

SUBJECT:- ROBOTICS PRACTICAL

INDEX

Sr. No.	TOPICS	Page No.
1	making a Raspberry Pi headless, and reaching it from the network using WiFi and SSH	2
2	Using sftp upload files from PC.	8
3	Write Python code to test motors.	14
4	Write a script to follow a predetermined path	15
5	Develop Python code for testing the sensors.	19
6	Add the sensors to the Robot object and develop the line-following behaviour code.	21
7	Using the light strip develop and debug the line follower robot	26
8	Add pan and tilt service to the robot object and test it	31
9	Create an obstacle avoidance behavior for robot and test it.	37
10	Detect faces with Haar cascades	42
11	Use the robot to display its camera as a web app on a phone or desktop, and then use the camera to drive smart color and face-tracking behaviours	44
12	Use a Raspberry Pi to run the Mycroft environment and connect it to a speaker/microphone combination	48

Practical no :-1

Aim: Making Raspberry pi headless, and reaching it from the network using wifi and SSH.

Components Required:-

SD Card -16 Gb ,Desktop

Procedure :-

STEP I: Go to www.raspberry.com and click on software tab. Download raspberry pi imager for windows. And connect SD card to laptop.



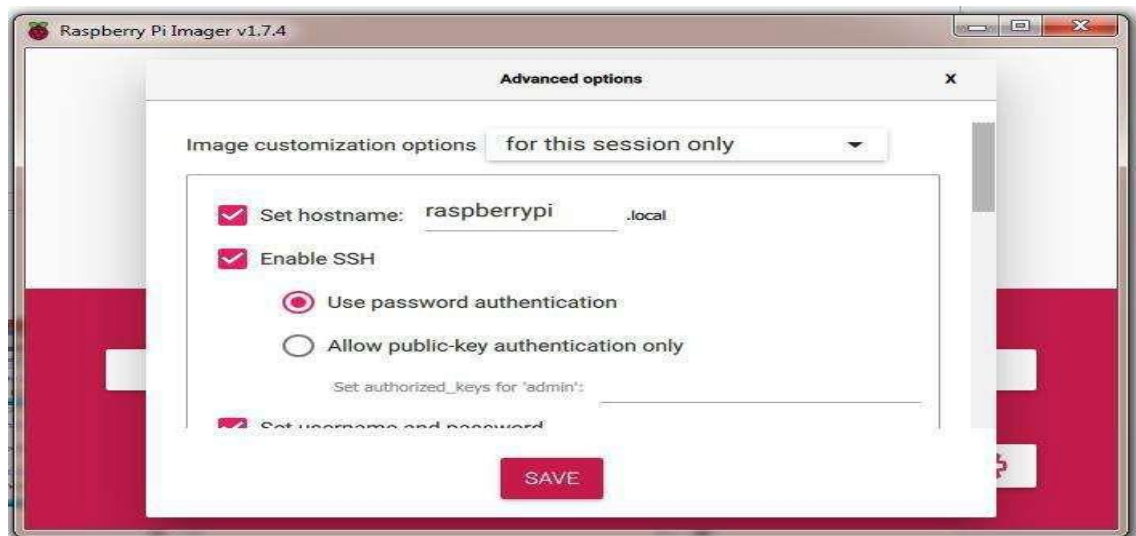
STEP II : Select operating system (raspberry pi OS 32-bit)



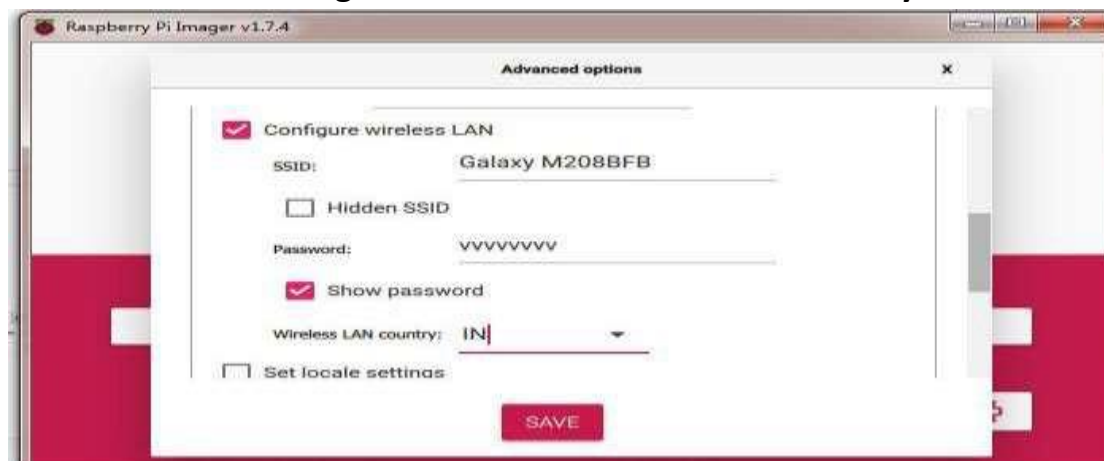
STEP III : Select Storage



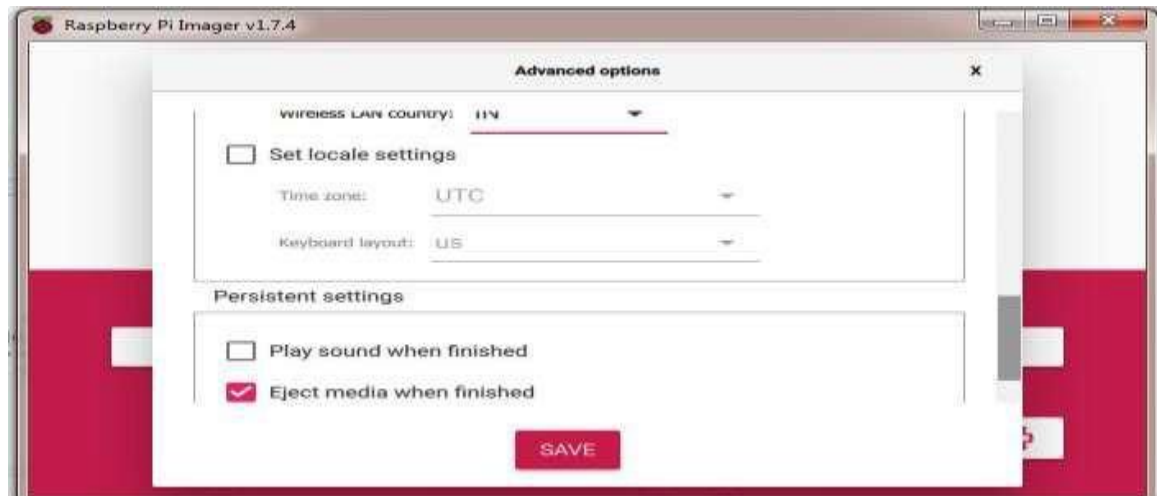
STEP IV : Click on Setting Icon . and set hostname. Click to enable SSH.



STEP V : Click to configure wireless LAN. And select country.



STEP VI : Click to save.



STEP VII : Click on write.



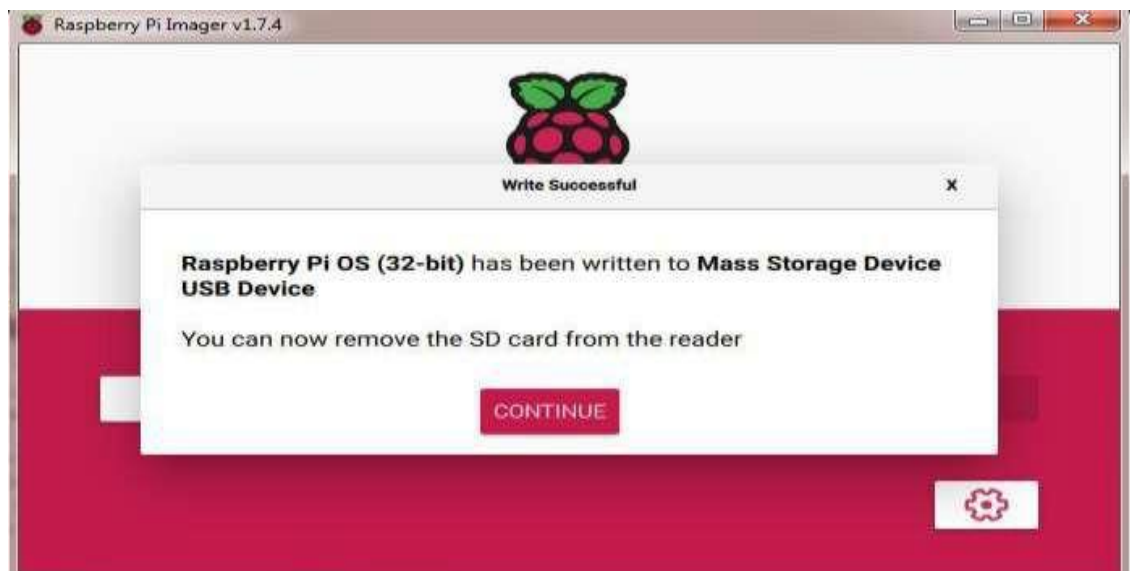
STEP VIII : Click on yes.



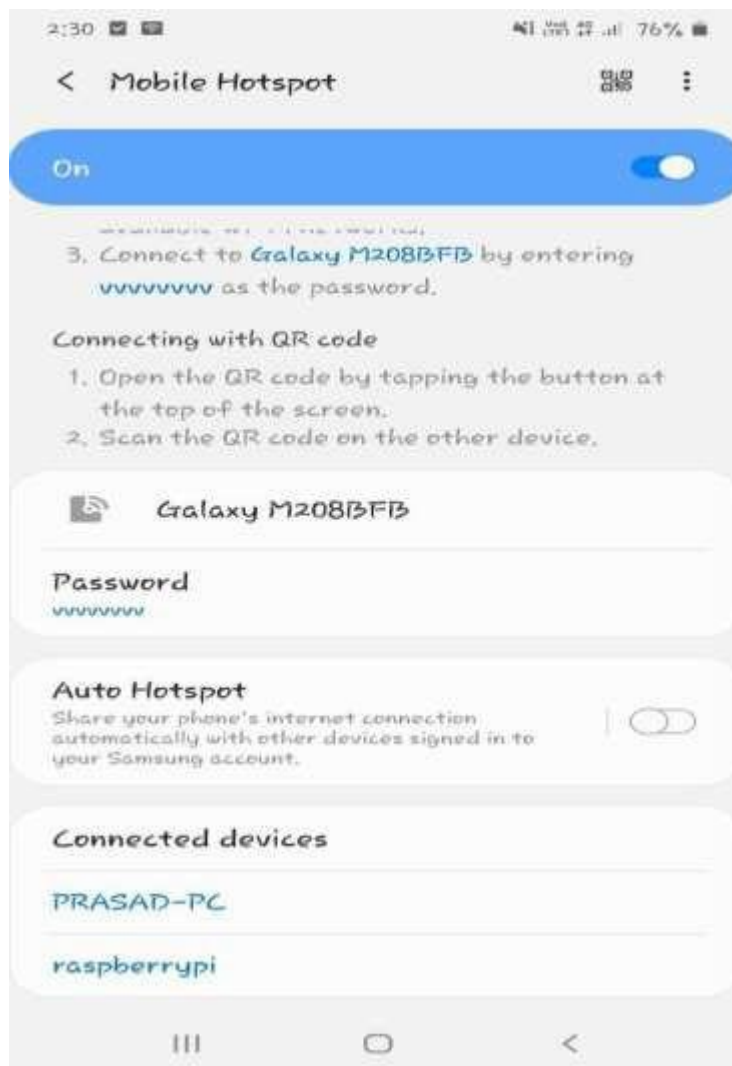
STEP IX : Verifying raspberry pi .



STEP X : Click on continue .

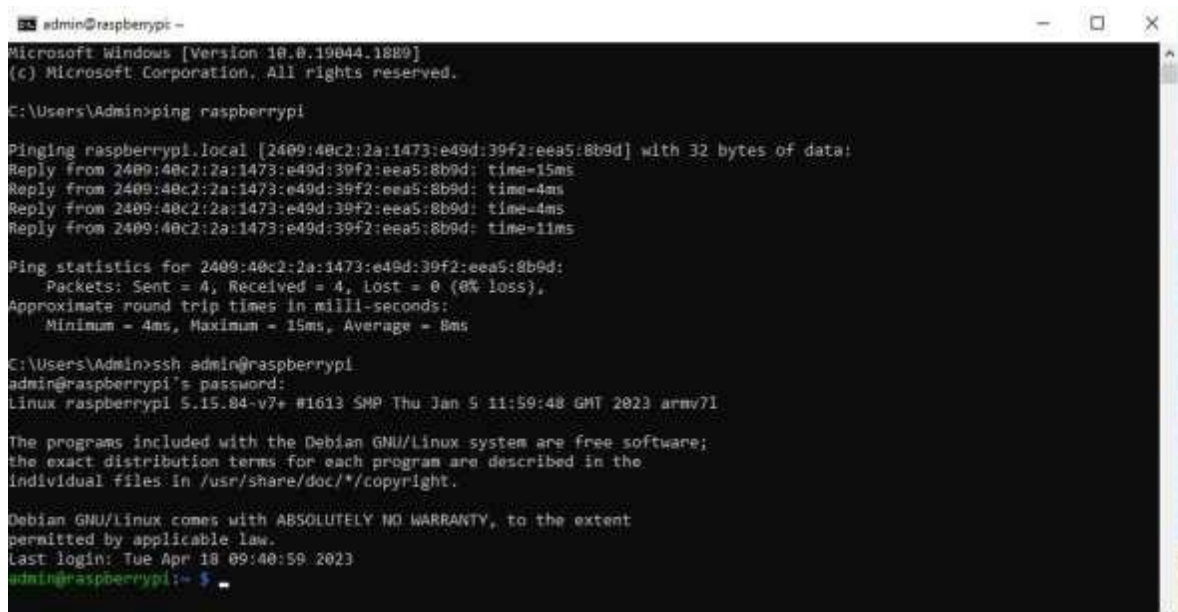


Step XI : check the connection in your mobile hotspot of raspberrypi



STEP X: On cmd prompt execute these command :

- 1. ping raspberrypi**
- 2. ssh admin@raspberrypi**



```
admin@raspberrypi ~
Microsoft Windows [Version 10.0.19044.1889]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>ping raspberrypi

Pinging raspberrypi.local [2409:40c2:2a:1473:e49d:39f2:eea5:8b9d] with 32 bytes of data:
Reply from 2409:40c2:2a:1473:e49d:39f2:eea5:8b9d: time=15ms
Reply from 2409:40c2:2a:1473:e49d:39f2:eea5:8b9d: time=4ms
Reply from 2409:40c2:2a:1473:e49d:39f2:eea5:8b9d: time=4ms
Reply from 2409:40c2:2a:1473:e49d:39f2:eea5:8b9d: time=11ms

Ping statistics for 2409:40c2:2a:1473:e49d:39f2:eea5:8b9d:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 4ms, Maximum = 15ms, Average = 8ms

C:\Users\Admin>ssh admin@raspberrypi
admin@raspberrypi's password:
linux raspberrypi 5.15.84-v7+ #1613 SMP Thu Jan 5 11:59:48 GMT 2023 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

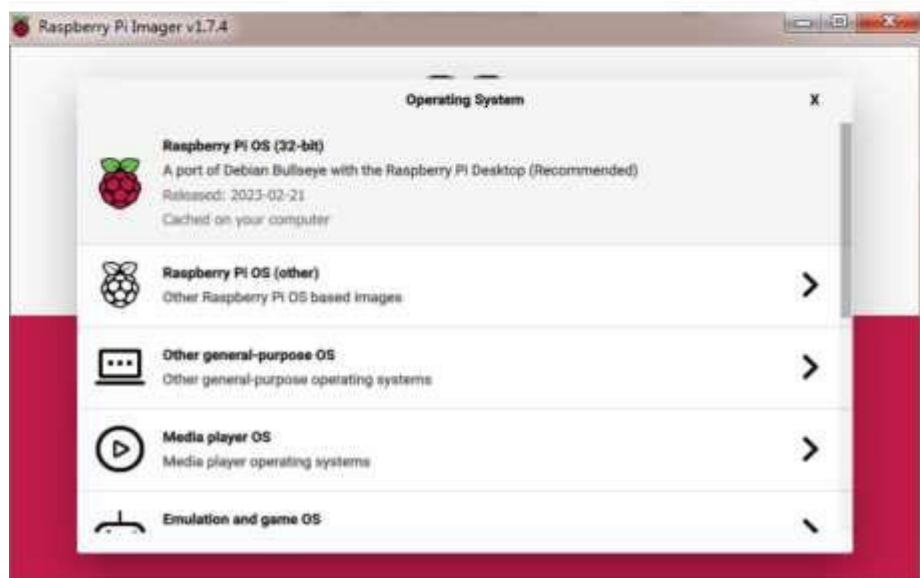
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
last login: Tue Apr 18 09:40:59 2023
admin@raspberrypi:~$
```

Practical 2

Aim:-Using sftp upload files from PC.

