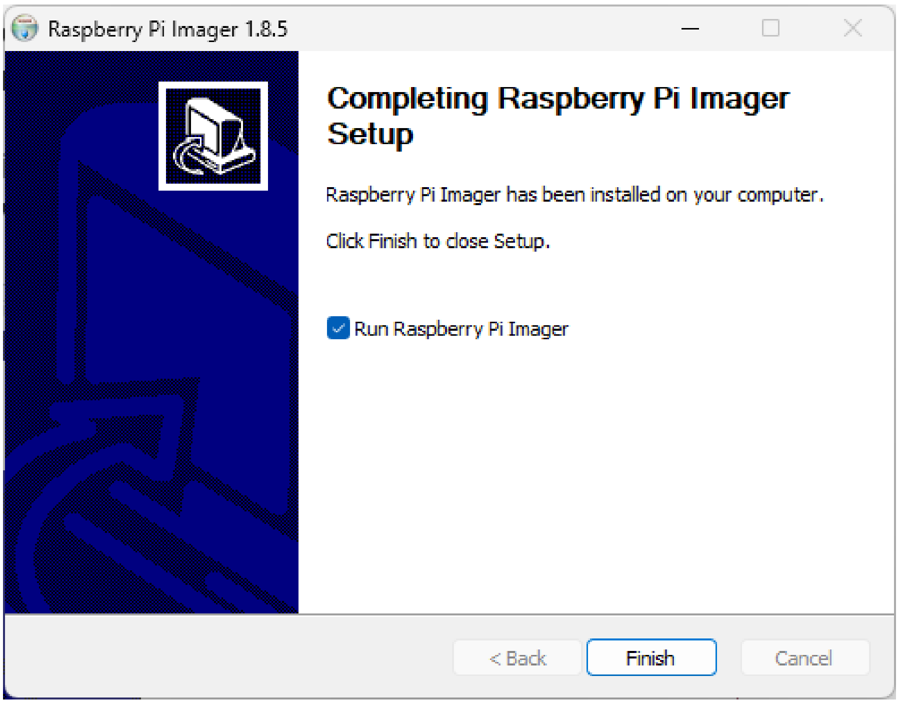
Practical 1

Aim: - Making a Raspberry Pi headless, and reaching it from the network using WiFi and SSH.

Prerequisite:

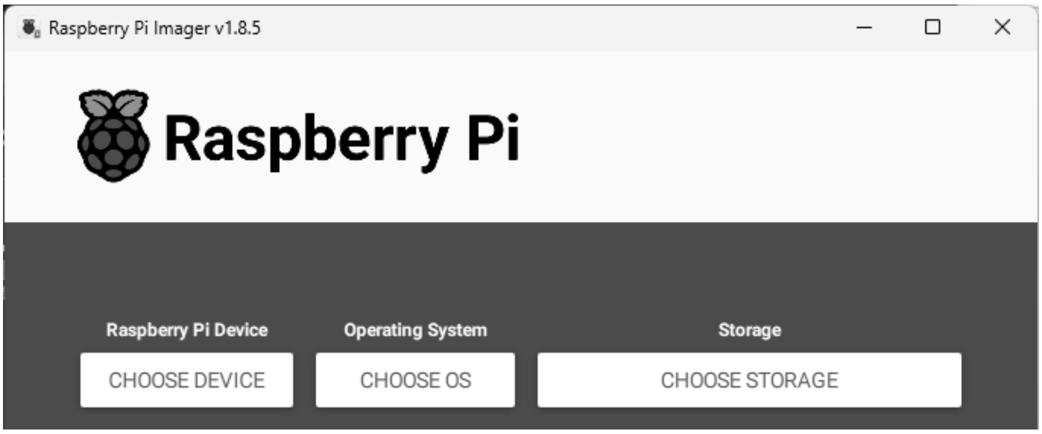
- Raspberry pi imager download [here]

Step 1: Install Raspberry pi imager

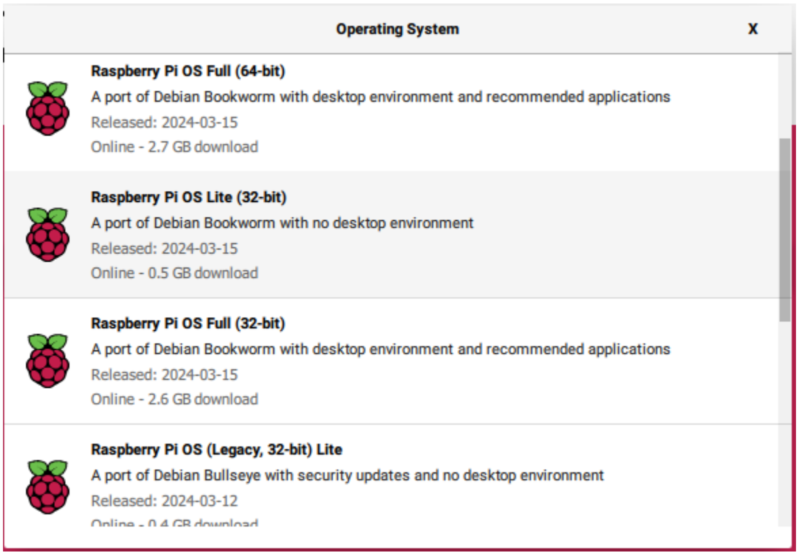


Step 2: Create a bootable SD card

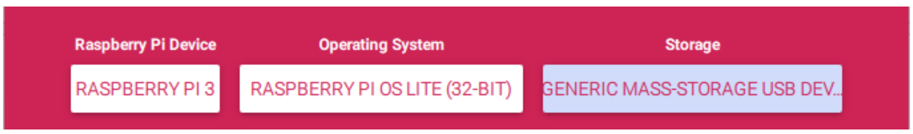
Open Raspberry Pi Imager



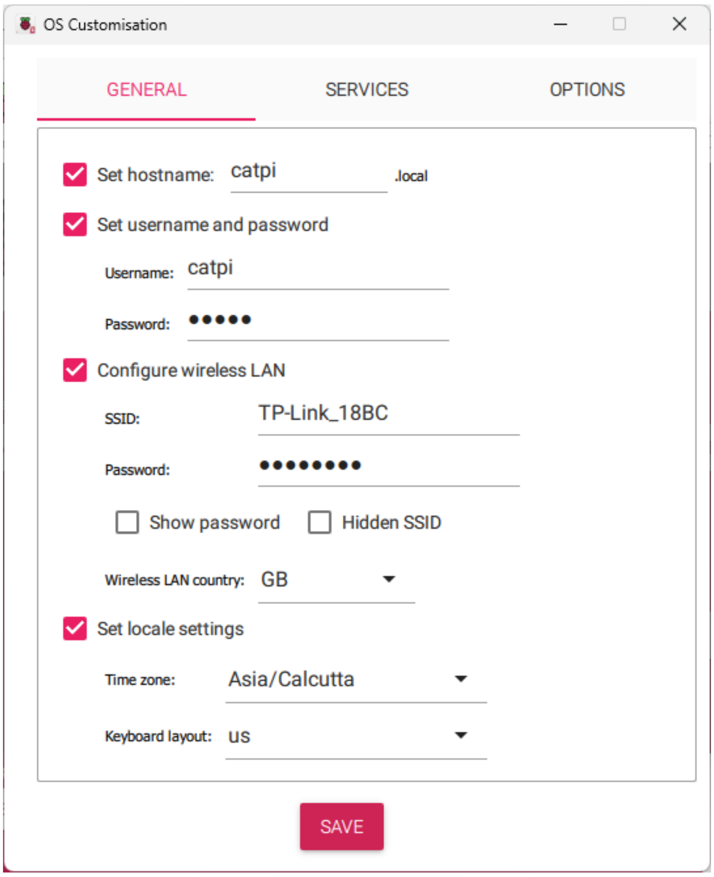
Under **Others** choose **Raspberry Pi OS Lite (32 bit)**



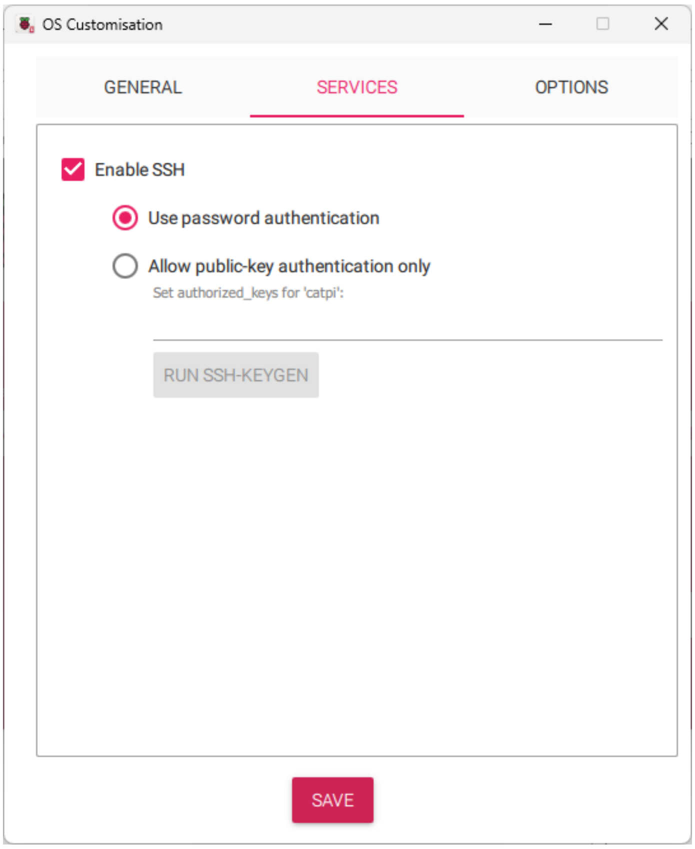
Choose the appropriate **device** and the **storage**



