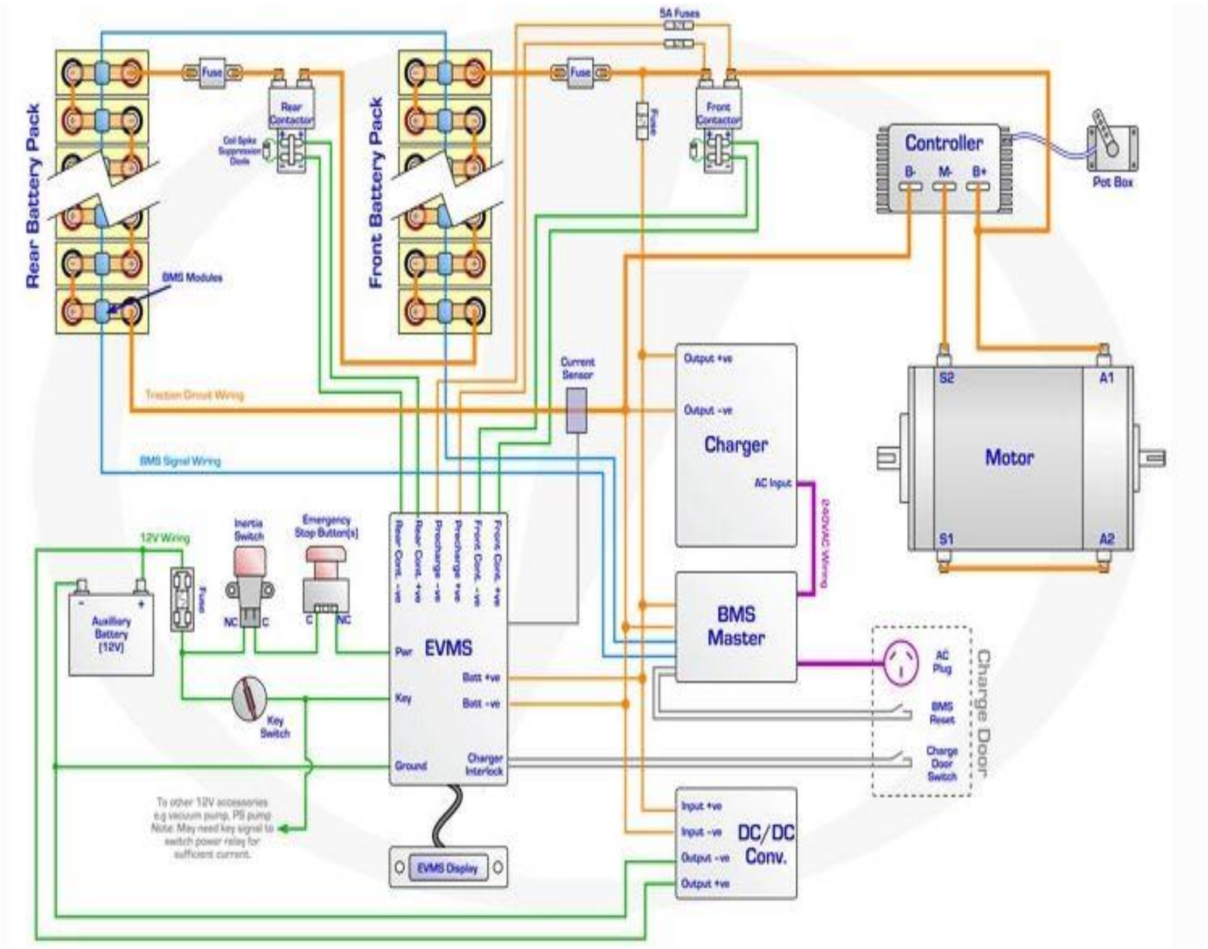
## Research

The Volvo FE Electric trucks join NFI's fleet of more than 4,500 heavy-duty tractors that support its dedicated transportation and port drayage services for customers spanning from manufacturing to retail. The pilot trucks will be based out of one of NFI's warehouse facilities in Southern California that serves as a central distribution centre for the region. "As the future of goods movement in the U.S. changes from more of a long-haul operation to regional and hub and spoke models, not only is that NFI's wheelhouse, it's an ideal scenario to immerse electrification into our regional hauling strategy," said Jim O'Leary, vice president, Assets/Fleet Services, NFI Industries. "Our executive team is excited to collaborate with the Volvo LIGHTS team to accelerate our transition to a zero-emission fleet, so that we can lower our carbon footprint, reduce our operating costs and provide a better work environment for our drivers."