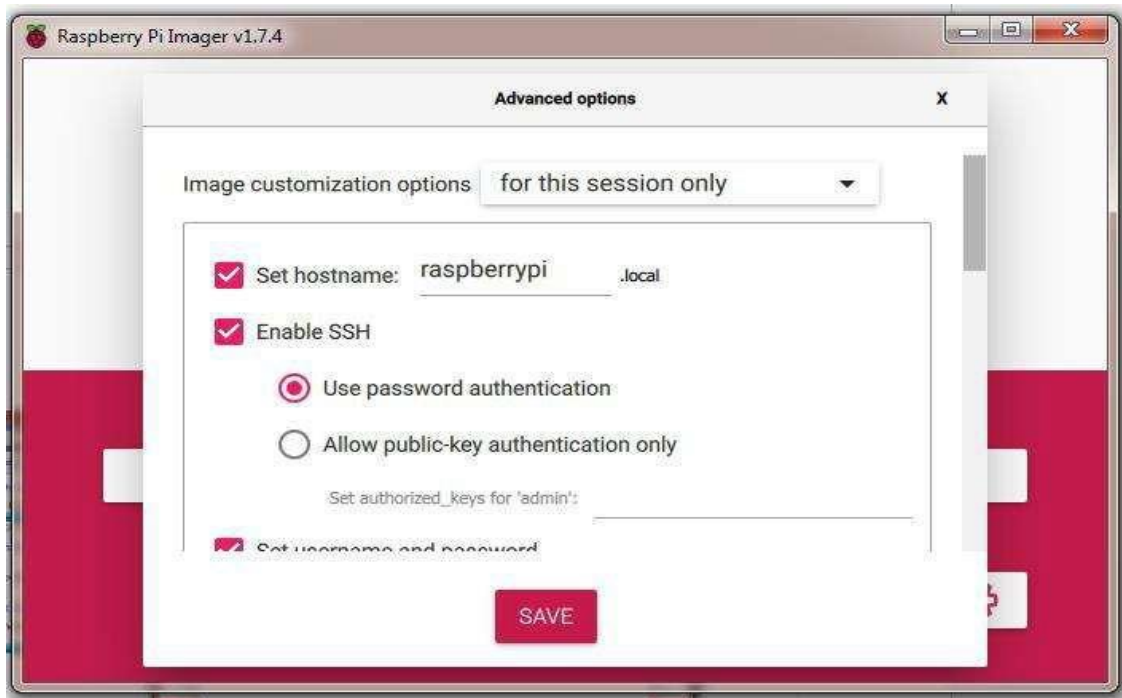
Software required:-Filezilla, Github

Step 1: Install the Raspberry Pi Imager

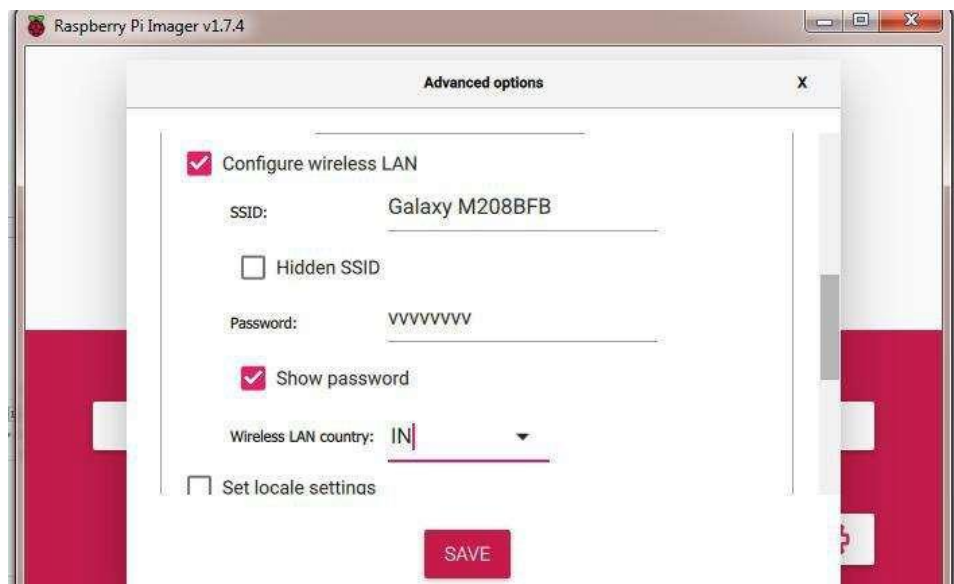


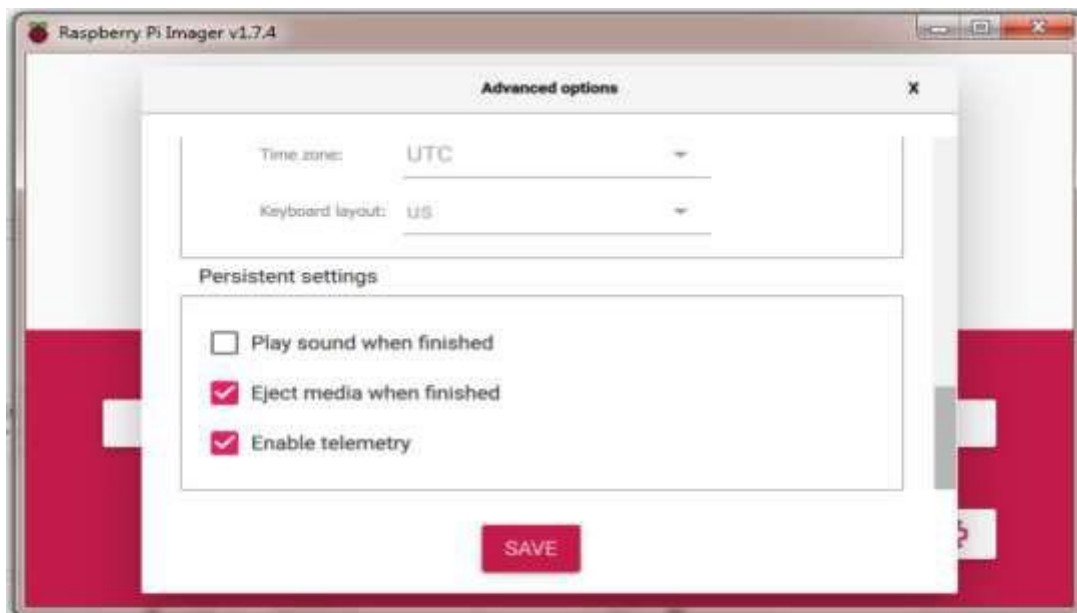
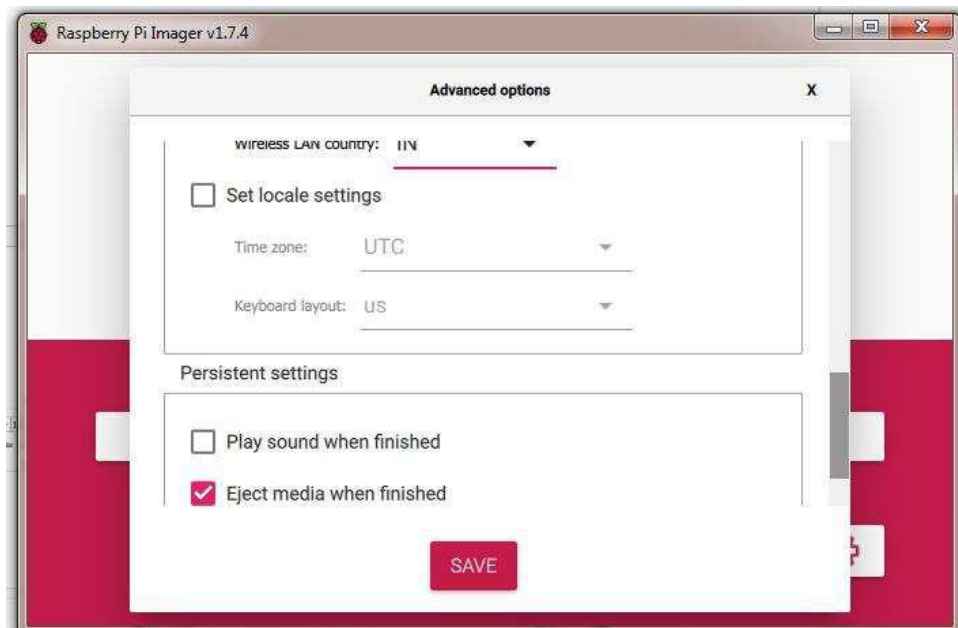


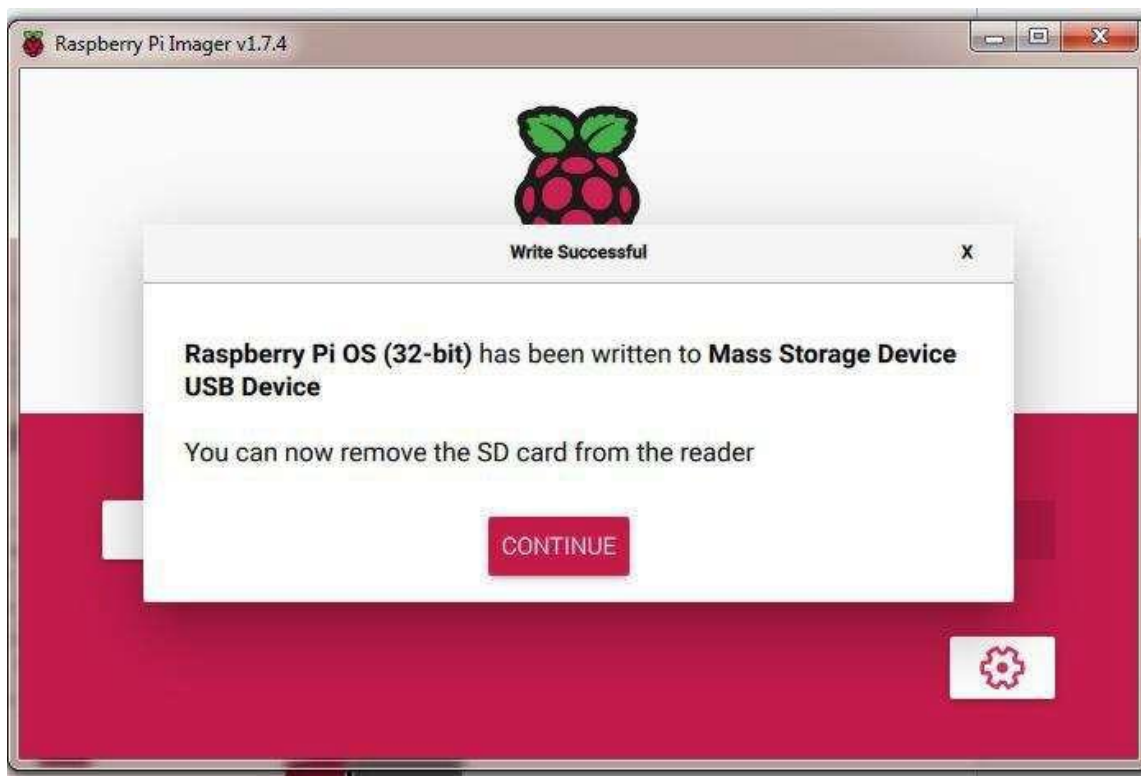
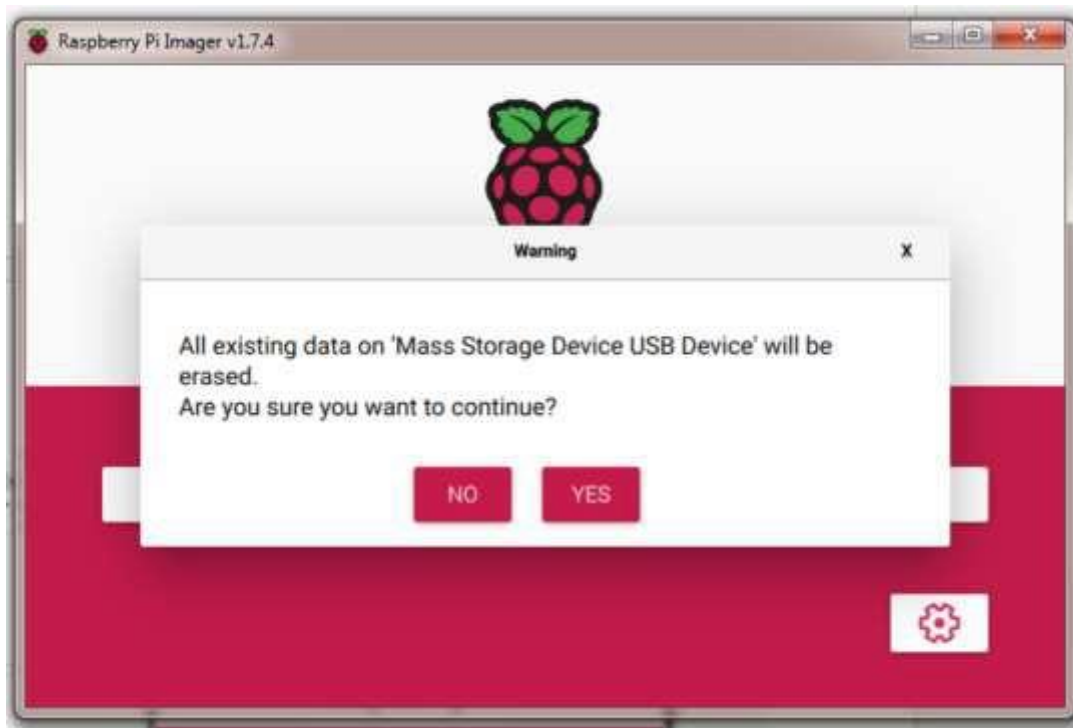
Step 2: Set Host Name , enable SSH, Set Username and Password.



Step 3: Set SSID and Password of hotspot which is used .