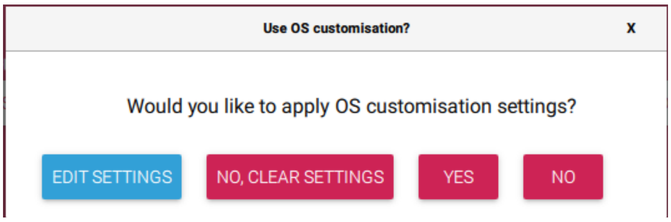
After Clicking **Next** Click on **Edit Settings**
Enter basic Information for wifi and login credentials

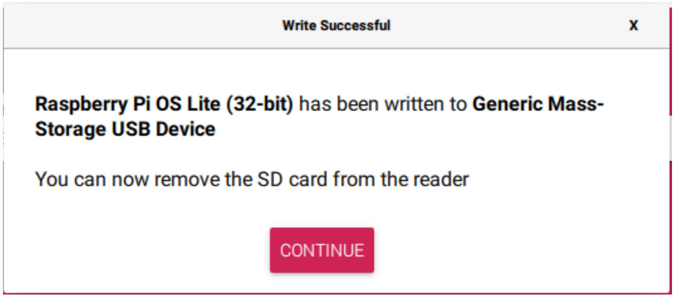


Enable SSH Service



Click Yes and wait for it to finish



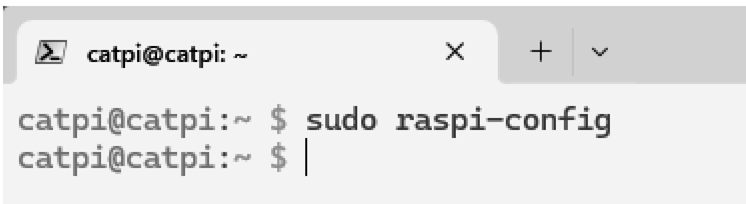


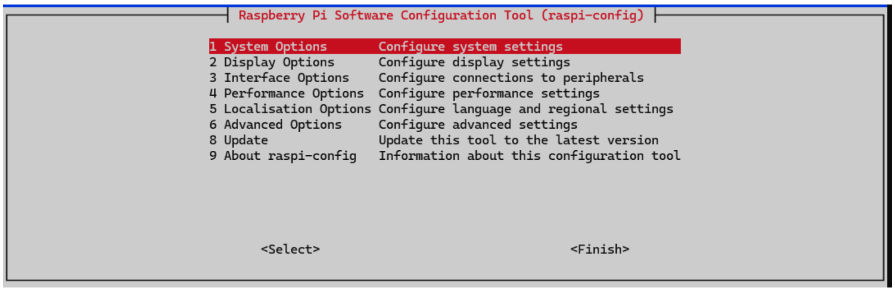
Step 3: Connect Raspberry Pi using ssh
(Note: I am using powershell 7 that has builtin ssh)

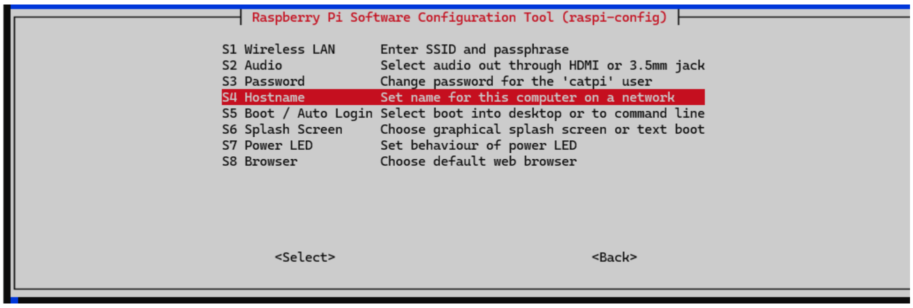
```
ssh <username>@<ip_of_rasp>
```

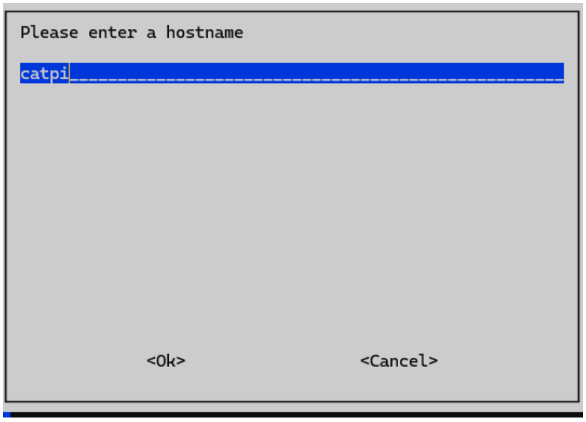
```
PS D:\> ssh catpi@10.128.0.130
catpi@10.128.0.130's password:
Linux catpi 6.6.28+rpt-rpi-v7 #1 SMP Raspbian 1:6.6.28-1+rpt1 (2024-04-22) armv7l

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 May  7 16:29:57 2024 from 10.128.0.119
catpi@catpi:~ $
```

Step 4: Change hostname of the system









Step 5: Update the system

```
sudo apt update && sudo apt upgrade
```

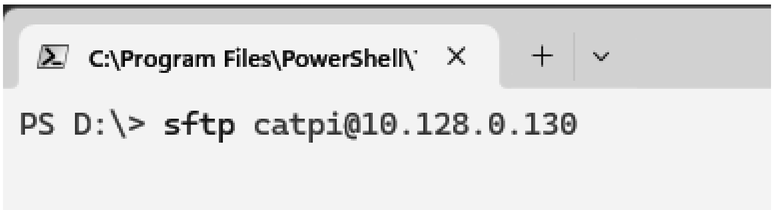
```
catpi@catpi:~ $ sudo apt update && sudo apt upgrade -y
Get:1 http://archive.raspberrypi.com/debian bookworm InRelease [23.6 kB]
Get:2 http://raspbian.raspberrypi.com/raspbian bookworm InRelease [15.0 kB]
Get:3 http://archive.raspberrypi.com/debian bookworm/main arm64 Packages [387 kB]
Get:4 http://archive.raspberrypi.com/debian bookworm/main armhf Packages [397 kB]
Get:5 http://raspbian.raspberrypi.com/raspbian bookworm/main armhf Packages [14.5 MB]
Ign:5 http://raspbian.raspberrypi.com/raspbian bookworm/main armhf Packages
Get:5 http://raspbian.raspberrypi.com/raspbian bookworm/main armhf Packages [14.5 MB]
Fetched 13.2 MB in 1min 20s (165 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
40 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: http://raspbian.raspberrypi.com/raspbian/dists/bookworm/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/
apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following NEW packages will be installed:
  linux-headers-6.6.28+rpt-common-rpi linux-headers-6.6.28+rpt-rpi-v6 linux-headers-6.6.28+rpt-rpi-v7
  linux-headers-6.6.28+rpt-rpi-v7l linux-image-6.6.28+rpt-rpi-v6 linux-image-6.6.28+rpt-rpi-v7
  linux-image-6.6.28+rpt-rpi-v7l linux-kbuild-6.6.28+rpt pastebinit python3-pycryptodome
The following packages have been kept back:
  linux-image-rpi-v8:arm64
The following packages will be upgraded:
```

Practical 2

Aim: - Using sftp upload files from PC.
(Note: I am using powershell 7 that has builtin ssh)