**Reference:**

- Integrated, feed-forward hybrid electric vehicle simulation in SIMULINK and its use for power management studies
- Energy management strategy for a parallel hybrid electric truck,
- Energy management strategy for a parallel hybrid electric truck
- https://ieeexplore.ieee.org/abstract/document/7587102
- https://www.tandfonline.com/doi/abs/10.1080/00423110412331291553

**Simulation Design:**



**Analysis**

## TWO MODEL COMPARISON

| SPECIFICATIONS | Volvo Fe | Mack LR |
|---|---|---|
| • COST | 39,900$ | ₹ 15.29 Lakh - ₹ 16.82 Lakh |
| • GROSS COMBINATION WEIGHT | Up to 27 Tonnes | 7300 kg |
| • RANGE | UP to 200 km | More than 100 KMs |
| • BATTERY | Lithium-ion batteries | Lithium-ion batteries |
| • BATTERY CAPACITY | 200-265 kWh, 3,4 batteries | 62.5 kWh-Octillion Make-(10 S1P) |
| • CHARGING TIME(FULL CHARGE) | 11h with AC (22 kW) & 2h with DC (150 kW) | 2 hrs |
| • DRIVELINE/MOTOR | 2 electric motors, 2-Speed gearbox Max torque electric motors 850 Nm. Max torque rear axle 28 kNm. | The 4SPCR engine that offers 100hp of power along with 300Nm of torque from 1,200 to 2,200rpm and brushless asynchronous induction motor |
| • PERFORMANCE | Up to 225kw power (300hp) | 100hp |
| • ELECTRIC MOTOR TORQUE FOR PTO(PEAK/CONTINUOUS) | 530 Nm/270 Nm | 300Nm |

| SPECIFICATIONS | Volvo Fe | Mack LR |
|---|---|---|
| • INVERTER | Traction | Traction |
| • WHEEL BASE | From 3 900 mm up to 5000 mm1 | 3310mm |
| • CONTROLLER | PI Controller | PI controller |

## EV MODEL DESIGN

- COST – ₹ 15.29 Lakh to ₹ 16.82 Lakh
- GROSS COMBINATION WEIGHT – 7300 kg
- RANGE – More than 100 KMs
- BATTERY - nickel-metal hydride
- BATTERY CAPACITY – 60–120 wH/kg
- CHARGING TIME(Full Charge) – 3-4hrs
- DRIVELINE/MOTOR – Permanent Magnet Synchronous Motor (PMSM)
- PERFORMANCE – 100hp
- ELECTRIC MOTOR TORQUE PTO(peak/continuous)- 180Nm
- INVERTER – Grid-tied string
- WHEEL BASE - 3310mm
- CONTROLLER – PID Controller

## Conclusion

The progress that the electric vehicle industry has seen in recent years is not only extremely welcomed, but highly necessary in light of the increasing global greenhouse gas levels. As demonstrated within the economic, social, and environmental analysis sections of this webpage, the benefits of electric vehicles far surpass the costs. The biggest obstacle to the widespread adoption of electric-powered transportation is cost related, as gasoline and the vehicles that run on it are readily available, convenient, and less costly. As is demonstrated in our timeline, we hope that over the course of the next decade technological advancements and policy changes will help ease the transition from traditional fuel-powered vehicles. Additionally, the realization and success of this industry relies heavily on the global population, and it is our hope that through mass marketing and environmental education programs people will feel incentivized and empowered to drive an electric-powered vehicle. Each person can make a difference, so go electric and help make a difference!

**Mini project 9 – Parking System [Individual]**

**Module:- Classic Autosar Basic to Intermediate**

## Requirements

### High Level Requirement:

| High level Requirement | Description |
|---|---|
| HLR_1 | Light will be on when Door is open |
| HLR_2 | Light will be in on state until all the door is correctly closed |
| HLR_3 | Dashboard Light will ON when the car is unlocked |
| HLR_4 | Lights will be OFF after 10 sec when the lock button pressed on the key |
| HLR_5 | In Night Foot step light is automatically ON |
| HLR_6 | Lights can be turn off and on manually with the switch |