Step 4: Connect Raspberry Pi WIFI and Laptop WIFI to Mobile Device



Step 5: Open CMD and type following command

I) ping raspberrypi or ping 162.168.207.244

II) ssh admin@ raspberrypi or ssh

admin@ 162.168.207.244

And type password of Admin

```
admin@raspberrypi: ~
Request timed out.

Ping statistics for 192.168.207.244:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\Users\Admin>ping 192.168.207.244

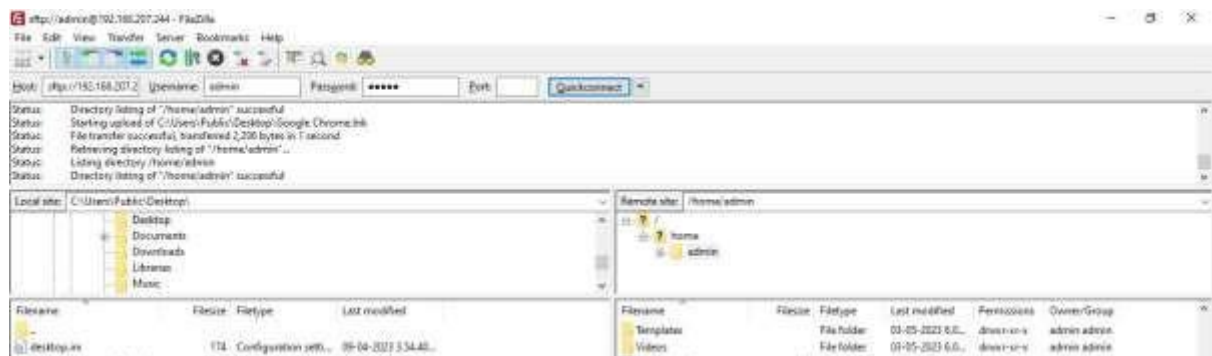
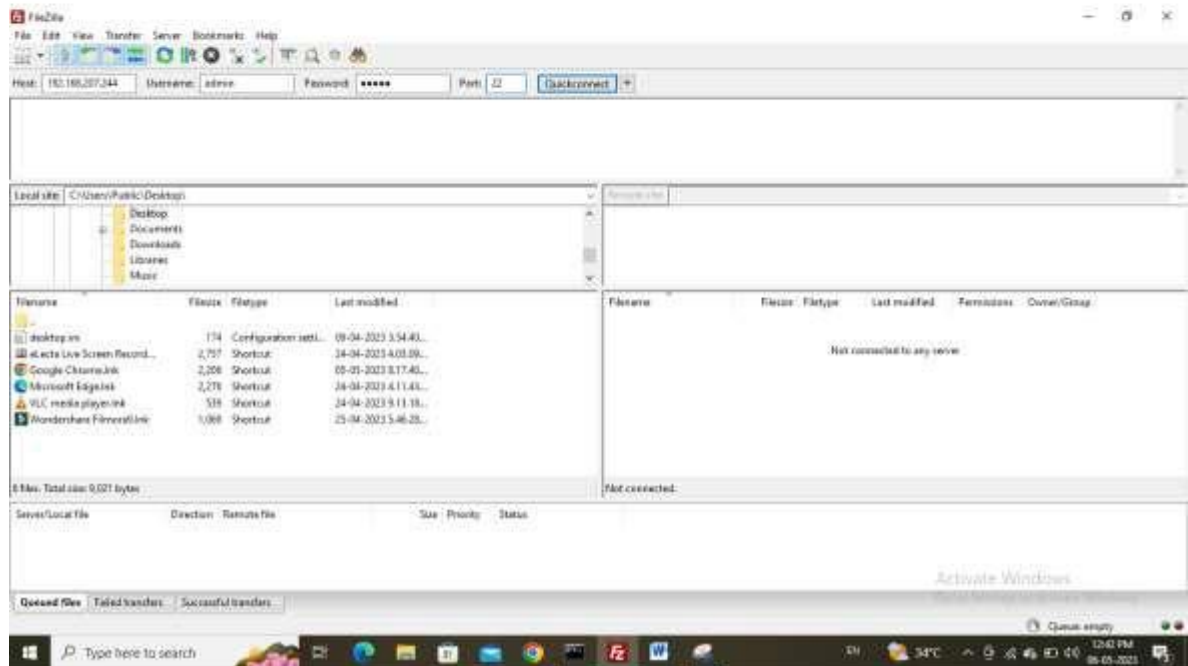
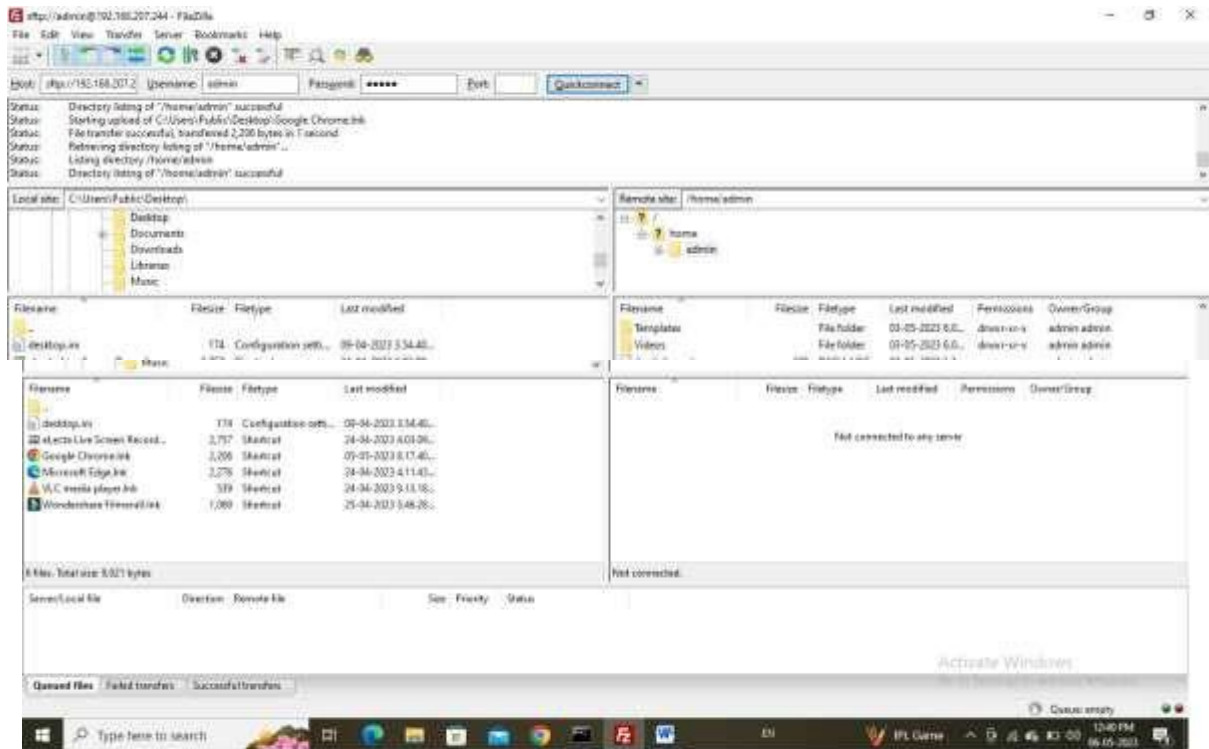
Pinging 192.168.207.244 with 32 bytes of data:
Reply from 192.168.207.244: bytes=32 time=21ms TTL=64
Reply from 192.168.207.244: bytes=32 time=11ms TTL=64
Reply from 192.168.207.244: bytes=32 time=9ms TTL=64
Reply from 192.168.207.244: bytes=32 time=10ms TTL=64

Ping statistics for 192.168.207.244:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 9ms, Maximum = 21ms, Average = 12ms

C:\Users\Admin>ssh admin@192.168.207.244
The authenticity of host '192.168.207.244 (192.168.207.244)' can't be established.
ECDSA key fingerprint is SHA256:qZw2aLXcb81PnFDfJMhtKANxs5KbGF1/X9PLVqS/Hb0.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.207.244' (ECDSA) to the list of known hosts.
admin@192.168.207.244's password:
Linux raspberrypi 6.1.21-v7+ #1642 SMP Mon Apr  3 17:20:52 BST 2023 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed May  3 06:07:06 2023
admin@raspberrypi:~ $
```



Practical No:-3

Aim :-Write a python code to test motors

Source Code:-

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)
button=12
DC_motor_a=7
DC_motor_b=11
GPIO.setup(DC_motor_a,GPIO.OUT)
GPIO.setup(DC_motor_b,GPIO.OUT)
GPIO.setup(button,GPIO.IN,pull_up_down=GPIO.PUD_UP)

while(1):
    if GPIO.input(button)==GPIO.LOW:
        GPIO.output(DC_motor_a,GPIO.HIGH)
        GPIO.output(DC_motor_b,GPIO.LOW)
        time.sleep(0.1)

    else:
        GPIO.output(DC_motor_a,GPIO.LOW)
        GPIO.output(DC_motor_b,GPIO.HIGH)
        time.sleep(0.1)
```

Practical NO -4

Aim :-Write a script to follow a predetermined path.

Components:-

Raspberry pi Board3

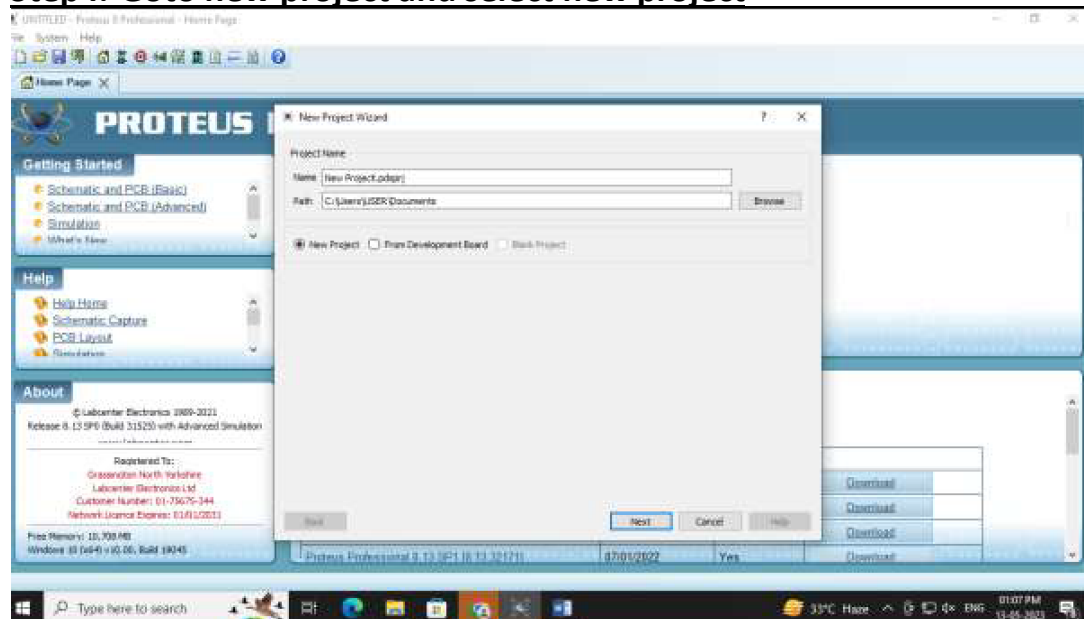
L293D

Simple DC Motor

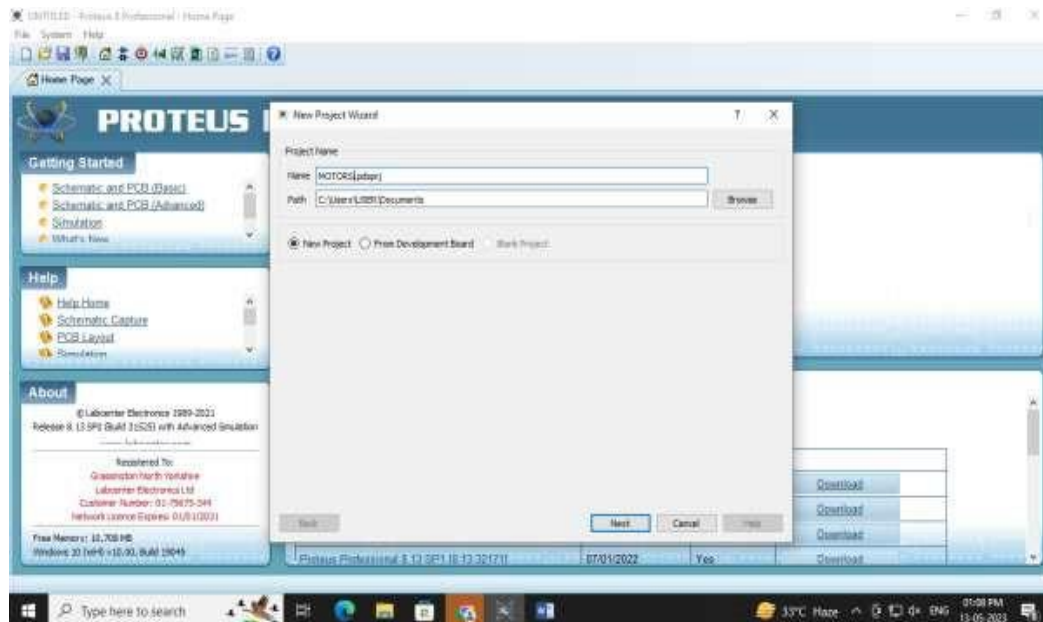
Button

Process of Creation of Project:-

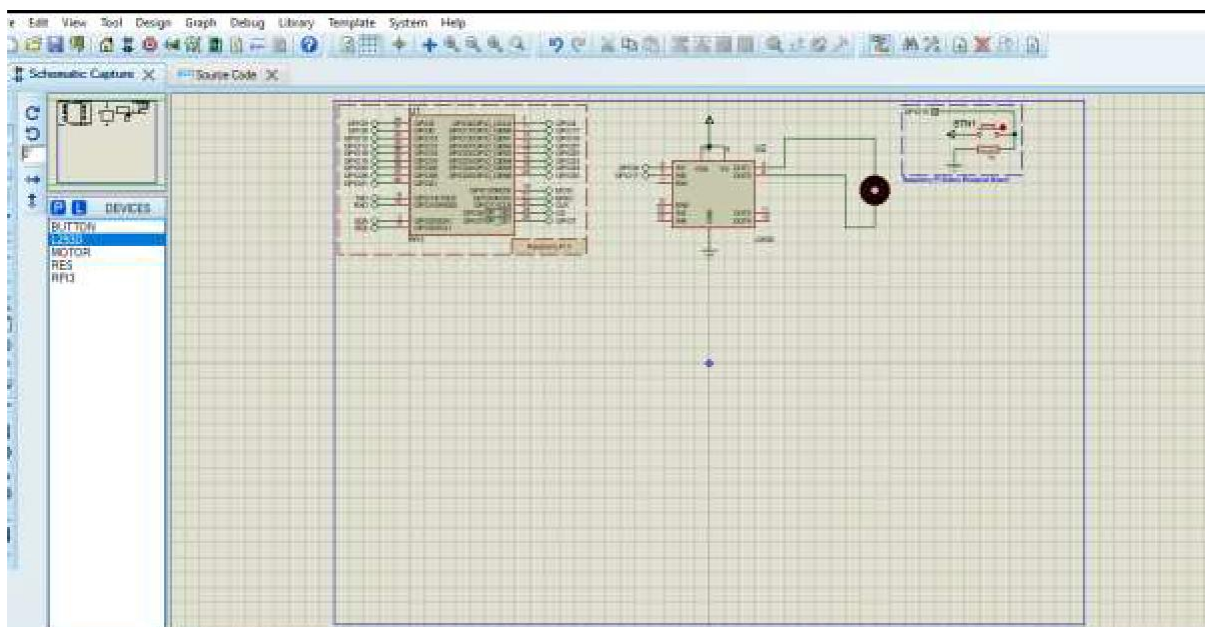
Step I:-Goto new project and select new project



Step2:-Change the name of the project and save As Motors.psdprj



Step 3:-Implementation of the Circuit :-



Source Code:-