Step 1: Run sftp

```
sftp <username>@<rasp_ip>
```



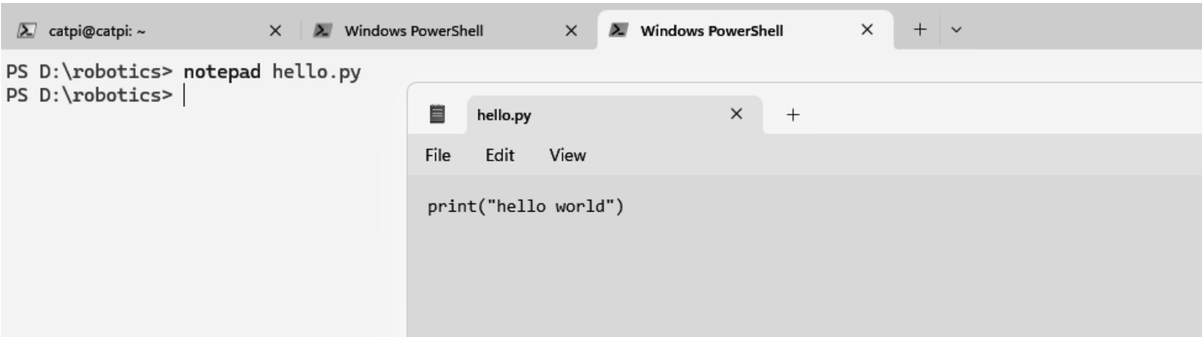
Help menu

```
sftp> help
Available commands:
bye                                Quit sftp
cd path                            Change remote directory to 'path'
chgrp [-h] grp path                Change group of file 'path' to 'grp'
chmod [-h] mode path               Change permissions of file 'path' to 'mode'
chown [-h] own path                Change owner of file 'path' to 'own'
df [-hi] [path]                    Display statistics for current directory or
                                   filesystem containing 'path'
exit                               Quit sftp
get [-afpR] remote [local]         Download file
help                               Display this help text
lcd path                           Change local directory to 'path'
lls [ls-options [path]]            Display local directory listing
lmkdir path                        Create local directory
ln [-s] oldpath newpath            Link remote file (-s for symlink)
lpwd                               Print local working directory
ls [-1afhlnrSt] [path]             Display remote directory listing
lumask umask                       Set local umask to 'umask'
mkdir path                         Create remote directory
progress                           Toggle display of progress meter
put [-afpR] local [remote]         Upload file
pwd                                Display remote working directory
quit                               Quit sftp
reget [-fpR] remote [local]        Resume download file
rename oldpath newpath             Rename remote file
reput [-fpR] local [remote]        Resume upload file
rm path                            Delete remote file
rmdir path                         Remove remote directory
symlink oldpath newpath            Symlink remote file
version                            Show SFTP version
!command                           Execute 'command' in local shell
!                                  Escape to local shell
?                                  Synonym for help
sftp>
```

Check the current directory of raspberry pi we are in

```
sftp> pwd
Remote working directory: /home/catpi
sftp>
```

Step 2: Create a python file on main system



Step 3: Upload the created file on Raspberry Pi

```
sftp> put D:\robotics\hello.py
Uploading D:/robotics/hello.py to /home/catpi/hello.py
D:/robotics/hello.py
sftp>
```

Step 4: Check the uploaded file on Rasberry Pi

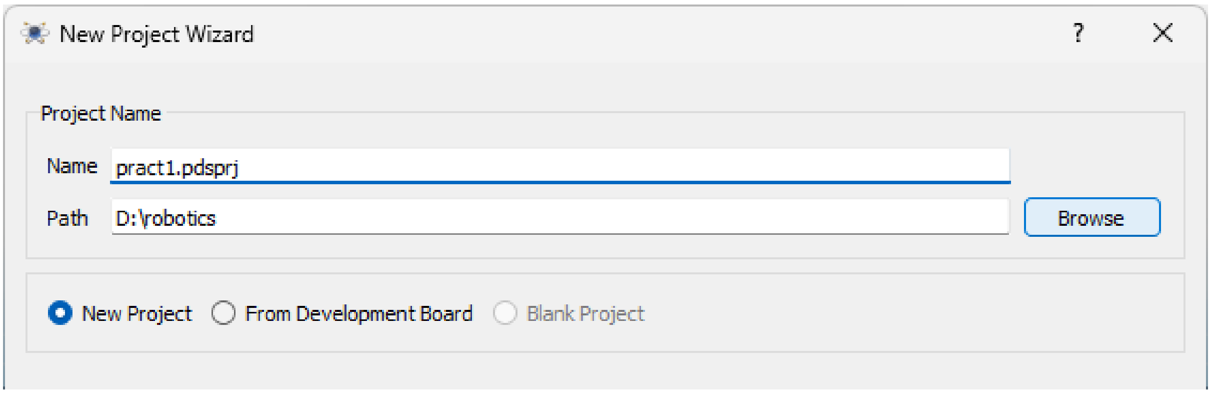
```
catpi@catpi:~ $ ls
hello.py
catpi@catpi:~ $ python hello.py
hello world
catpi@catpi:~ $
```

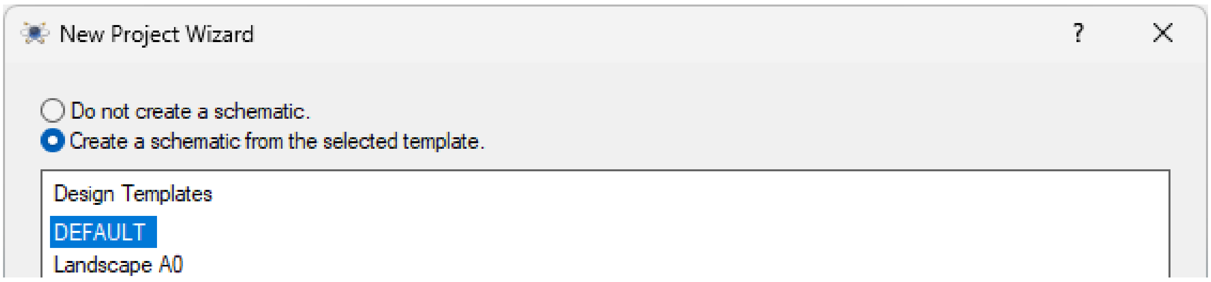
```
catpi@catpi:~ $ cat hello.py
print("hello world")catpi@catpi:~ $
```

Practical 3

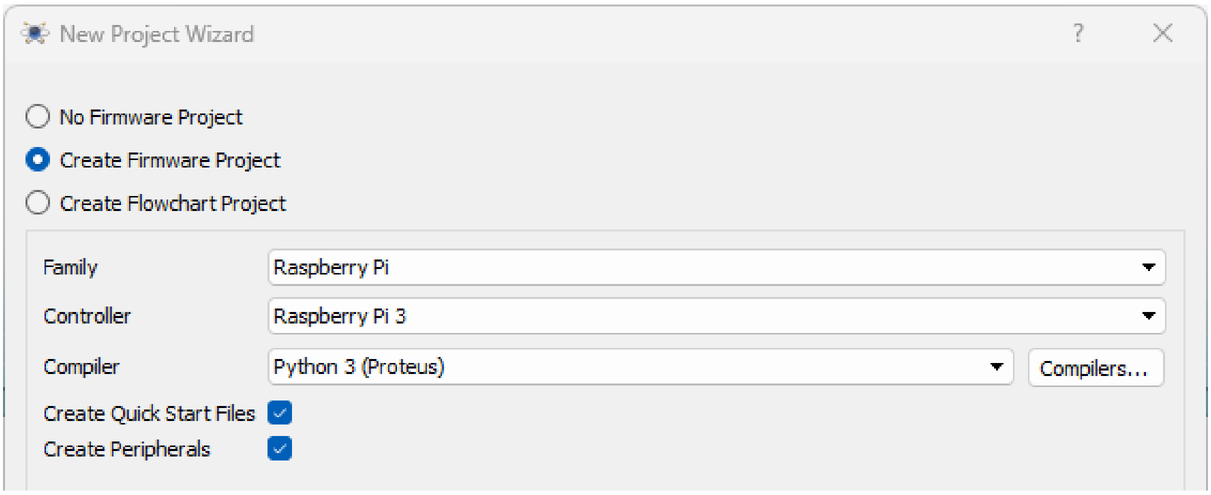
Aim: - Write Python code to test motors.

Step 1:Create a new proteus project



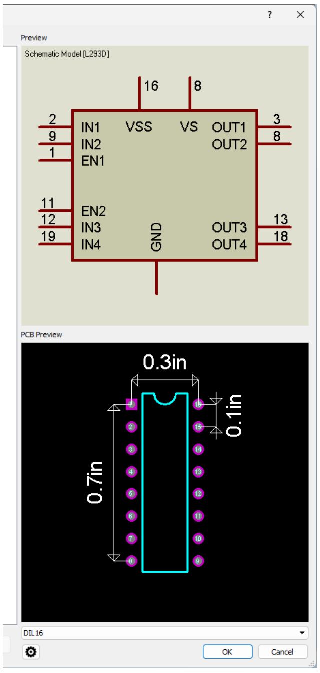


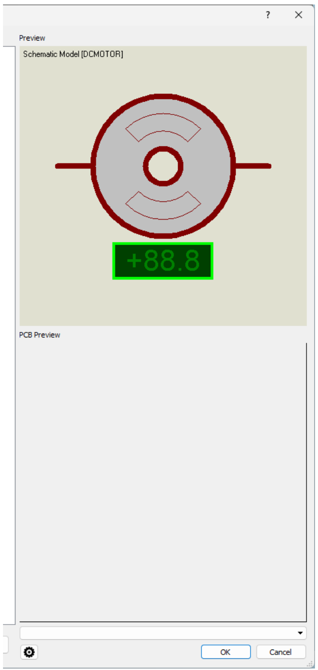
Step 2: Select Raspberry pi as option under Create Firmware Project