### Low Level Requirement:

| Low level Requirement | Description |
|---|---|
| LLR_1 | Voice command to turn on and off the light |
| LLR_2 | Lights will be turned on when unlocks the car with the remote key from outside |
| LLR_3 | Add Multicolour LED |
| LLR_4 | The light will turn-off after 5 seconds when we lock the car through the key |
| LLR_5 | Safety Indicator |

## Design

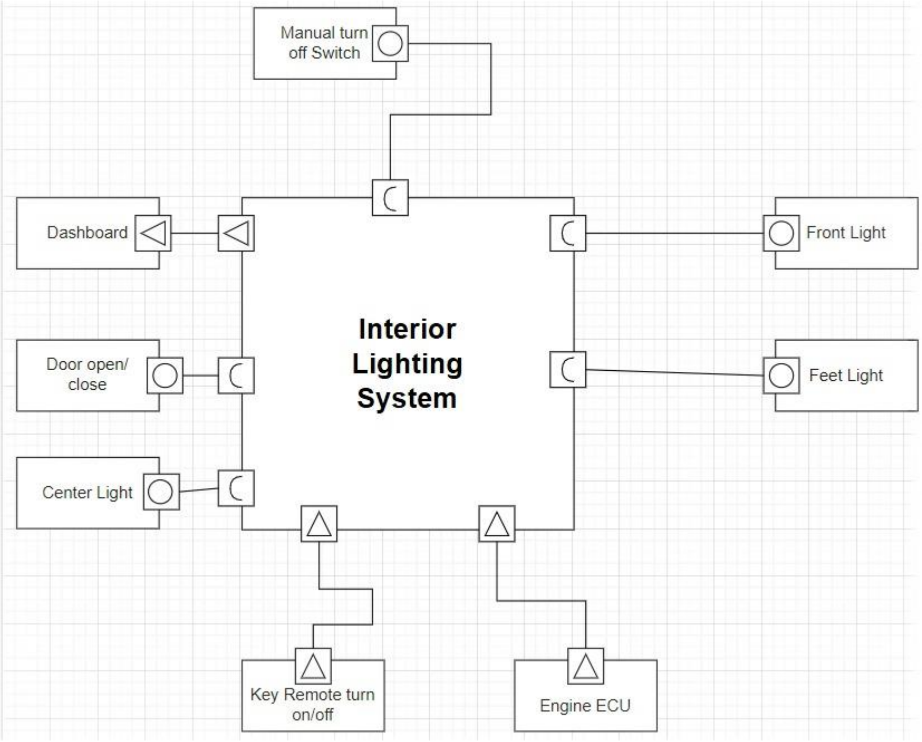


Figure 5 VFB Diagram

## Individual Contribution and Highlights

1. Interior Light Control Case Study
2. Source code management using GitHub
3. AtomicSwComponent
4. SWCInternalBehavior
5. SWCImplementation

## Learning of Essential of python

## Outcomes:

Lesson 1: print()
It’s the tradition. Print **“Hello World!”**

Lesson 2: Variables
Variables are probably the most fundamental building blocks in high-level programming. Learn and Practice variables with this Python course.

Lesson 3: Data Types
Learn Python data types: int, float and str. They have different functions to store, process and represent different types of data.

Lesson 4: Type Conversion
Sometimes it makes sense to convert Python data types between each other (when possible). int, float and str are also functions for converting data types. And when you’re not sure of a variable’s type, you can use type function!

Lesson 5: Data Structures
In this Python course data gets a bit more structured. Python lists, dictionaries and tuples are famous sequences that can contain various type of data. You will learn most common Python data structures along with functions to create them.

Lesson 6: Lists
A closer look at Python lists and some of their built-in methods and functions. This lesson introduces a lot of fundamental Python topics but it’s so worth it. Make sure you take your time and get comfortable with Python lists as you will be using them a lot.

Lesson 7: Tuples
Python tuple concept, difference between tuple and list along with some tuple examples and built-in tuple methods in Python.

Lesson 8: Dictionaries

Python data structure: dictionaries will be unraveled in detail. Python Dictionaries new perspectives to data such as usage of key and unindexed structure.

Lesson 9: Strings

Good ol' strings revisited. More string methods, more built-in functions and more string examples. When you think about string, it's everywhere. Web data, reports, news, social media, books, descriptive text, user input, survey answers, gui and many more. So it deserved a revisit.

Lesson 10: len()
A practical Python function to get length of different types of data in Python.