```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BOARD)
```



```
GPIO.setwarnings(False)
button=12
DC_motor_a=7
DC_motor_b=11

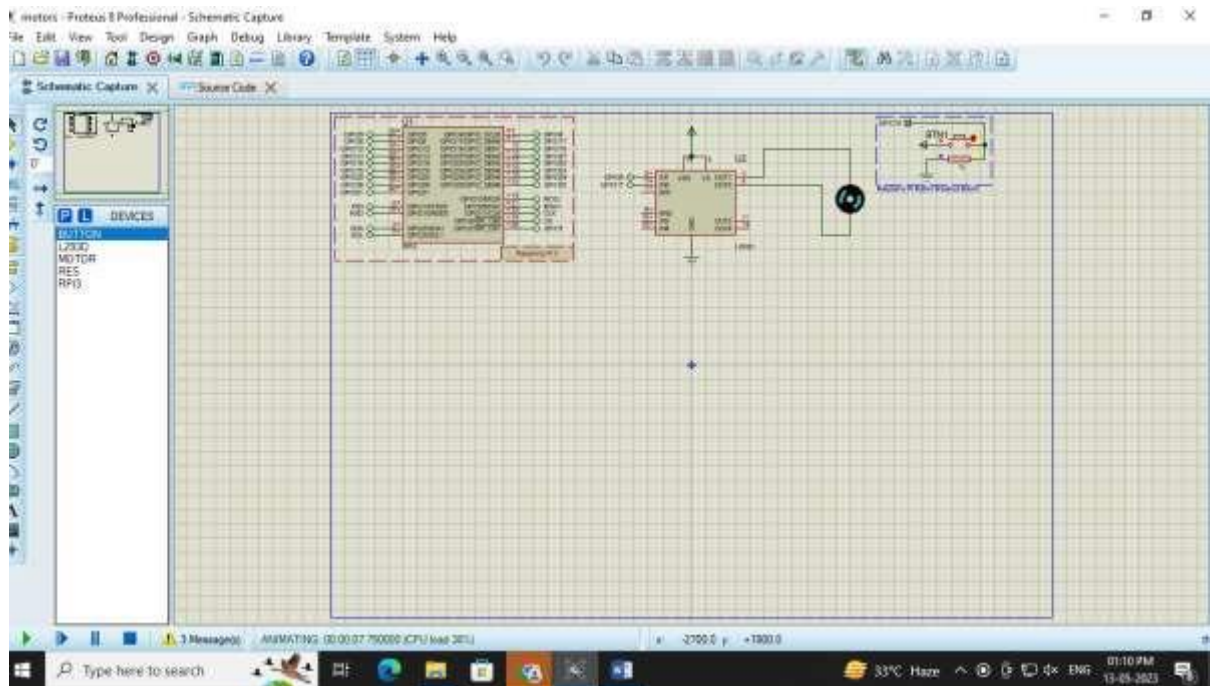
GPIO.setup(DC_motor_a,GPIO.OUT)
GPIO.setup(DC_motor_b,GPIO.OUT)
GPIO.setup(button,GPIO.IN,pull_up_down=GPIO.PUD_UP)

while(1):
    if GPIO.input(button)==GPIO.LOW:
        GPIO.output(DC_motor_a,GPIO.HIGH)
        GPIO.output(DC_motor_b,GPIO.LOW)
        time.sleep(0.1)

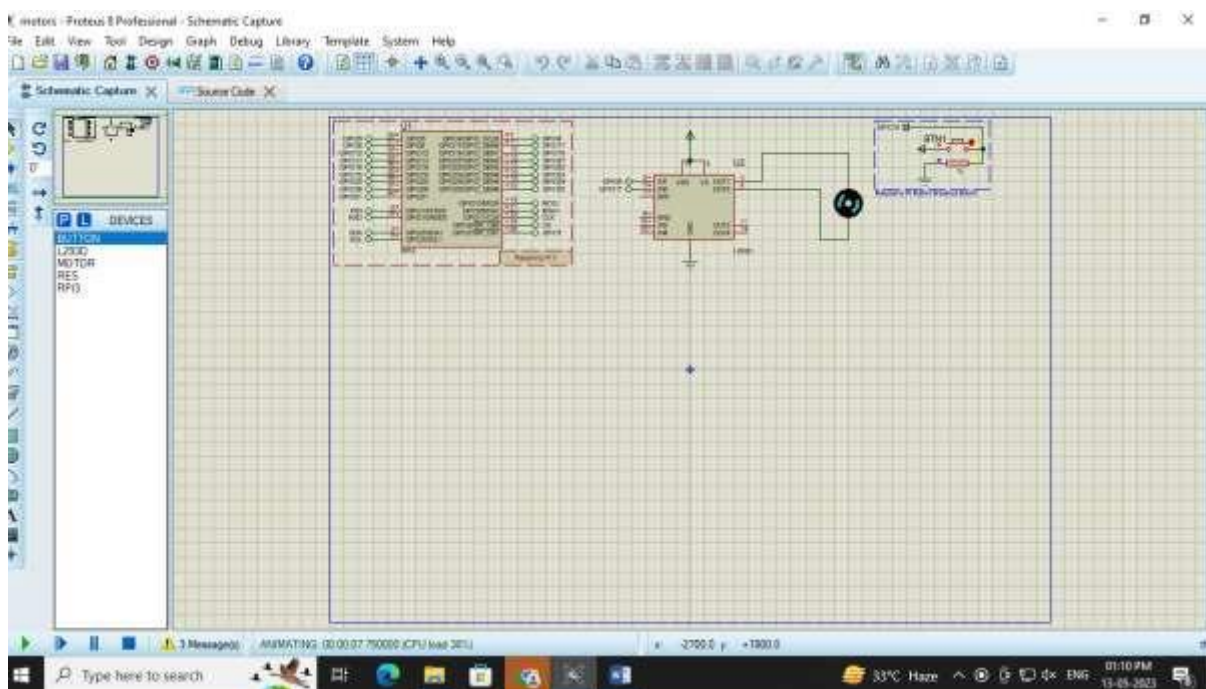
    else:
        GPIO.output(DC_motor_a,GPIO.LOW)
        GPIO.output(DC_motor_b,GPIO.HIGH)
        time.sleep(0.1)
```

Conclusion:- The movement of the motors are used to represent motion on a path given

Output:-
Motors moving in Clockwise direction



Motors moving in anticlockwise direction



Practical no 5:-

Aim: Develop Python code for testing the

sensors. Step 1: Place the following component in

TinkerCard. Compontes:

- a. PIR Sensor
- b. Resistor
- c. Piezo
- d. Arduino Uno R3
- e. LED RGB

Step 2:Type the following

```
code int pirsensor = 0;
```

```
void setup()
```

```
{
```

```
    pinMode(2, INPUT);
```

```
    pinMode(12, OUTPUT);
```

```
    pinMode(13, OUTPUT);
```

```
}
```

```
void loop()
```

```
{
```

```
    pirsensor = digitalRead(2);
```

```
    if (pirsensor == HIGH)
```

```
    {
```

```
        digitalWrite(13,HIGH);
```

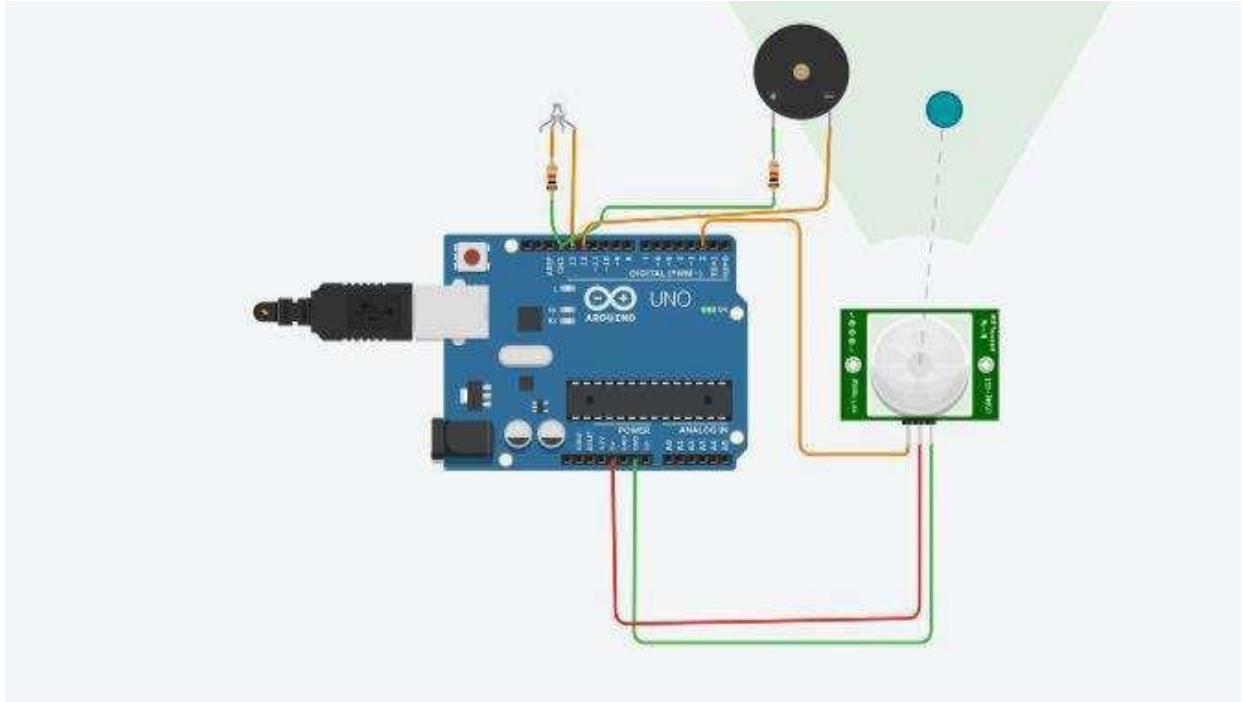
```
        tone(12,500,500);
```