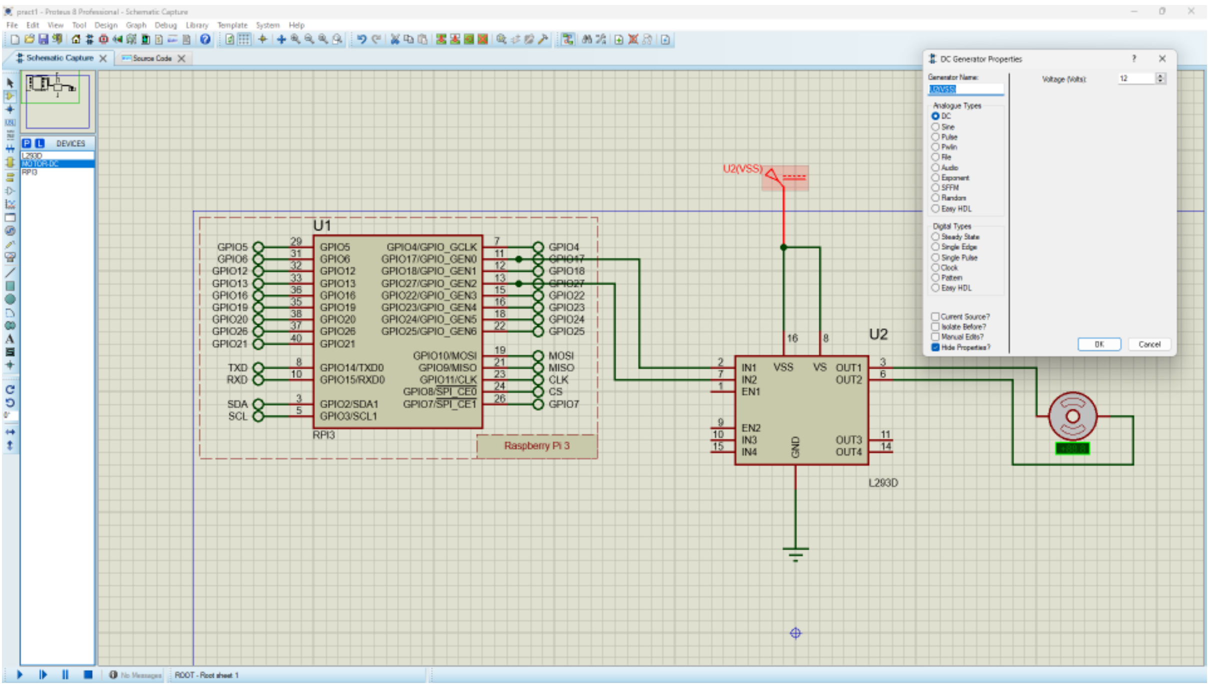
Step 3: Add the components necessary to test the motor.

- **DC (Generators) (Set to 12 Volts)**
- **MOTOR-DC (Devices)**
- **L293D (Devices)**
- **GROUND (Terminals)**



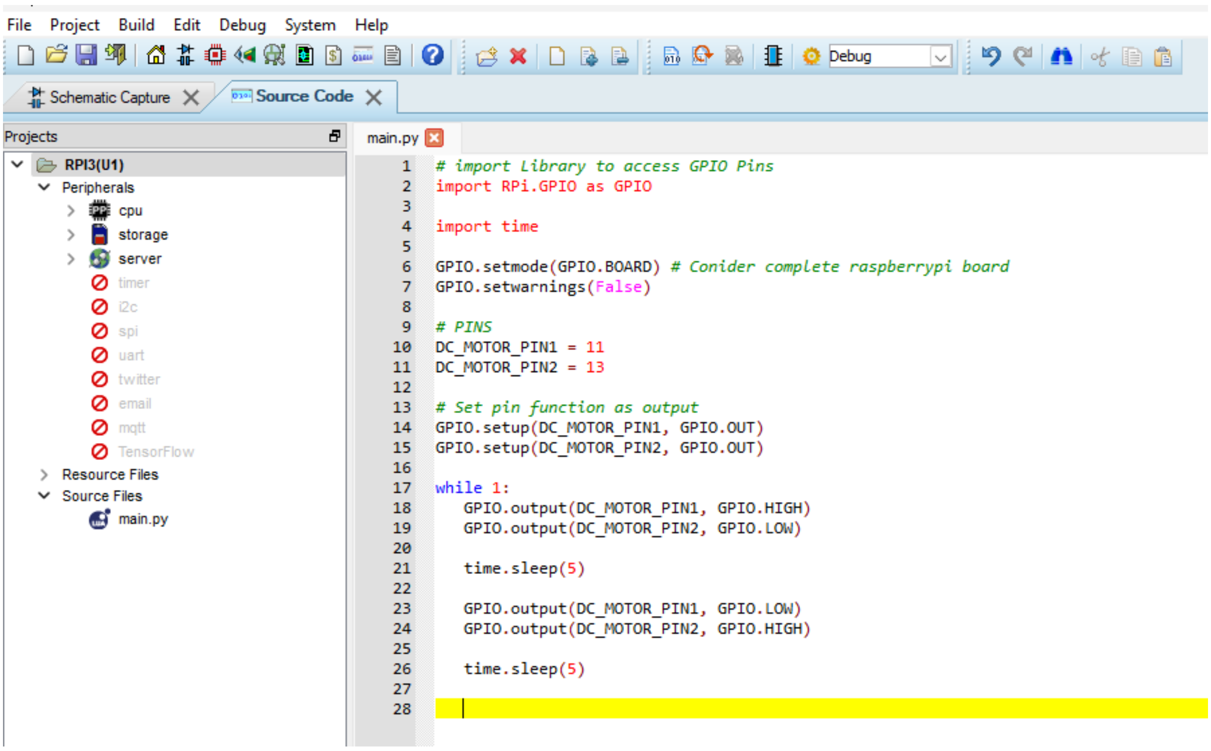


Step 4: Complete the connections



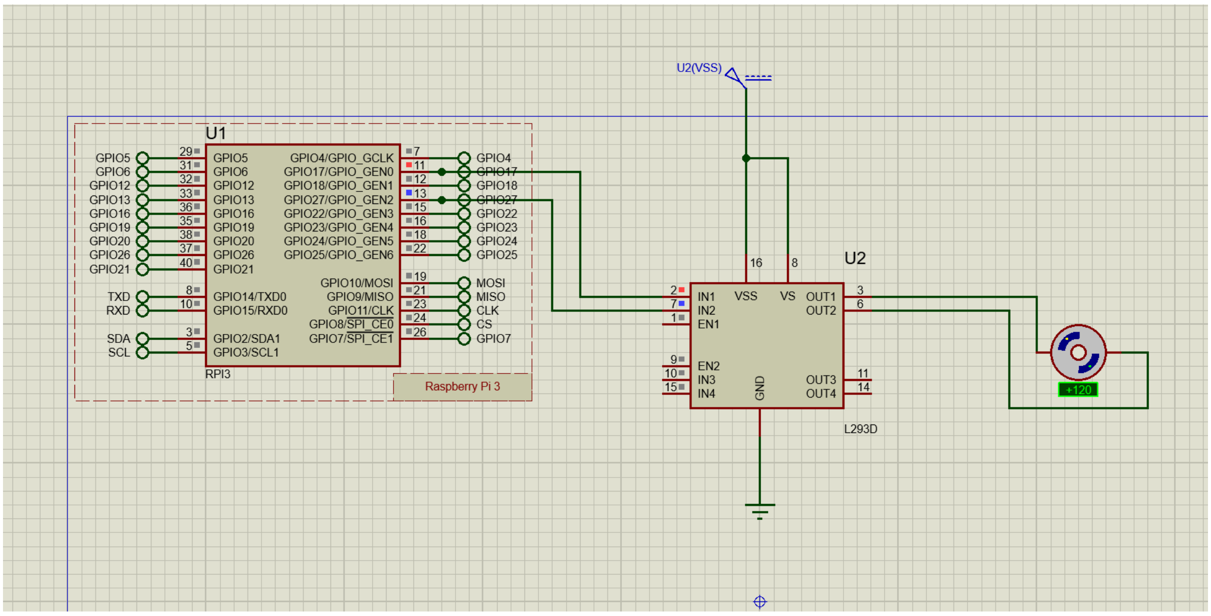
Step 5: Edit the source code with appropriate pins for motor control

```python
# import Library to access GPIO Pins
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD) # Conider complete raspberrypi board
GPIO.setwarnings(False)
# PINS
DC_MOTOR_PIN1 = 11
DC_MOTOR_PIN2 = 13
# Set pin function as output
GPIO.setup(DC_MOTOR_PIN1, GPIO.OUT)
GPIO.setup(DC_MOTOR_PIN2, GPIO.OUT)

while 1:
   GPIO.output(DC_MOTOR_PIN1, GPIO.HIGH)
   GPIO.output(DC_MOTOR_PIN2, GPIO.LOW)
   time.sleep(5)
   GPIO.output(DC_MOTOR_PIN1, GPIO.LOW)
   GPIO.output(DC_MOTOR_PIN2, GPIO.HIGH)
   time.sleep(5)
```