## Lesson 11: .sort()

This list method can be very useful to sort data in a list. Later in intermediate lessons its cousin **sorted** function will be introduced along with slightly more advanced concepts. sort is a list method while sorted is a built-in function.

## Lesson 12: .pop() method

In this course an interesting dictionary method, pop, will be introduced. It’s also an opportunity to polish our Python dictionary knowledge.

## Lesson 13: input()

One of the most exciting Python function for many beginners, input allows interacting with users. You can ask questions or share messages with users and harvest their answers to use them in your computer program. Input also provides opportunities to practice Python data types.

## Lesson 14: range() |

Range function is practical and it can be used to create range objects in Python. Range objects are very useful when used with for loops and they can also be used to create lists of numbers (int or float) with different steps (default 1).

## Lesson18:PythonOperators

You have probably been using Python Operators all along. In this course you will be officially introduced to different Python operators

# Learnings of Electrical vehicles

## Domain Knowledge Videos

1. Understanding Hill Start Assist!- https://youtu.be/aXEPnWgRnjk?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
2. Differential | How does it work?- https://youtu.be/nC6fsNXdcMQ?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
3. Seatbelt | How does it work?- https://youtu.be/uRaU1HMJyCo?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
4. Understanding Wheel Alignment !- https://youtu.be/7d2K_mKgsZ0?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
5. How does the Steering Wheel automatically returns to its center?- https://youtu.be/wLbs8kBXgrw?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
6. Understanding your Car's Steering & Power Steering !- https://youtu.be/em1O8mz7sF0?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
7. Understanding Anti-lock Braking System (ABS) !- https://youtu.be/98DXe3uKwfc?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
8. Torque Converter, How does it work ?- https://youtu.be/bRcDvCj_JPs?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
9. Why you should not PARTIALLY press the Clutch ?- https://youtu.be/_hKvS6xTC0E?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
10. Clutch, How does it work ?- https://youtu.be/devo3kdSPQY?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
11. Electric cars vs Petrol cars- https://youtu.be/ewcWN-rHQ6Q?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
12. How does an Electric Car work ? | Tesla Model S- https://youtu.be/3SAxXUIre28?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
13. Understanding PLANETARY GEAR set !- https://youtu.be/ARd-Om2VyiE?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
14. Automatic vs Manual Transmission- https://youtu.be/auQgOtveQi0?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
15. Working of Dual Clutch Transmission (DSG)- https://youtu.be/lFAtc-zOKZs?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
16. Manual Transmission, How it works ?- https://youtu.be/wCu9W9xNwtI?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
17. Automatic Transmission, How it works ?- https://youtu.be/u_y1S8C0Hmc?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
18. Petrol (Gasoline) Engine vs Diesel Engine- https://youtu.be/bZUoLo5t7kg?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
19. Diesel Engine, How it works ?- https://youtu.be/DZt5xU44IfQ?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg
20. How a Differential works ?- https://youtu.be/SOgoejxzF8c?list=PLuUdFsbOK_8rJsh_osoqVKfIRUkb8-rOg

## Electrical vehicle basics:

1. EPT trainings learning content

   Video link :EPT Trainings Learning Content | Microsoft Stream (mcas.ms)

   1. EV Architecture and components

   2. Inverter Hardware and software -part 1

   3. Inverter Hardware and

   software -part 24.EV Lab

   and testing training

   5.Worst case analysis Tolerance analysis

   6. Design calculations -Inverter losses & thermal design

   7.Hardware Simulation and control simulation of Dc Dc

   converter topologies8.software closed loop control DC -DC

   Converter topologies

2.BMS (Battery management system)

   Video link EV Learning Content | Microsoft

   Stream (mcas.ms)1.system requirements

   ,specification feature and DFMEA 2.BMS -

   software application and Algorithm

   3.FUSA-1

   4.FUSA-2

   5.Wireless BMS

   6.BMS testing and BI HIL

   7.EV lab Demo and

   Amaze BMS 8.Overall

   BMS architecture and

   platform

3.System level -conventional/EV

   Video link : System Level - Conventional/EV | Microsoft Stream (mcas.ms)

   1. Inviting for Battery Management System
   2. introduction to Functional Safety
   3. Function Safety Session 2
   4. Overview of Engine After treatment System, Engine Sensors and transmissionSystem
   5. Inviting for Battery Management System – Session 2

6. Overview of different Vehicle architectures
7. DC – DC converter