```
    }
```

```
    digitalWrite(13,LOW)
```

```
}
```

Output:



Practical No -6

Aim:-Add the sensors to the robot object and develop the line follower behaviour code

Components required:-

Arduino

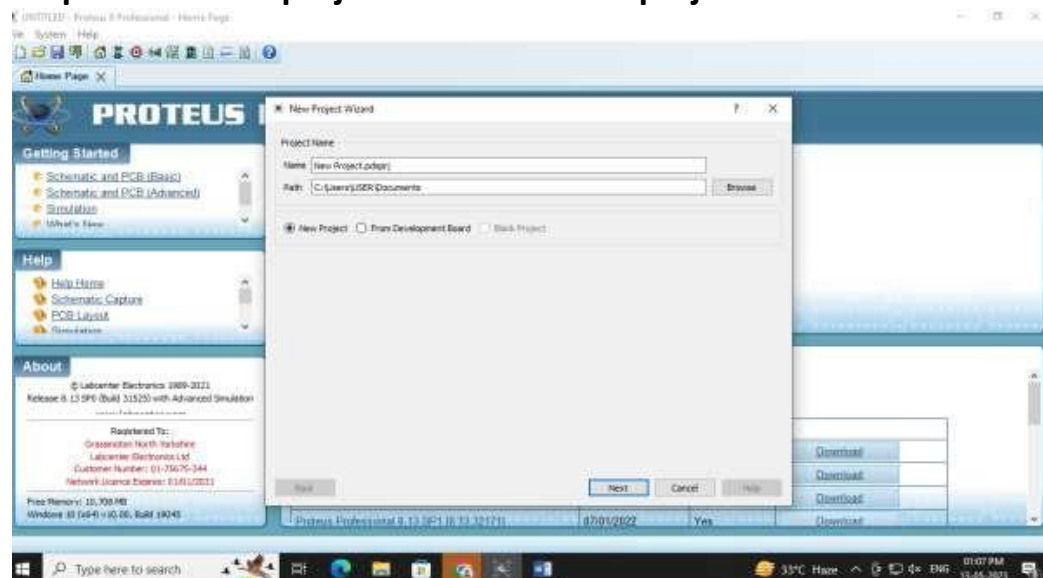
Button

Zumo robot

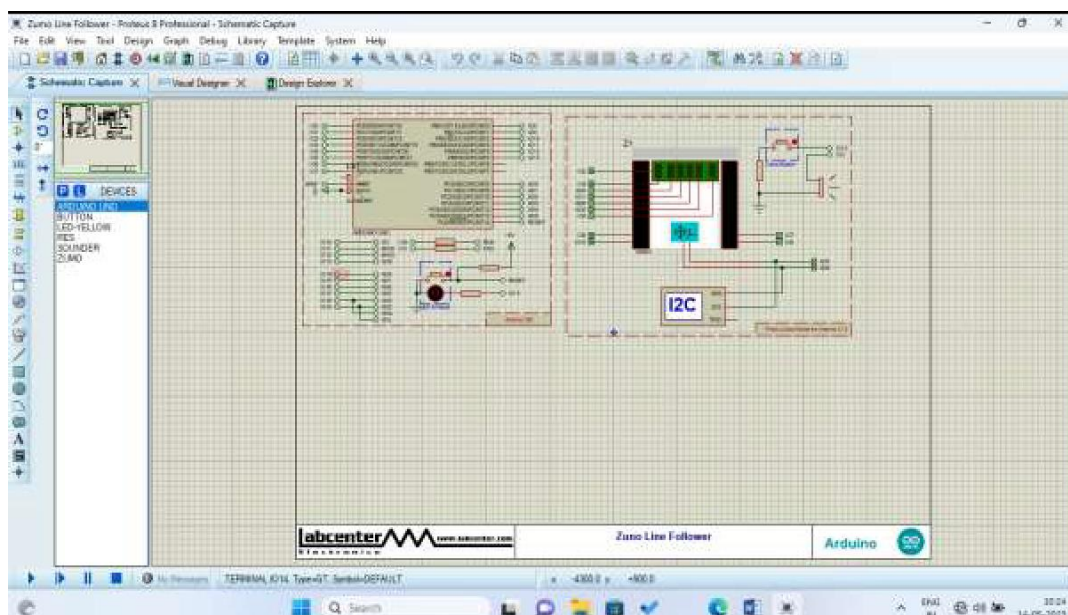
Proteus 8.13 simulator

Process of Creation of Project:-

Step 1:-Goto new project and select new project



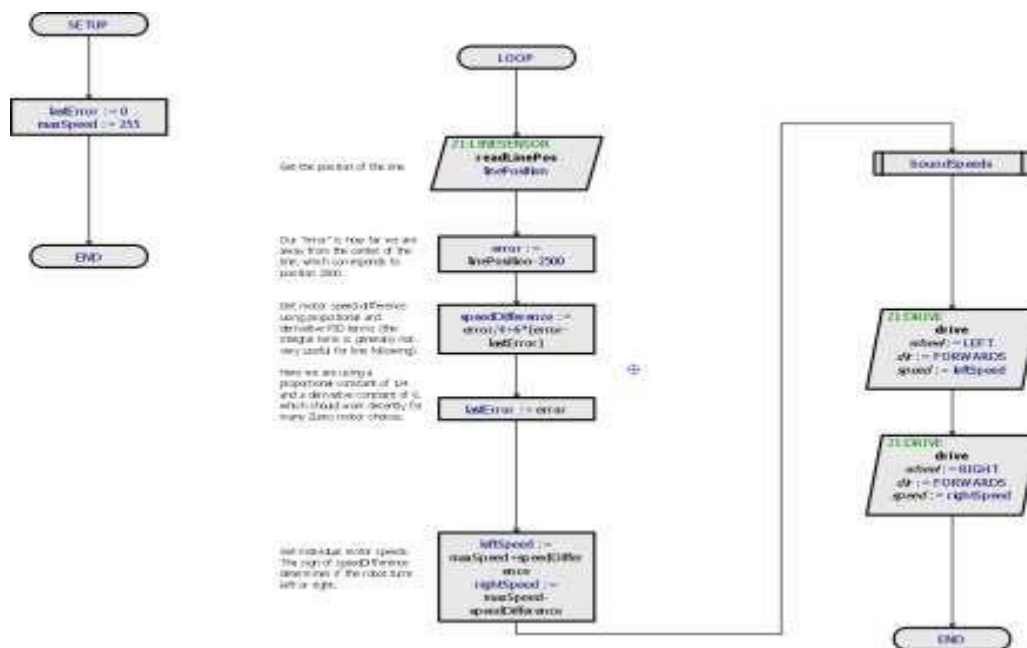
Step2:-Change the name of the project and save As Linefollower.psdprj that save it and Export Compiled Library Step 4: Upload this HEX file in Arduino



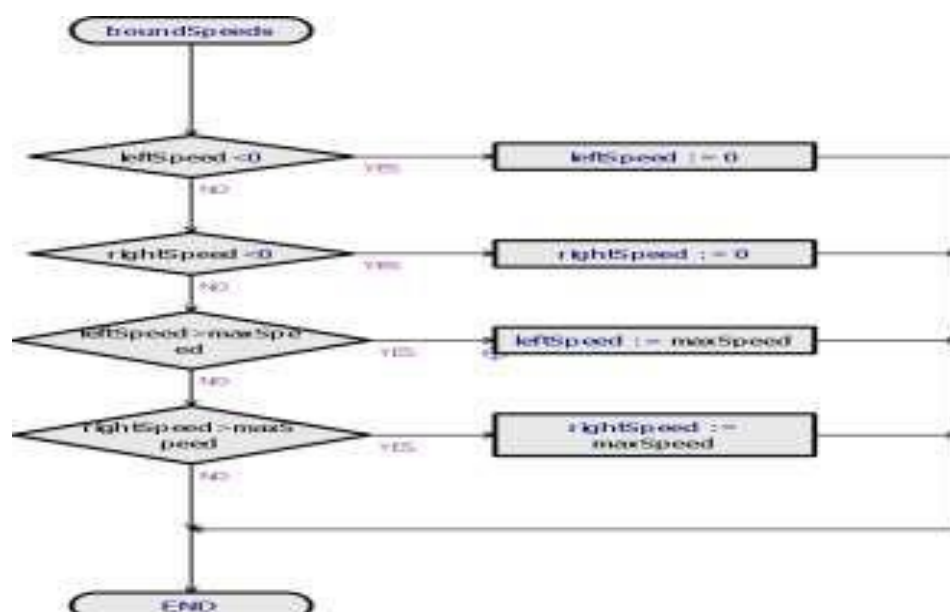
Flowchart for the project

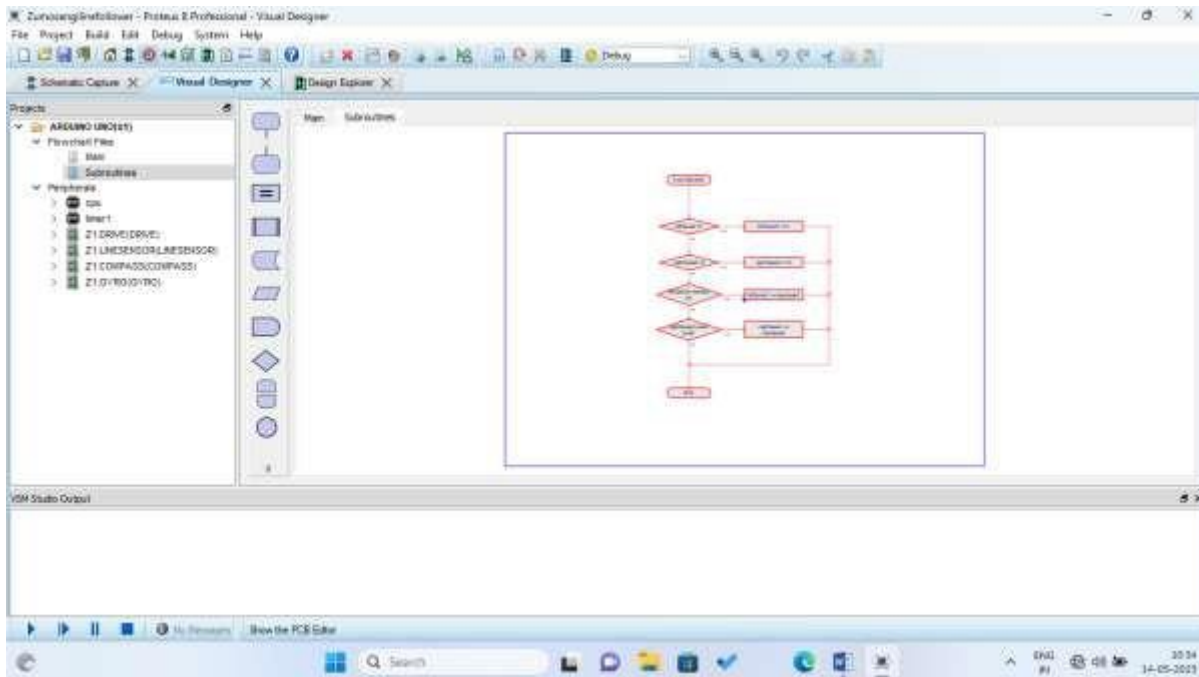
Flowchart for the project

MAIN FLOWCHART:-



Subroutine Flowchart:-





Source code:-

```
#pragma GCC push_options #pragma
GCC optimize ("Os")
```

```
#include <core.h> // Required by cpu
#include <cpu.h>
#include <TimerOne.h>
#include <L3G.h> // Required by Z1:DRIVE #include
<LSM303.h> // Required by Z1:DRIVE #include
<Wire.h> // Required by Z1:DRIVE #include <Servo.h>
// Required by Z1:DRIVE #include <Zumo.h>
```

```
#pragma GCC pop_options
```

```
// Peripheral Constructors
CPU &cpu = Cpu;
TimerOne &timer1 = Timer1;
DRIVE Z1_DRIVE = DRIVE (8, 10, 7, 9);
LINESENSOR Z1_LINESENSOR = LINESENSOR (4, A3, 11, A0, A2, 5, 2); COMPASS
Z1_COMPASS = COMPASS ();
GYRO Z1_GYRO = GYRO ();
```

```
void peripheral_setup () {
  Z1_DRIVE.begin ();
  Z1_LINESENSOR.begin ();
```

```

Z1_COMPASS.begin ();
Z1_GYRO.begin ();
}

void peripheral_loop() {
}
//---CONFIG_END---
// Flowchart Variables
long var_linePosition;
long var_error;
long var_lastError;
long var_speedDifference;
long var_leftSpeed;
long var_rightSpeed;
long var_maxSpeed;
float var_magX; float
var_magY; float
var_magZ;

// Flowchart Routines
void chart_SETUP() {
var_lastError=0,var_maxSpeed=255;
}

void chart_LOOP() {
var_linePosition=Z1_LINESENSOR.readLinePos();
var_error=var_linePosition-2500;
var_speedDifference=var_error/4+6*(var_error-var_lastError);
var_lastError=var_error;
var_leftSpeed=var_maxSpeed+var_speedDifference,var_rightSpeed=var_maxSpeed- var_speedDifference;
chart_boundSpeeds();
Z1_DRIVE.drive(1,1,var_leftSpeed);
Z1_DRIVE.drive(2,1,var_rightSpeed);
}

void chart_boundSpeeds() {
if(var_leftSpeed<0) { var_leftSpeed=0;
} else {
if(var_rightSpeed<0) {
var_rightSpeed=0;
} else {
if(var_leftSpeed>var_maxSpeed) {
var_leftSpeed=var_maxSpeed;
} else {
if(var_rightSpeed>var_maxSpeed) {
var_rightSpeed=var_maxSpeed;
}
}
}
}
}

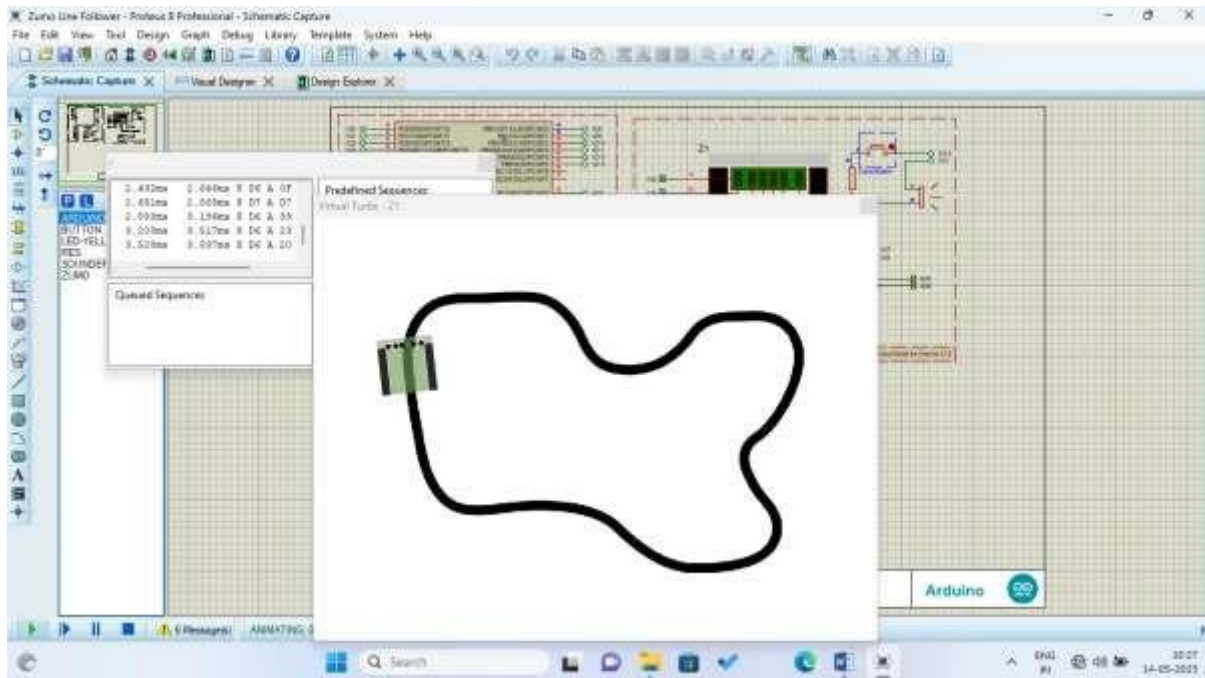
// Entry Points and Interrupt Handlers

```



```
void setup () { peripheral_setup(); chart_SETUP(); }  
void loop () { peripheral_loop(); chart_LOOP(); }
```

OUTPUT:



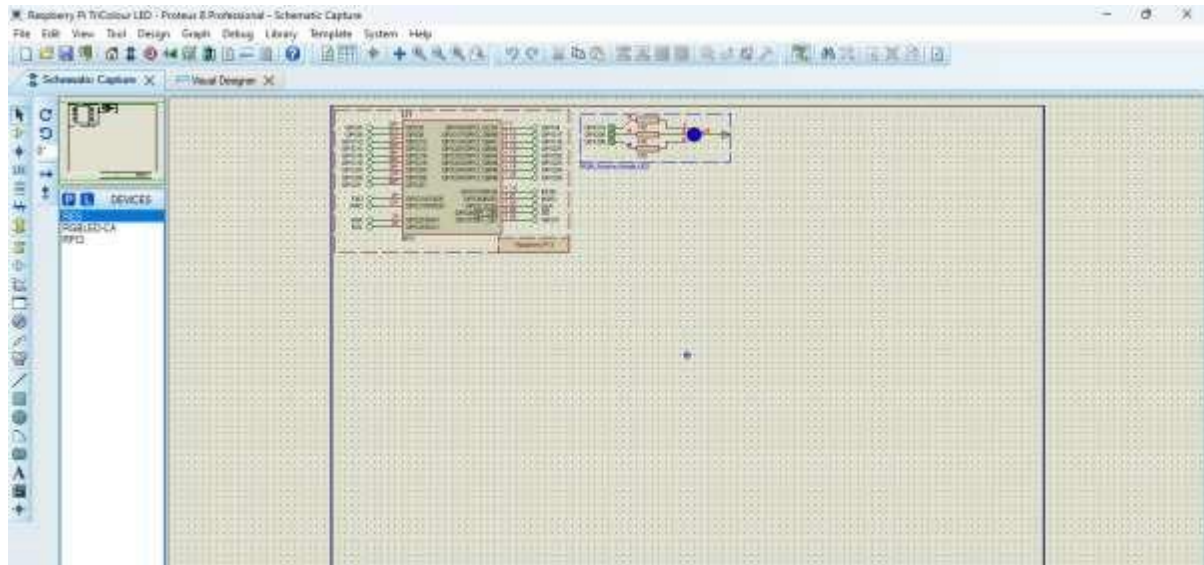
Practical no7:-

Aim:-Using Light strip to develop and debug the line follower robot

Components required:

Raspberry pr ,Strip rgb led

Circuit connection:-



Source Code in python

```
from goto import with_goto
from stddef import *
import var
import pio
import resource
from datetime import datetime
# Peripheral Configuration Code (Do Not Edit)
#---CONFIG_BEGIN---
import cpu
import FileStore
import timer
import VFP
import Generic
def peripheral_setup () :
# Peripheral Constructors
pio.cpu=cpu.CPU ()
pio.storage=FileStore.FileStore ()
pio.timer=timer.Timer ()
pio.server=VFP.VfpServer ()
pio.RGBLED1=Generic.RgbLedCa (pio.GPIO19, pio.GPIO20, pio.GPIO26)
pio.storage.begin ()
pio.server.begin (0)
# Install interrupt handlers
```

```

def peripheral_loop () :
    pio.timer.poll ()
    pio.server.poll ()
    #---CONFIG_END---
def variables_setup () :
    # Flowchart Variables
    pass
    # Flowchart Routines
    @with_goto
def chart_SETUP () :
    return
    @with_goto
def chart_LOOP () :
    pio.RGBLED1.set (True, True, True)
    sleep((500)*0.001)
    pio.RGBLED1.set (True, False, False)
    sleep((500)*0.001)
    pio.RGBLED1.set (True, True, False)
    sleep((500)*0.001)
    pio.RGBLED1.set (False, True, False)
    sleep((500)*0.001)
    pio.RGBLED1.set (False, True, True)
    sleep((500)*0.001)
    pio.RGBLED1.set (False, False, True)
    sleep((500)*0.001)
    pio.RGBLED1.set (True, False, True)
    sleep((500)*0.001)
    pio.RGBLED1.set (False, False, False)
    sleep((500)*0.001)
    return