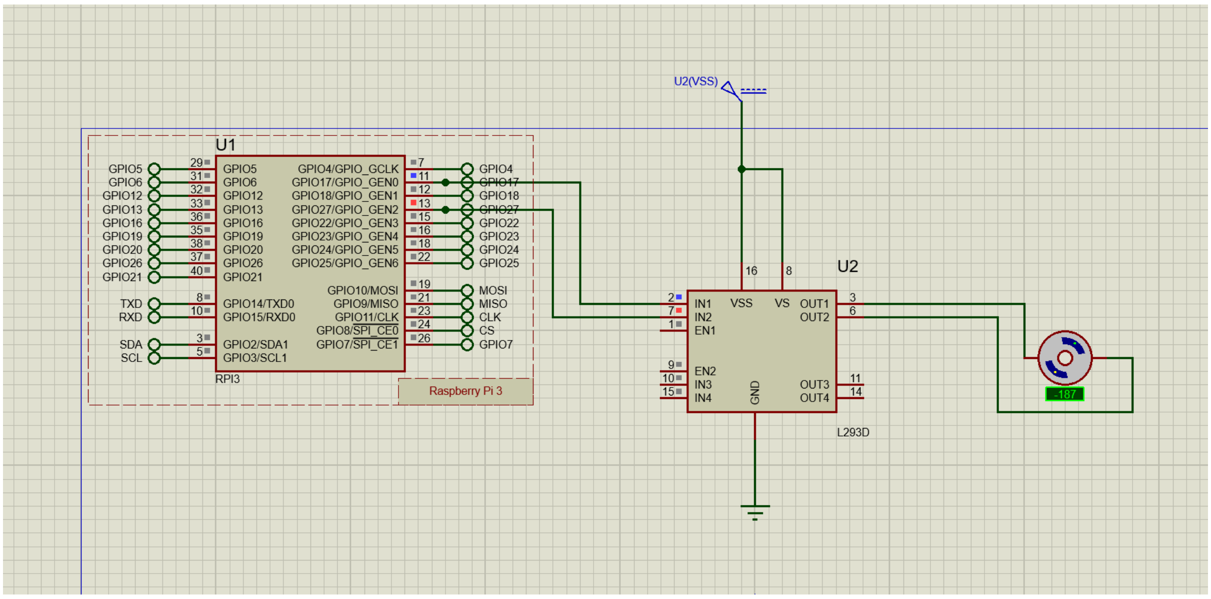


Step 6: Run the simulator. The motor should run clockwise and anticlockwise at 5 second intervals

Clockwise



AntiClockwise



Practical 4

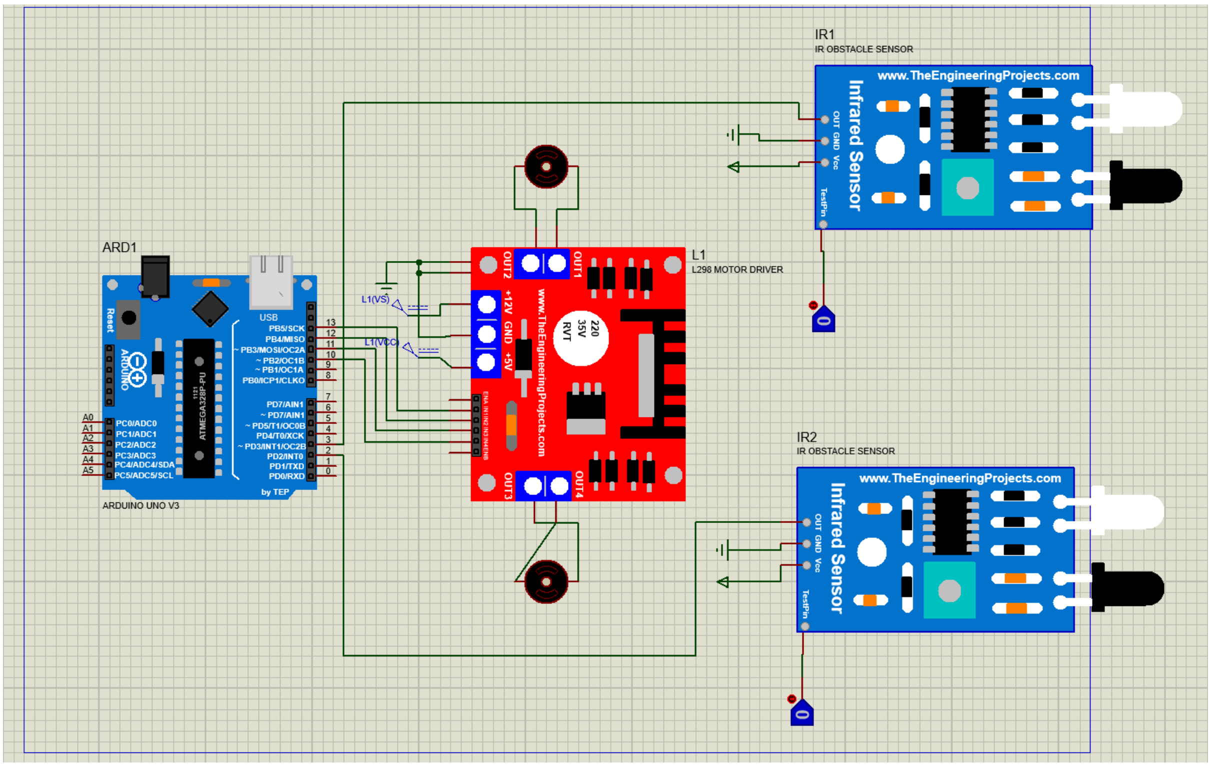
Aim: - Add the sensors to the Robot object and develop the line following behaviour code.

(Note: Add the provided libraries following this guide)

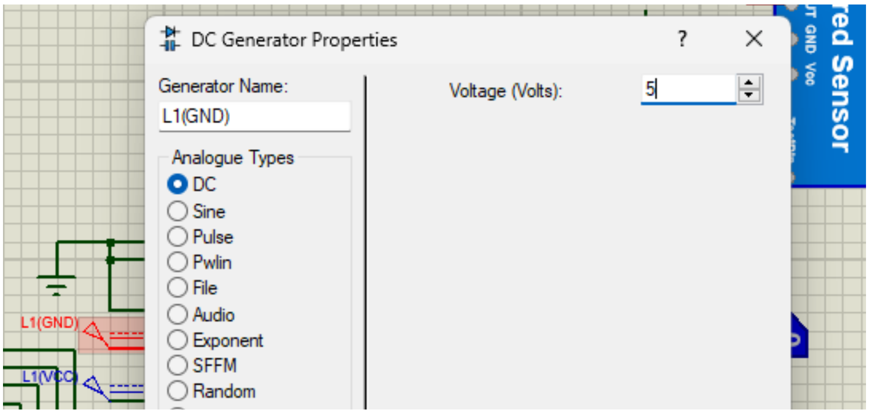
Components

- **L298 Motor Driver**
- **2 IR Obstacle Sensor**
- **Arduino UNO 3**
- **2 Logic Toggle**
- **2 Motor**
- **3 Ground (Terminals)**
- **2 Power (Terminals)**
- **2 DC (Generators)**

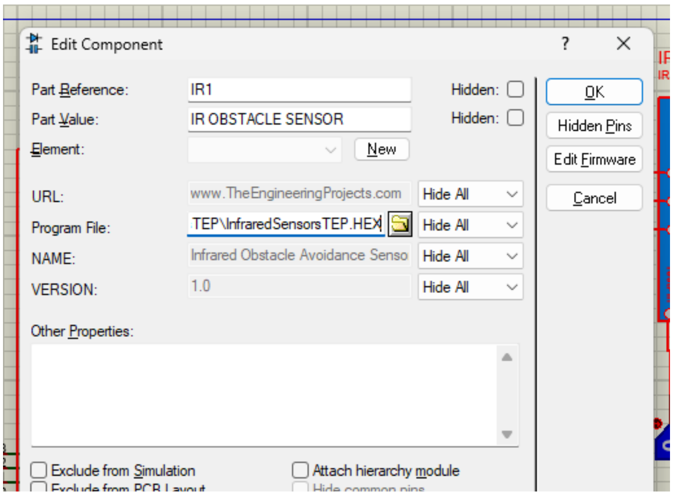
Step 1: Create the following circuit



Step 2: Set both DC Generators voltage to 5



Step 3: Add the provided InfraredSensorsTEP.Hex file to both IR sensors

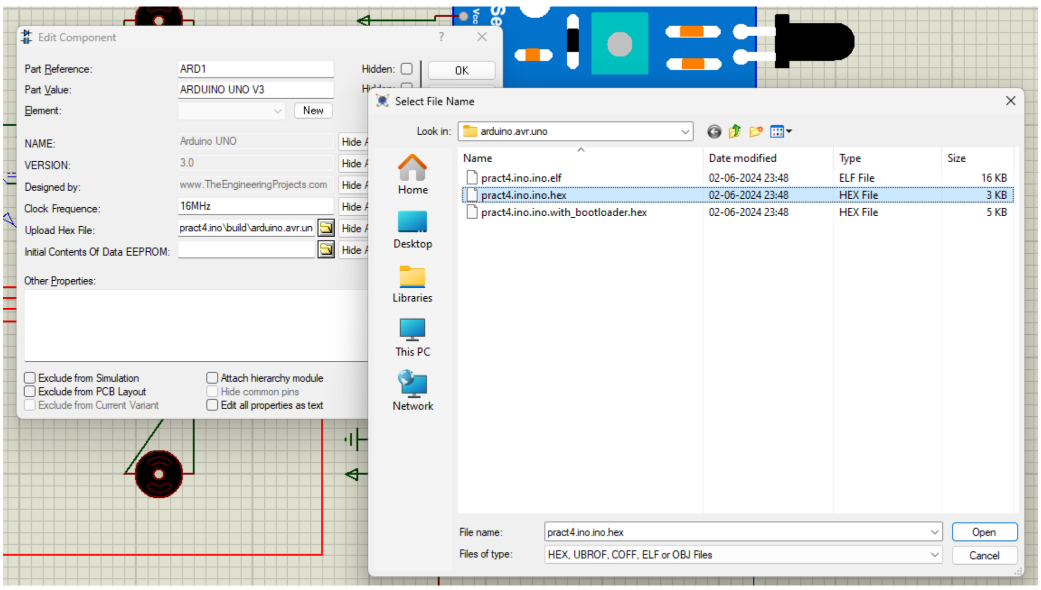


Step 3: Write the Code for Arduino and extract the hex file

```
void setup() {
  pinMode(2, INPUT);
  pinMode(3, INPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
}

void loop() {

  int v = digitalRead(2);
  int s = digitalRead(3);

  if (v == 1 and s == 1) {
```