# Main function
def main () :
    # Setup
    variables_setup ()
    peripheral_setup ()
    chart_SETUP ()
    # Infinite loop
    while True :
        peripheral_loop ()
        chart_LOOP ()
    # Command line execution
    if __name__ == '__main__' :
        main()

```

Flowchart of project:
OUTPUT:-

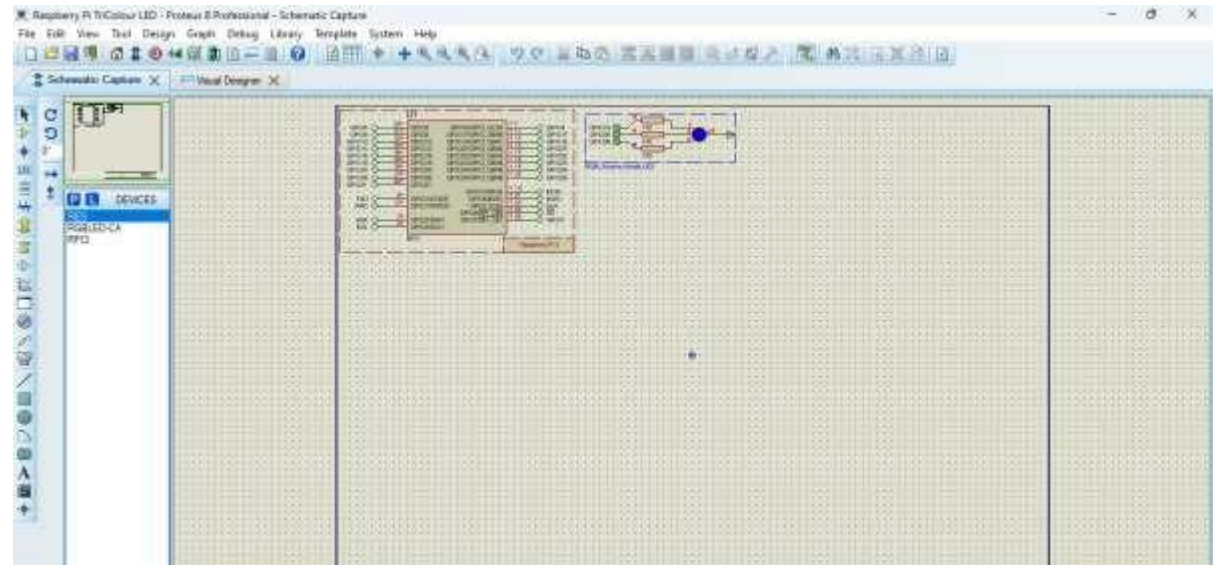
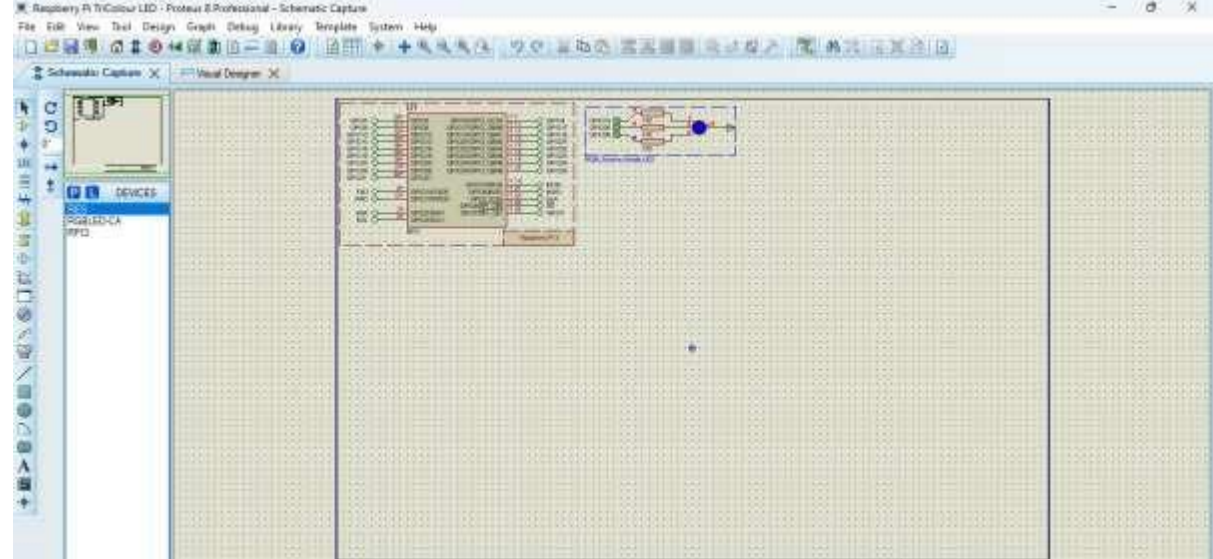


Figure of the led showing blue color

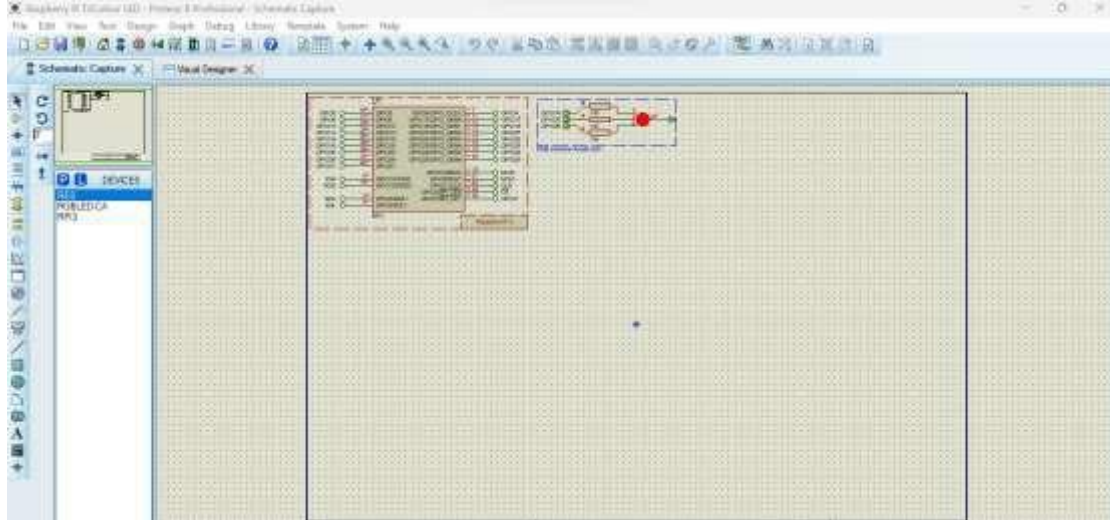


Figure of the led showing red color

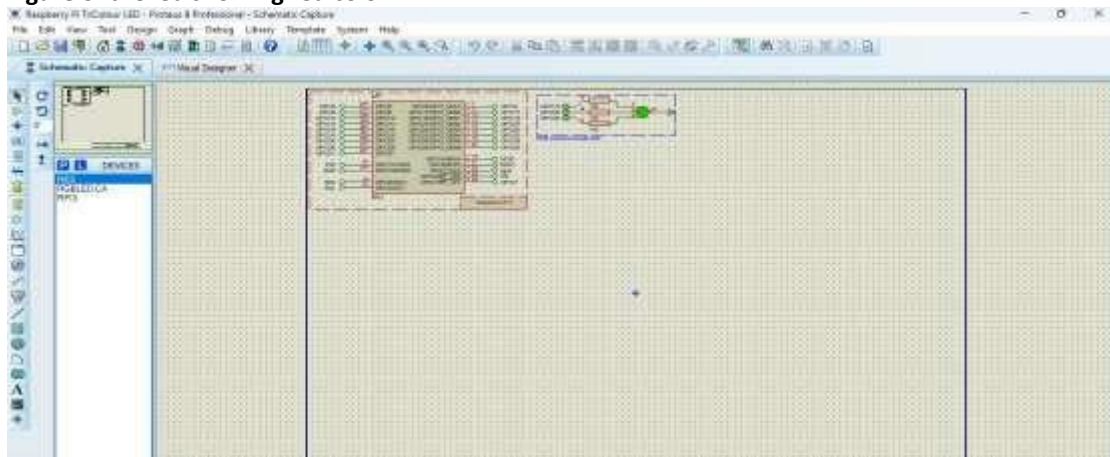
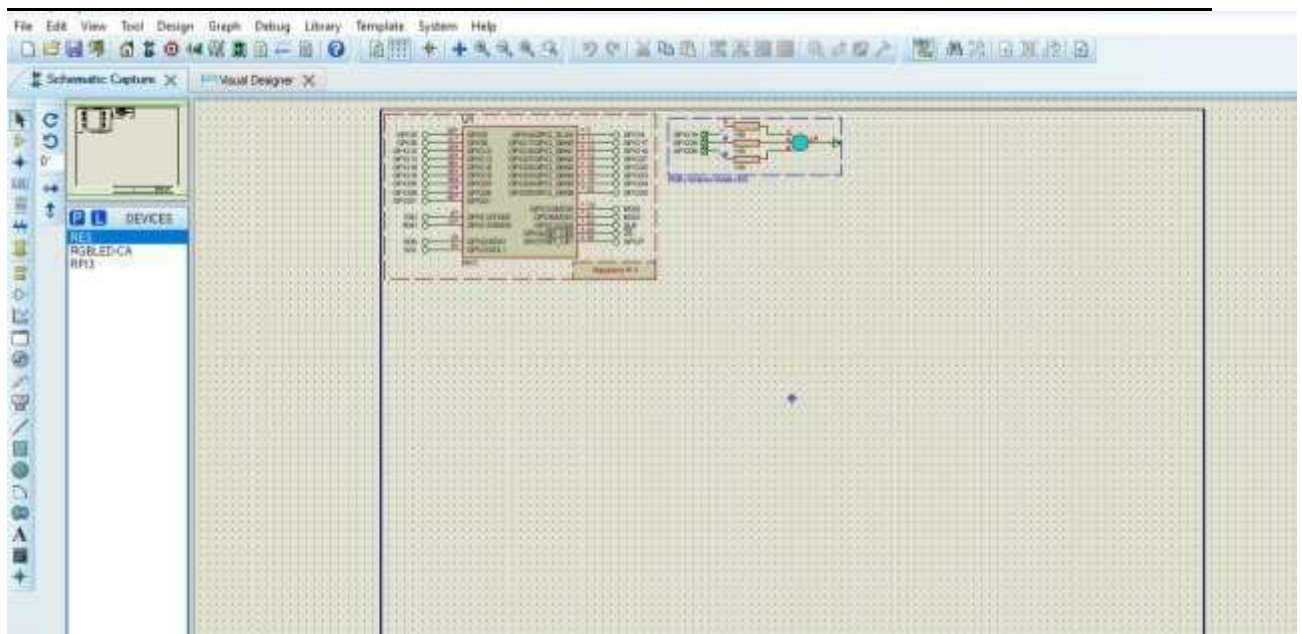
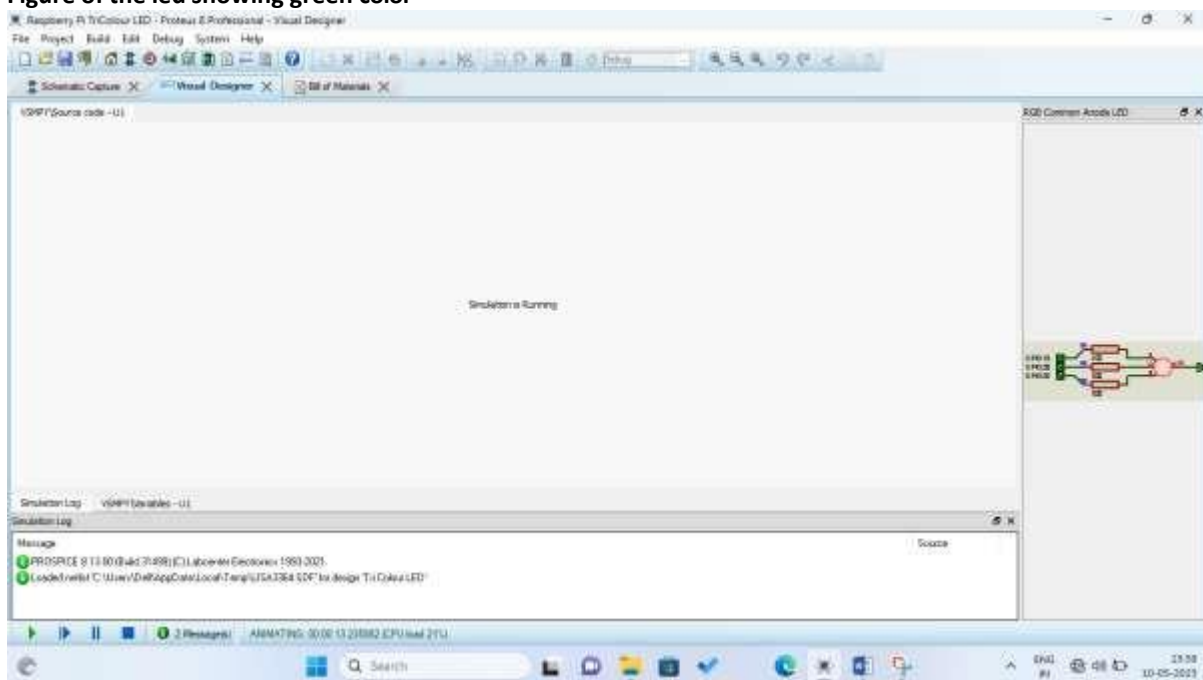


Figure of the led showing green color



Conclusion:-

Hence we have programmed the rgb strip led for the observation of various colors used to identify the paths.

Practical 8

Aim: Add pan and tilt service to the robot object and test it

Source code :

```
#include <Servo.h>
```

```
#include <Turtle.h>
```

```
#pragma GCC pop_options
```

```
// Peripheral Constructors
```

```
CPU &cpu = Cpu;
```

```
TimerOne &timer1 = Timer1;
```

```
TurtleDrive T1_DRIVE = TurtleDrive(2, 4, 3, 6, 7, 5);
```

```
TurtleSonarHead T1_SH = TurtleSonarHead(8, 9, 10);
```

```
TurtleLineHunter T1_LH = TurtleLineHunter(11, 12, A0);
```

```
Servo panServo;
```

```
Servo tiltServo;
```

```
void peripheral_setup() {
```

```
    T1_DRIVE.begin();
```

```
    T1_SH.begin();
```

```
    T1_LH.begin();
```

```
    panServo.attach(/*pan servo pin*/);
```

```
    tiltServo.attach(/*tilt servo pin*/);
```

```
}
```

```
void peripheral_loop() {
```

```
// Add any necessary servo control logic here  
}
```

```
//---CONFIG_END---
```

```
// Flowchart Variables
```

```
long var_speed;
```

```
long var_dir;
```

```
long var_count;
```

```
long var_range;
```

```
long var_fast;
```

```
long var_slow;
```

```
long var_tstart;
```

```
long var_tstop;
```

```
// Flowchart Routines
```

```
void chart_SETUP() {
```

```
    var_speed = 180, var_range = 10, var_dir = 0;
```

```
    T1_SH.setAngle(0);
```

```
    T1_SH.setRange(25);
```

```
    T1_DRIVE.forwards(var_speed);
```

```
}
```

```
void chart_LOOP() {
```

```
    if (!(T1_LH(0, 0, 0))) {
```

```
        if (T1_LH(1, 1, 1)) {
```

```
            chart_Correct();
```

```
        } else {
```

```
            if (T1_LH(0, 1, 1)) {
```



```

T1_DRIVE.drive(1, 1, 5 * var_speed / 4);
T1_DRIVE.drive(2, 1, var_speed / 2);
var_dir = 10;
chart_Avoid();
} else {
    if (T1_LH(0, 0, 1)) {
        T1_DRIVE.drive(1, 1, 5 * var_speed / 4);
        T1_DRIVE.drive(2, 0, var_speed / 5);
        var_dir = 30;
    } else {
        if (T1_LH(1, 1, 0)) {
            T1_DRIVE.drive(2, 1, 5 * var_speed / 4);
            T1_DRIVE.drive(1, 1, var_speed / 2);
            var_dir = -10;
            chart_Avoid();
        } else {
            if (T1_LH(1, 0, 0)) {
                T1_DRIVE.drive(2, 1, 5 * var_speed / 4);
                T1_DRIVE.drive(1, 0, var_speed / 5);
                var_dir = -30;
            } else {
                if (T1_LH(-1, 1, -1)) {
                    T1_DRIVE.forwards(var_speed);
                    var_dir = 0;
                    chart_Avoid();
                }
            }
        }
    }
}

```

```

    }
}
}
}
}

```

```

void chart_Correct() {
    var_count = 0;
l3;;
    if (var_dir > 0) {
        T1_DRIVE.drive(2, 1, var_speed);
        T1_DRIVE.drive(1, 0, var_speed / 3);
    } else {
        if (var_dir < 0) {
            T1_DRIVE.drive(1, 1, var_speed);
            T1_DRIVE.drive(2, 0, var_speed / 3);
        }
    }
    delay(1);
    var_count = var_count + 1;
    if (var_count < 1000) {
        if (T1_LH(1, 1, 1))
            goto l3;
    } else {
        T1_DRIVE.stop();
        var_dir = 0;
    }
}

```

```

void chart_Avoid() {
  if (T1_SH(var_range, 0)) {
    T1_DRIVE.backwards(2 * var_speed / 3);
    delay(250);
    T1_DRIVE.turn(80);
    do {
      delay(5);
    } while (!(T1_SH(1.5 * var_range, 0)) == false);
    T1_DRIVE.stop();
    delay(500);
    T1_DRIVE.turn(80);

    var_tstart = cpu.millis();
    while (!(T1_SH(1.5 * var_range, 0))) {
    }
    var_tstop = cpu.millis();
    var_count = (var_tstop - var_tstart) / 10;
    T1_DRIVE.stop();
    delay(500);
    T1_DRIVE.turn(-80);

    while (var_count > 0) {
      var_count = var_count - 1;
      delay(5);
    }
    T1_DRIVE.backwards(2 * var_speed / 3);
    delay(300);
  }
}

```

```
    T1_DRIVE.forwards(var_speed / 2);  
  }  
}
```

// Entry Points and Interrupt Handlers

```
void setup() {  
  peripheral_setup();  
  chart_SETUP();  
}
```

```
void loop() {  
  peripheral_loop();  
  chart_LOOP();  
}
```

Practical No :-9

Aim - Create an obstacle avoidance behaviour for robot and test it.

Components

Arduino

uno Zumo

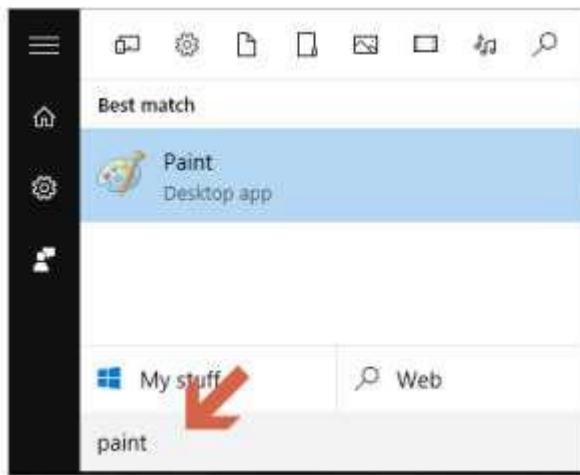
robot

Description of Zumo robot:-

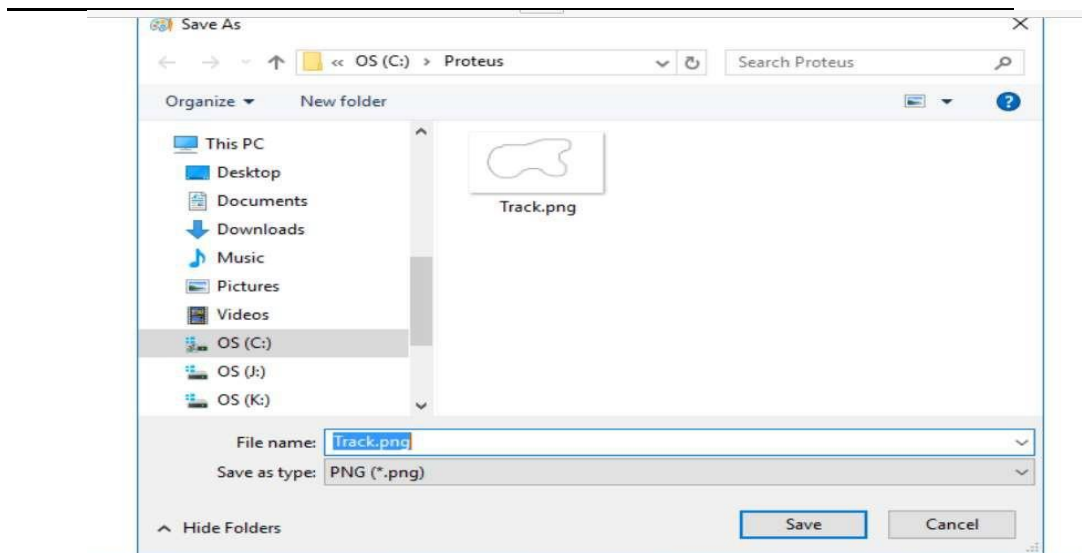
The Zumo robot for Arduino is an Arduino-controllable tracked robot platform. It includes two micro metal gearmotors coupled to a pair of silicone tracks, a stainless steel bulldozer-style blade, an array of six infrared reflectance sensors for line following or edge detection, a buzzer for simple sounds and music, a 3-axis accelerometer, magnetometer, and gyro for detecting impacts and tracking orientation. The Zumo is a more advanced turtle than the Funduino and can perform far better at line following and maze escape challenges but it does not have ultrasonic range finder and therefore is not as well suited to obstacle avoidance challenges.

To create and apply an environment for simulation

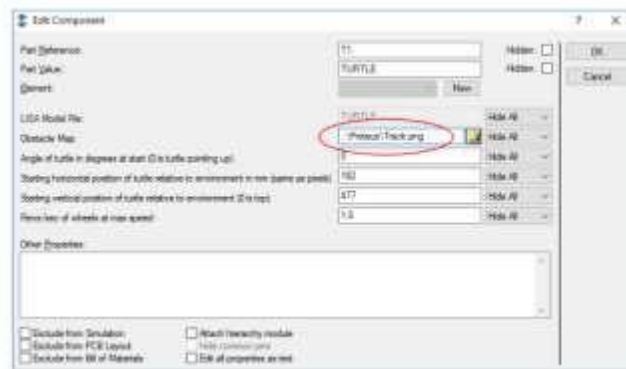
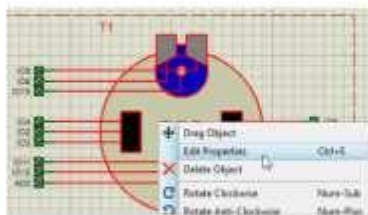
1) Draw the required route / obstacles in the graphics package of your choice. Remember that the scale is 1 pixel to 1 mm so drawing a 5 pixel wide line will equate to 5mm in the real world. Width of the line being followed can affect the algorithm (e.g. if all the sensors are never over the line) so it's important to give this some thought.



- 3) Save the graphic as a PNG file into the same directory as your Proteus project.



3) Edit the turtle component on your schematic and specify the graphic you have just saved as the obstacle map



Source Code:

```
//---CONFIG_BEGIN---
#pragma GCC push_options
#pragma GCC optimize("Os")

#include <core.h> // Required by
cpu #include <cpu.h>
#include <TimerOne.h>
#include <Servo.h>
#include <Turtle.h>

#pragma GCC pop_options

// Peripheral Constructors
CPU &cpu = Cpu;
TimerOne &timer1 = Timer1;
TurtleDrive T1_DRIVE = TurtleDrive(2, 4, 3, 6, 7, 5);
TurtleSonarHead T1_SH = TurtleSonarHead(8, 9, 10);
TurtleLineHunter T1_LH = TurtleLineHunter(11, 12, A0);

void peripheral_setup() {
```

```

T1_DRIVE.begin();
T1_SH.begin();
T1_LH.begin();
}

void peripheral_loop() {

//---CONFIG_END---
// Flowchart Variables
long var_speed;
long var_dir;
long var_count;
long var_range;
long var_fast;
long var_slow;
long var_tstart;
long var_tstop;

// Flowchart Routines
void chart_SETUP() {
  var_speed = 180, var_range = 10, var_dir = 0;
  T1_SH.setAngle(0);
  T1_SH.setRange(25);
  T1_DRIVE.forwards(var_speed);
}

void chart_LOOP() {
  if (!(T1_LH(0, 0, 0))) {
    if (T1_LH(1, 1, 1)) {
      chart_Correct();
    } else {
      if (T1_LH(0, 1, 1)) {
        T1_DRIVE.drive(1, 1, 5 * var_speed / 4);
        T1_DRIVE.drive(2, 1, var_speed / 2);
        var_dir = 10;
        chart_Avoid();
      } else {
        if (T1_LH(0, 0, 1)) {
          T1_DRIVE.drive(1, 1, 5 * var_speed / 4);
          T1_DRIVE.drive(2, 0, var_speed / 5);
          var_dir = 30;
        } else {
          if (T1_LH(1, 1, 0)) {
            T1_DRIVE.drive(2, 1, 5 * var_speed / 4);
            T1_DRIVE.drive(1, 1, var_speed / 2);
            var_dir = -10;
            chart_Avoid();
          } else {
            if (T1_LH(1, 0, 0)) {
              T1_DRIVE.drive(2, 1, 5 * var_speed / 4);
              T1_DRIVE.drive(1, 0, var_speed / 5);
              var_dir = -30;
            } else {
              if (T1_LH(-1, 1, -1)) {
                T1_DRIVE.forwards(var_speed);
                var_dir = 0;
                chart_Avoid();
              }
            }
          }
        }
      }
    }
  }
}

```

```

    }
    }
    }
    }
}

void chart_Correct() {
    var_count = 0;
l3::
    if (var_dir > 0) {
        T1_DRIVE.drive(2, 1, var_speed);
        T1_DRIVE.drive(1, 0, var_speed / 3);
    } else {
        if (var_dir < 0) {
            T1_DRIVE.drive(1, 1, var_speed);
            T1_DRIVE.drive(2, 0, var_speed / 3);
        }
    }
    delay(1);
    var_count = var_count + 1;
    if (var_count < 1000) {
        if (T1_LH(1, 1, 1))
            goto l3;
    } else {
        T1_DRIVE.stop();
        var_dir = 0;
    }
}

void chart_Avoid() {
    if (T1_SH(var_range, 0)) {
        T1_DRIVE.backwards(2 * var_speed / 3);
        delay(250);
        T1_DRIVE.turn(80);
        do {
            delay(5);
        } while ((!(T1_SH(1.5 * var_range, 0))) == false);
        T1_DRIVE.stop();
        delay(500);
        T1_DRIVE.turn(80);

        var_tstart = cpu.millis();
        while (!(T1_SH(1.5 * var_range, 0))) {
        }
        var_tstop = cpu.millis();
        var_count = (var_tstop - var_tstart) / 10;
        T1_DRIVE.stop();
        delay(500);
        T1_DRIVE.turn(-80);

        while (var_count > 0) {
            var_count = var_count - 1;
            delay(5);
        }
        T1_DRIVE.backwards(2 * var_speed / 3);
        delay(300);
        T1_DRIVE.forwards(var_speed / 2);
    }
}

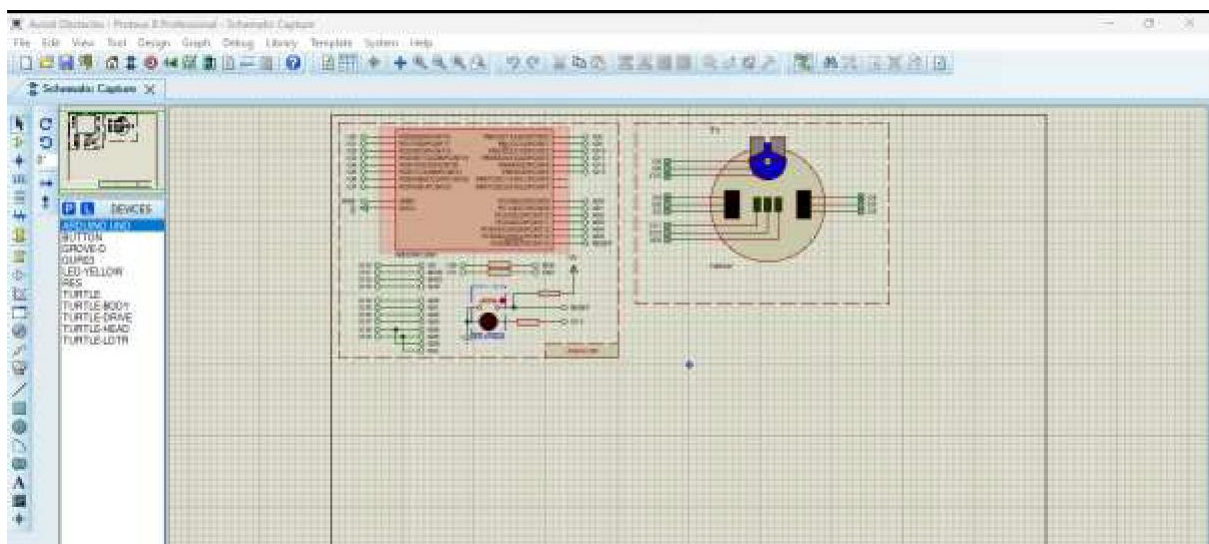
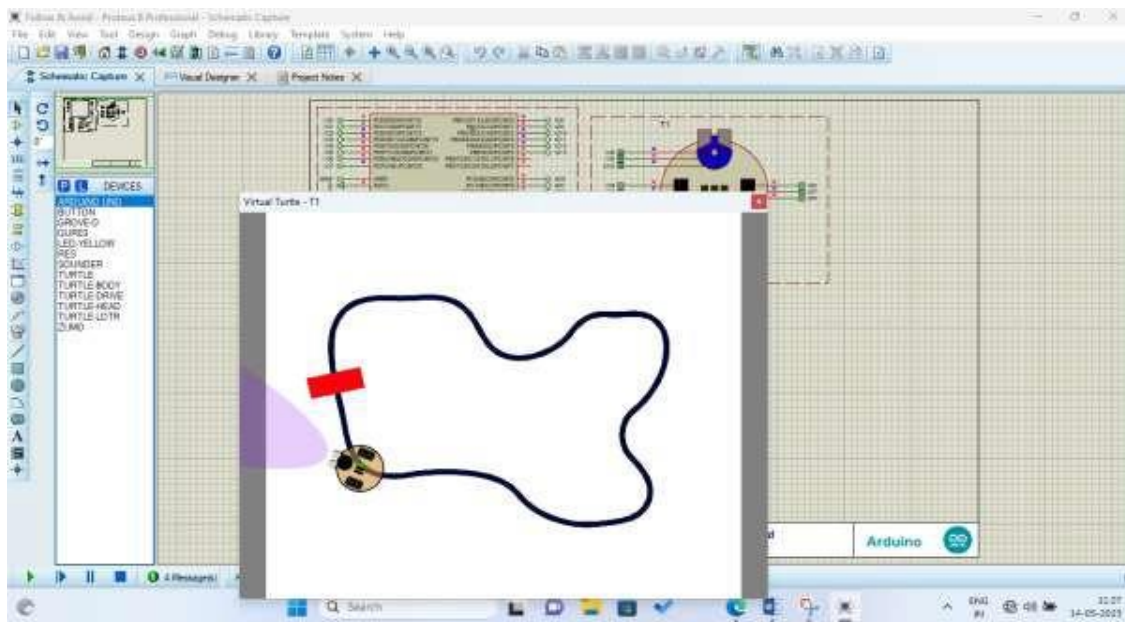
```


// Entry Points and Interrupt Handlers