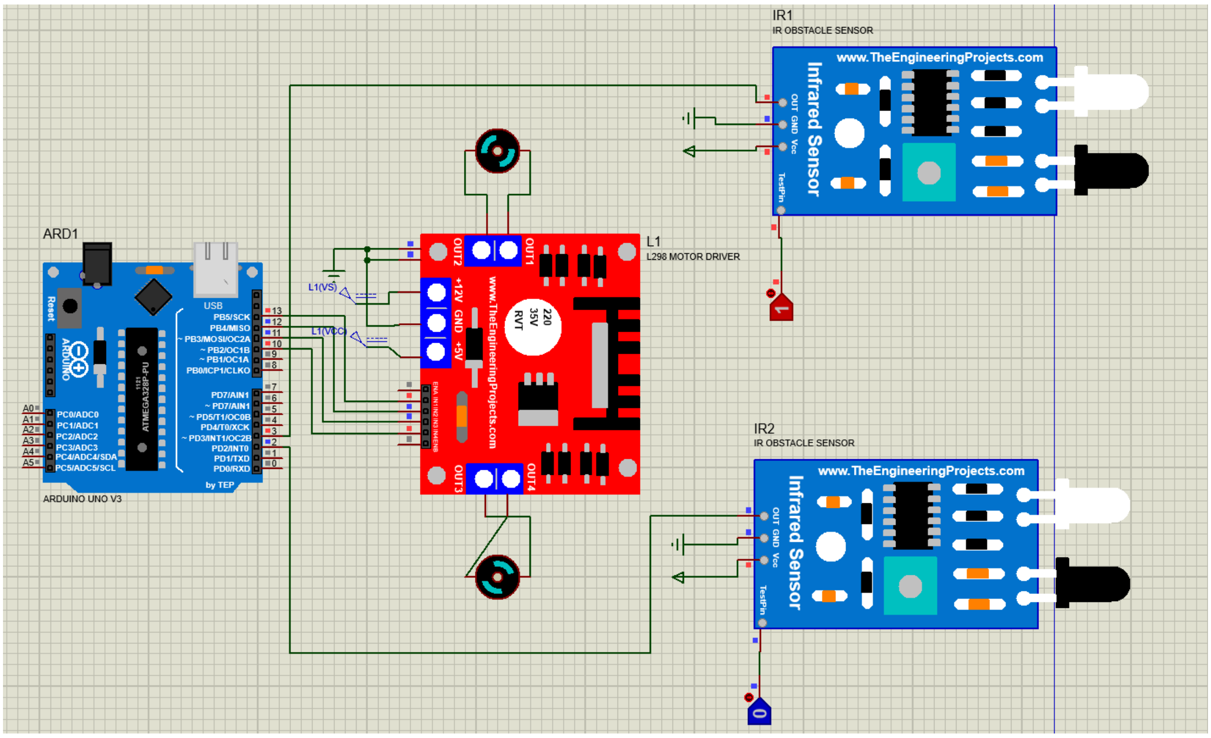
```
    digitalWrite(13, 1);
    digitalWrite(12, 0);
    digitalWrite(11, 1);
    digitalWrite(10, 0);
  }

  if (v == 1 and s == 0) {
    digitalWrite(13, 0);
    digitalWrite(12, 1);
    digitalWrite(11, 0);
    digitalWrite(10, 1);
  }

  if (v == 0 and s == 1) {

    digitalWrite(13, 1);
    digitalWrite(12, 0);
    digitalWrite(11, 0);
    digitalWrite(10, 1);
  }

  if (v == 0 and s == 0) {

    digitalWrite(13, 0);
    digitalWrite(12, 1);
    digitalWrite(11, 0);
    digitalWrite(10, 1);
  }
}
```

Step 4: Add the hex file to Arduino

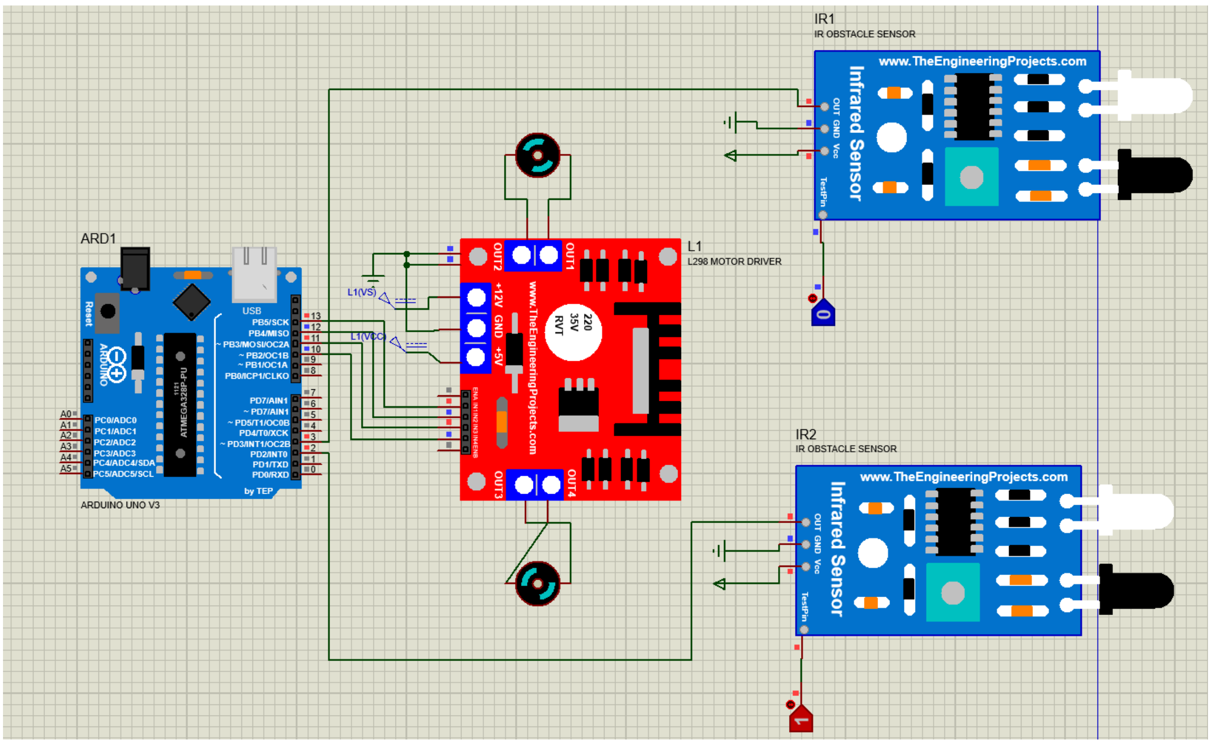


Step 5: Start the simulation

When the upper IR Sensor is on the motor spins in clockwise director (In the direction of sensor)



When lower sensor is on the lower motor spins the direction of sensor



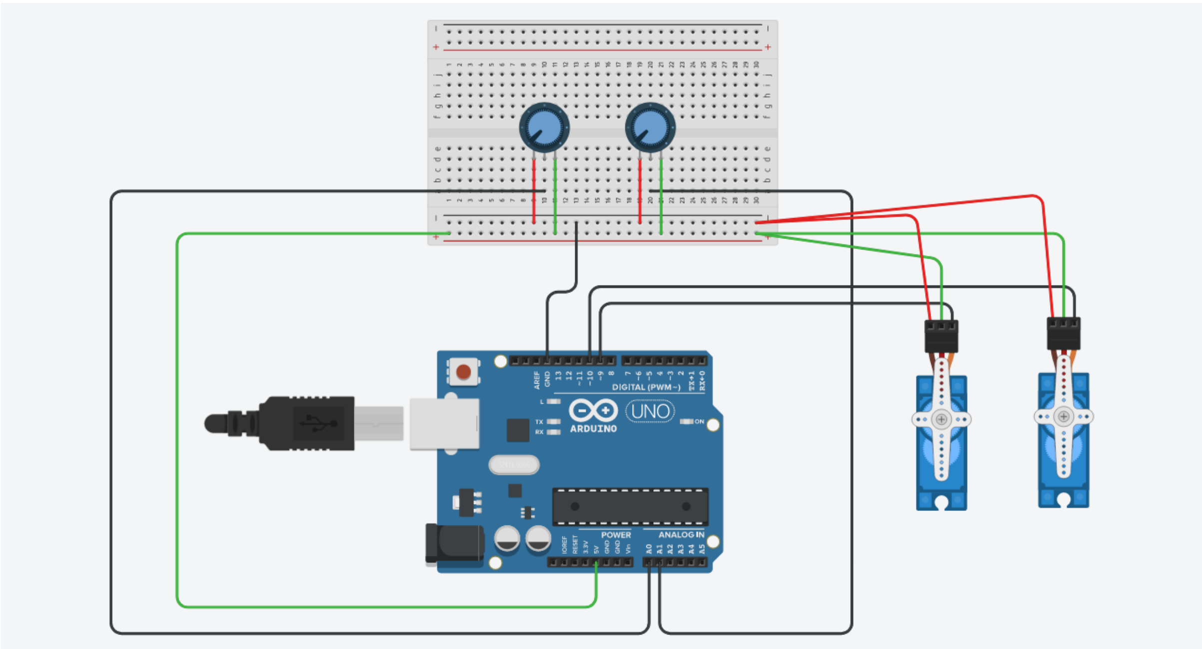
Practical 5

Aim: - Write Python code to test motors.