```
void setup() {  
  peripheral_setup();  
  chart_SETUP();  
}
```

```
void loop() {  
  peripheral_loop();  
  chart_LOOP();  
}
```

Output for obstacle avoidance using zumorobot:--



Conclusion

Hence the obstacle is avoided using the Zumo robot

Pract 10:

Aim :-Detect faces with haar cascades

pip install opencv_python

Coding:

```
import cv2
#reading the
image
img = cv2.imread("face.jpg")

#converting image to grayscale
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

#Loading the required haar-cascade.xml classifier file
haar_cascade =
cv2.CascadeClassifier(cv2.data.harcascades +
"haarcascade_frontalface_default.xml")

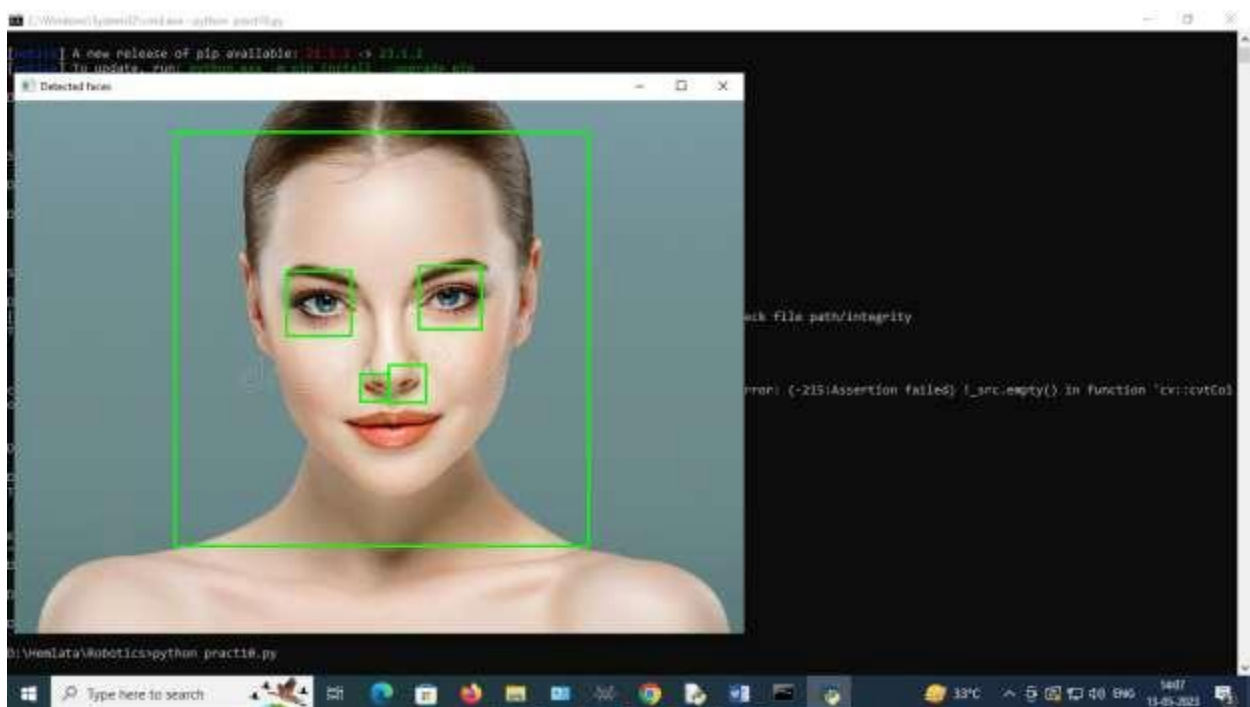
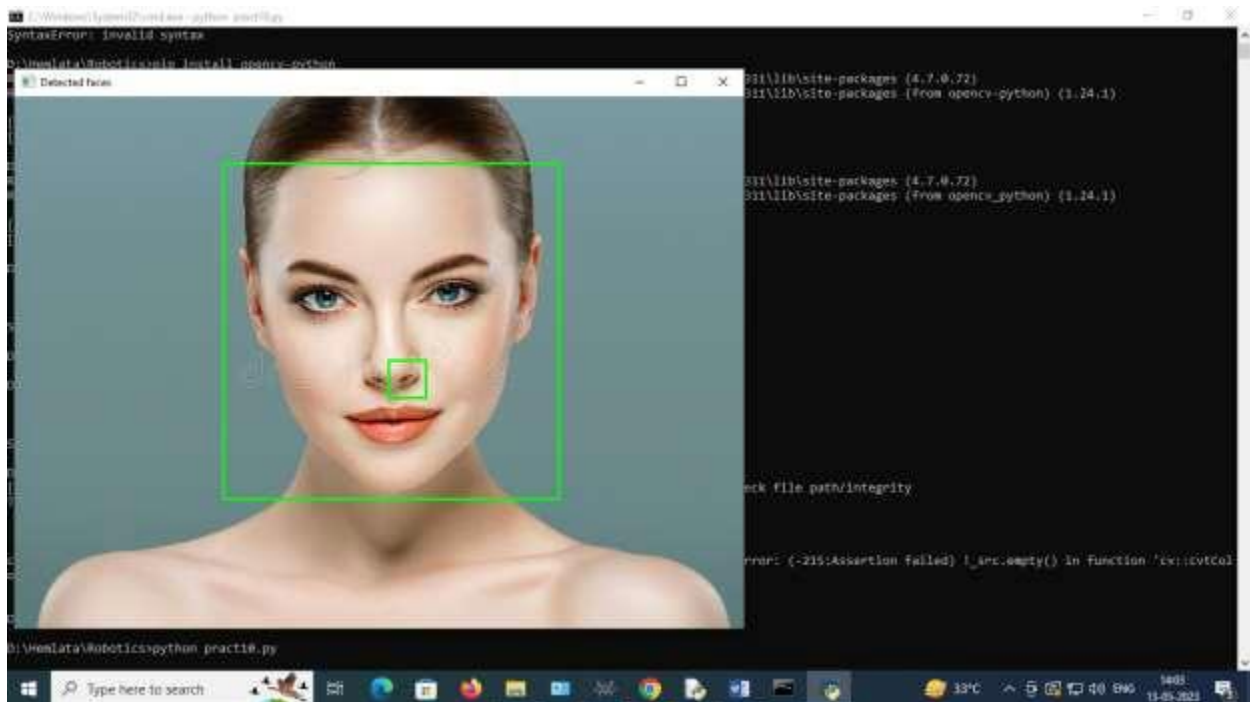
eye_cascade = cv2.CascadeClassifier(cv2.data.harcascades +
"haarcascade_eye.xml")

faces_rect = haar_cascade.detectMultiScale(gray_img, 1.3,

5) eyes = eye_cascade.detectMultiScale(gray_img)

for(x, y, w ,h) in faces_rect:
    cv2.rectangle(img, (x,y), (x+w, y+h), (0,255,0), 2)
    for(ex, ey, ew ,eh) in eyes:
        cv2.rectangle(img, (ex,ey), (ex+ew, ey+eh), (0,255,0), 2)
    cv2.imshow('Detected faces', img)
    cv2.waitKey(0)
```

OUTPUT:



OUTPUT:-Hence the face detection is done using haar cascades in python

Practical 11

Aim :Using the robot to display its camera as a web app on a phone or desktop and then use camera to drive smart Color and face tracking behaviours

Components needed:-

Proteus Simulator 8.9 and

above Lcd TFT

Button

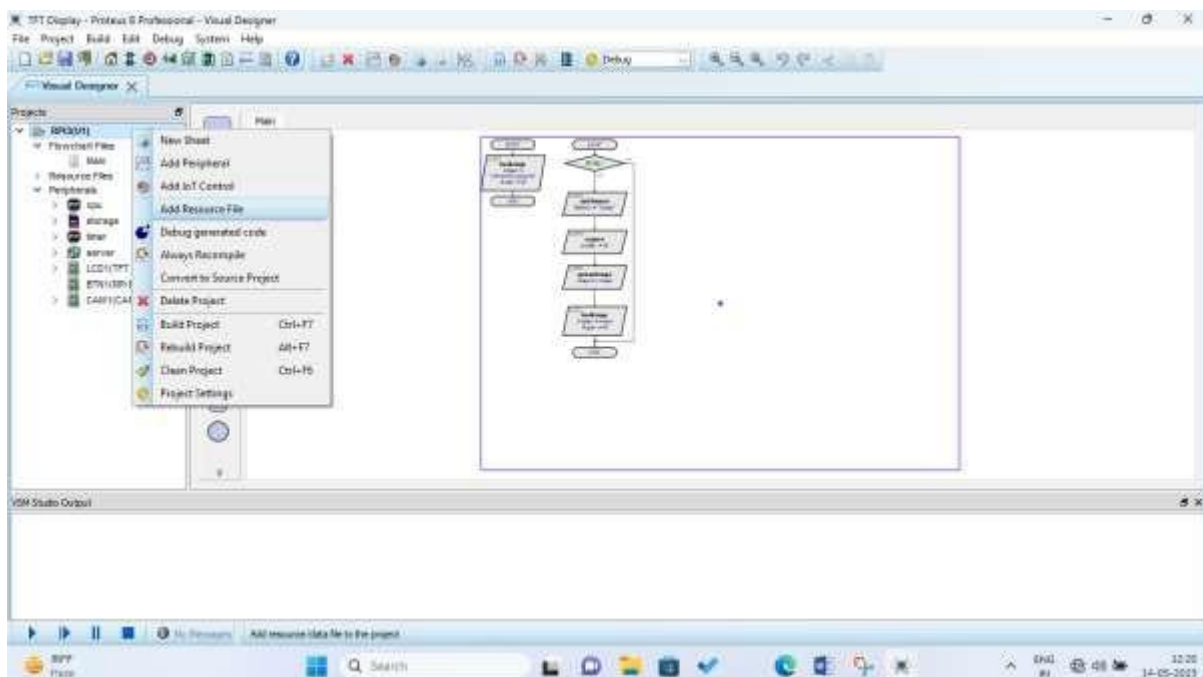
Raspberry pi camera

Procedure:-

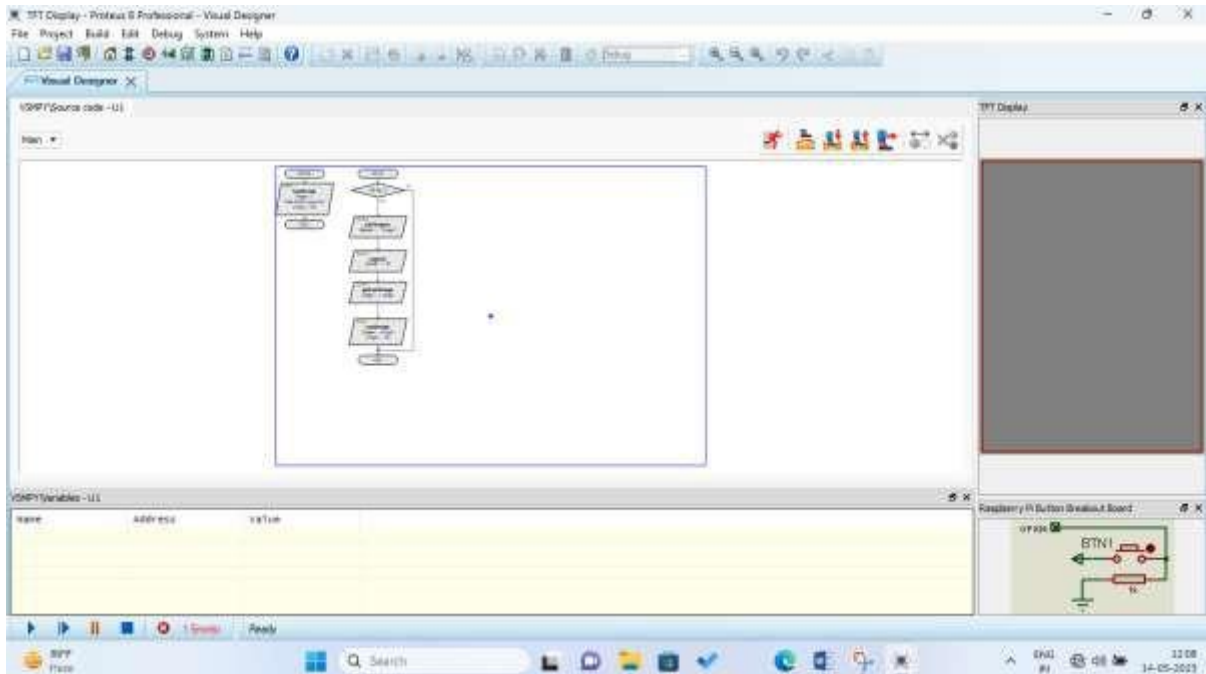
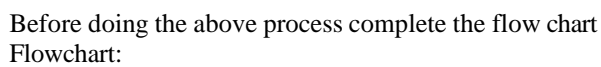
First the components from the library and place it in the schematic chart Steo 1:-Select new project from the tab , select flow chart project

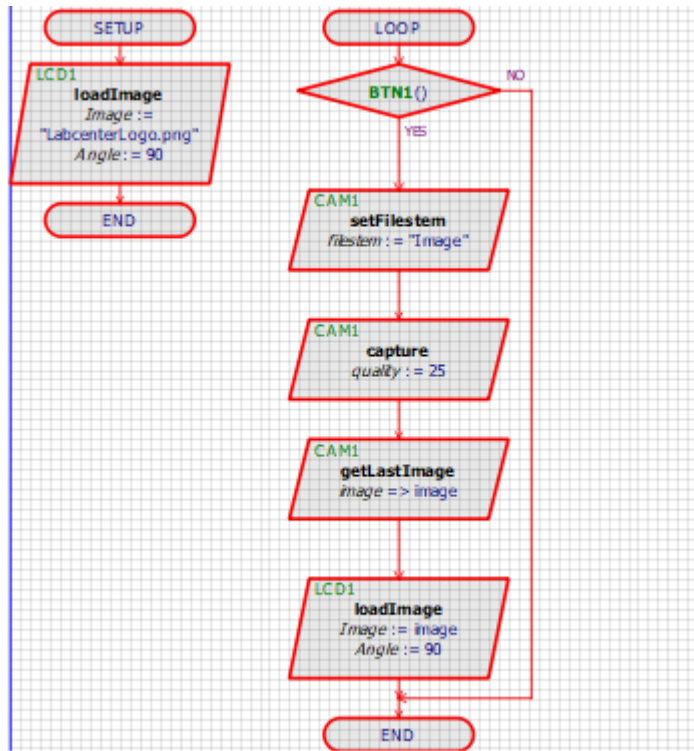
Step 2:- Create new folder and save the project by remaining as camera.psdrrpj in proteus.

Step 3:- Select a picture and place it in the resource file by right clicking on raspberry pi in the flowchart view



Save it and then run the simulation once you run the simulation you will see the output as shown below





Step7: Once you click the button in the schematic view we can see our image on the TFT LCD selected. For displaying your image please keep the camera of the laptop on incase desktop use the web camera