Components

- Arduino Uno R3
- Breadboard
- Micro Servo
- Potentiometer

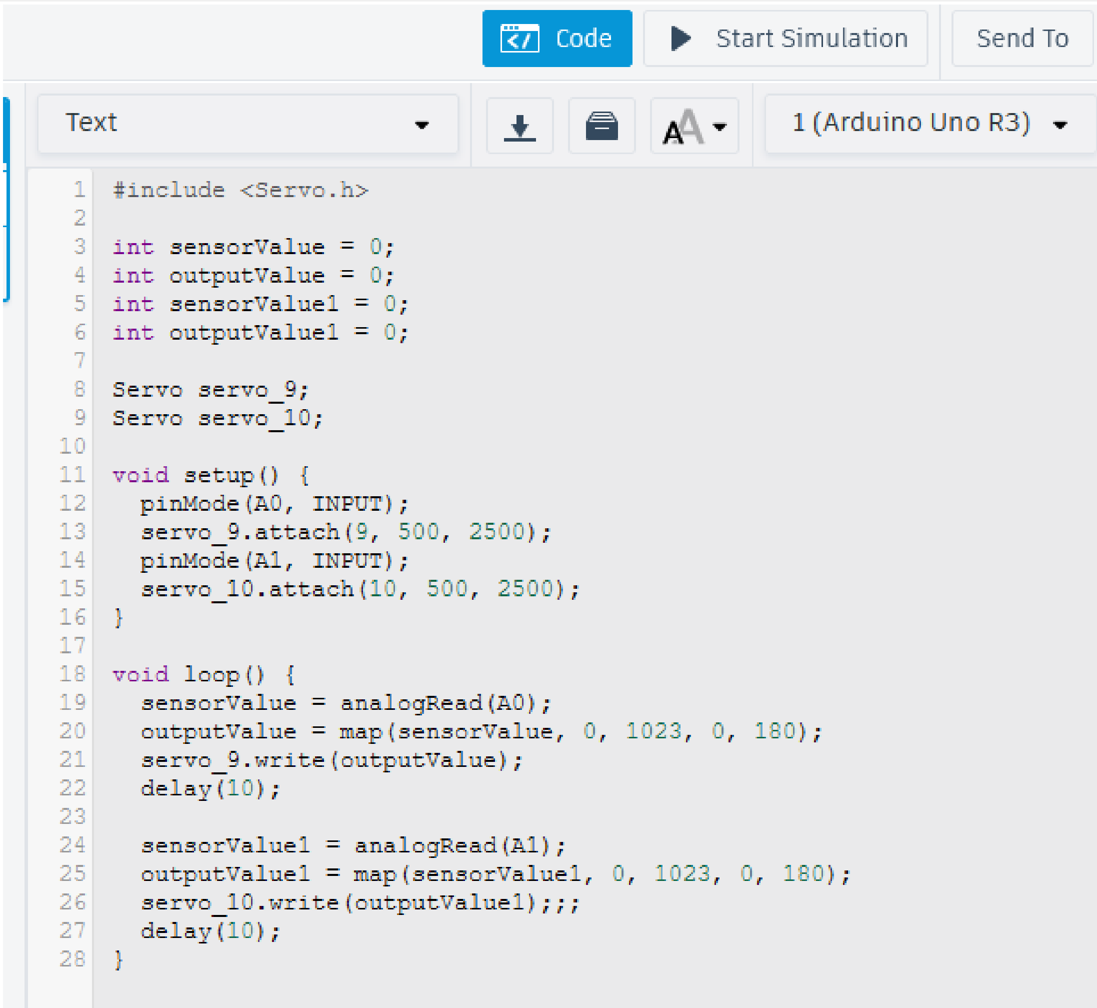
Step 1: Create the following circuit in [tinkercard]



Step 2: Write the following code

```
#include <Servo.h>

int sensorValue = 0;
int outputValue = 0;
int sensorValue1 = 0;
int outputValue1 = 0;

Servo servo_9;
Servo servo_10;
```

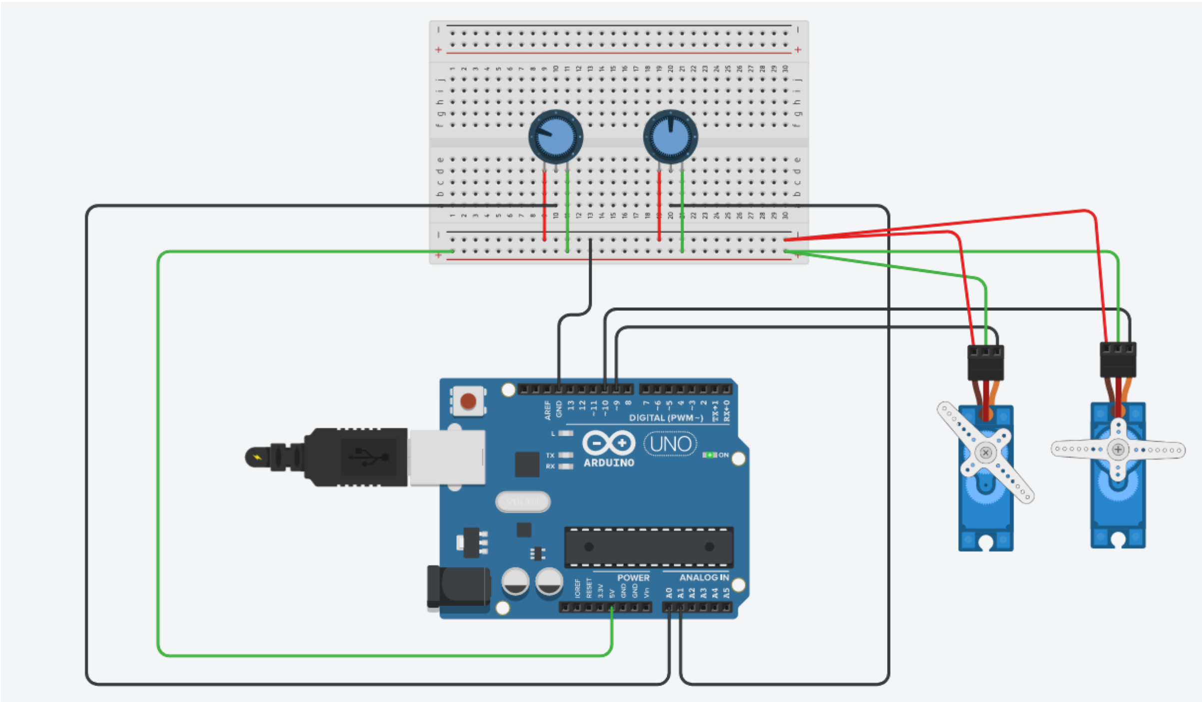
```
void setup() {
  pinMode(A0, INPUT);
  servo_9.attach(9, 500, 2500);
  pinMode(A1, INPUT);
  servo_10.attach(10, 500, 2500);
}

void loop() {
  sensorValue = analogRead(A0);
  outputValue = map(sensorValue, 0, 1023, 0, 180);
  servo_9.write(outputValue);
  delay(10);

  sensorValue1 = analogRead(A1);
  outputValue1 = map(sensorValue1, 0, 1023, 0, 180);
  servo_10.write(outputValue1);;;
  delay(10);
}
```




```
#include <Servo.h>

int sensorValue = 0;
int outputValue = 0;
int sensorValue1 = 0;
int outputValue1 = 0;

Servo servo_9;
Servo servo_10;

void setup() {
  pinMode(A0, INPUT);
  servo_9.attach(9, 500, 2500);
  pinMode(A1, INPUT);
  servo_10.attach(10, 500, 2500);
}

void loop() {
  sensorValue = analogRead(A0);
  outputValue = map(sensorValue, 0, 1023, 0, 180);
  servo_9.write(outputValue);
  delay(10);

  sensorValue1 = analogRead(A1);
  outputValue1 = map(sensorValue1, 0, 1023, 0, 180);
  servo_10.write(outputValue1);;;
  delay(10);
}
```

Step 3: Run the Simulation

By adjusting the potentiometer the servo also rotates



Practical 6

Aim: - Detect faces with Haar cascades.

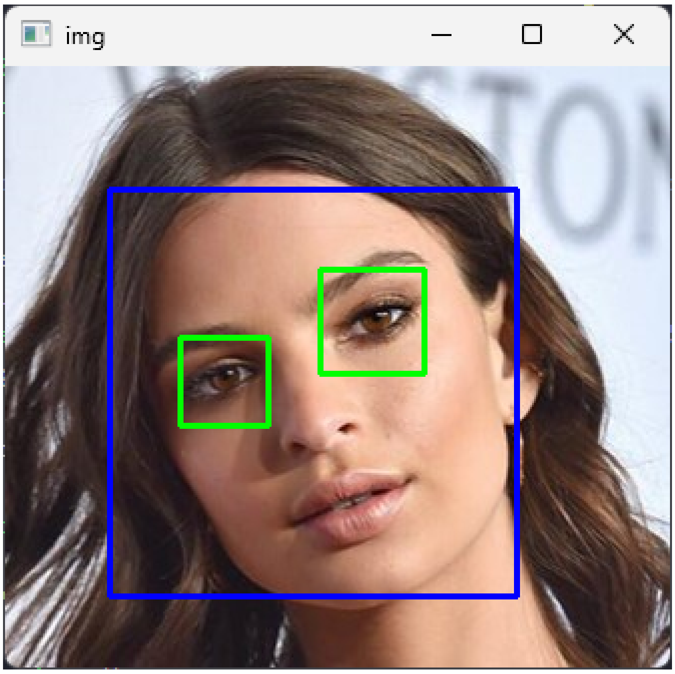
Pre Requisites

```
pip install opencv-python
```

Code: (Make sure all the files are in the same directory)

```
import numpy as np
import cv2

# First we need to load the required XML classifiers. Then load
our input image (or video) in grayscale mode.

face_cascade =
cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
eye_cascade = cv2.CascadeClassifier("haarcascade_eye.xml")

img = cv2.imread("test-image.jpg")
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Now we find the faces in the image. If faces are found, it
returns the positions of detected faces as Rect(x,y,w,h). Once we
get these locations, we can create a ROI for the face and apply
eye detection on this ROI (since eyes are always on the face !!!
).
faces = face_cascade.detectMultiScale(gray, 1.3, 5)
for x, y, w, h in faces:
      img = cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0),
2)
      roi_gray = gray[y : y + h, x : x + w]
      roi_color = img[y : y + h, x : x + w]
      eyes = eye_cascade.detectMultiScale(roi_gray)
      for ex, ey, ew, eh in eyes:
      cv2.rectangle(roi_color, (ex, ey), (ex + ew, ey + eh), (0,
255, 0), 2)

cv2.imshow("img", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:



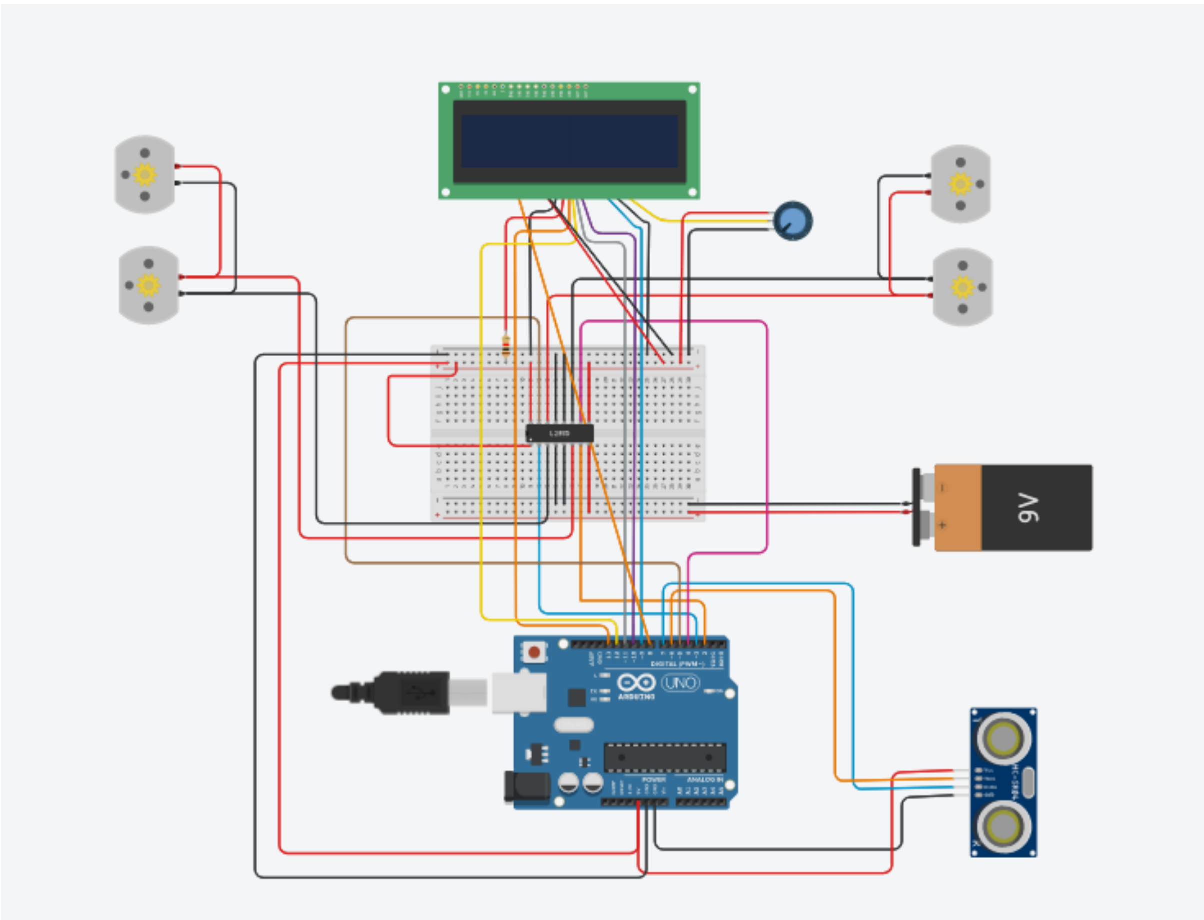
Practical 7

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

Components:

1) 4 Dc Motors
2) 9 Volt Battery
3) Arduino UNO R3
4) Breadboard Small
5) 1 Resistor
6) 1 Ultrasonic Sensor
7) 1 H-bridge Motor Driver
8) 1 Potentiometer

Step 1: Connect all the device with the wires and complete the circuit (Using TinkerCad)



Step 2: Now we have to write the code for the following circuit

Code:

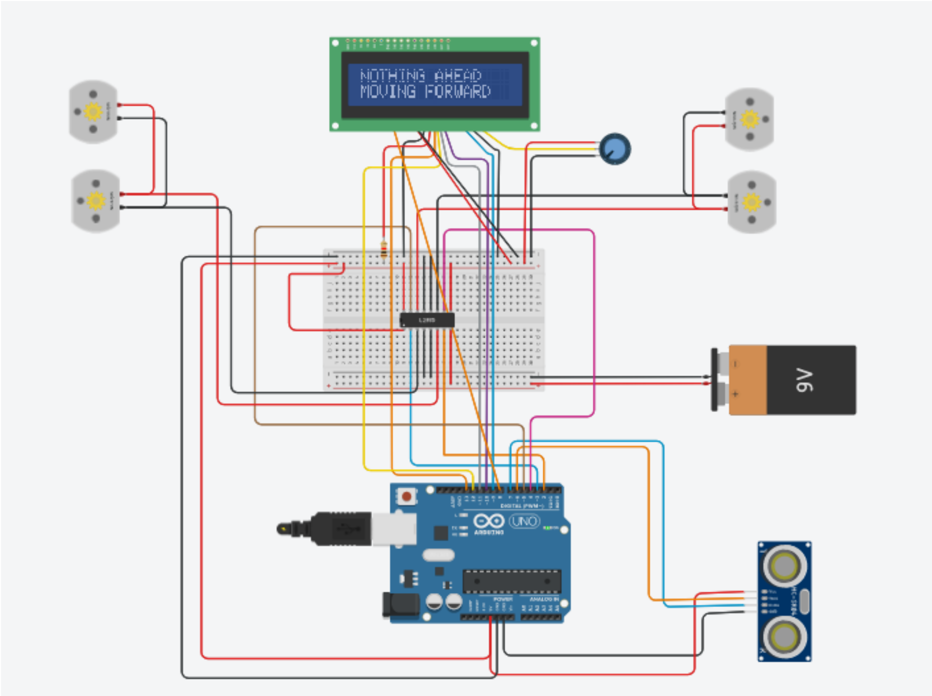
```
#include <LiquidCrystal.h>

// Initialize the library with the numbers of the interface pins
LiquidCrystal lcd(8, 9, 10, 11, 12, 13);

long cm, duration;
const int echoPin = 7;
const int trigPin = 6;
const int lm1 = 2;
const int lm2 = 3;
const int rm1 = 4;  // Corrected pin name from rm3 to rm1
const int rm2 = 5;  // Corrected pin name from rm4 to rm2

void setup() {
  pinMode(lm1, OUTPUT);
  pinMode(lm2, OUTPUT);
  pinMode(rm1, OUTPUT);
  pinMode(rm2, OUTPUT);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
  lcd.begin(16, 2);
}

void loop() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(5);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  // Converting time into distance in centimetres
  cm = duration * 0.034 / 2;

  if (cm < 20) {
    stop_bot();
    delay(2000);
    go_back();
    delay(2000);
```

```
    stop_again();
    delay(1000);
    go_left();
    delay(1000);
  } else {
    go_straight();
    delay(1000);
  }

  Serial.print("Distance: CM ");
  Serial.println(cm);
}

void go_straight() {
  lcd.setCursor(0, 0);
  lcd.print("NOTHING AHEAD");
  lcd.setCursor(0, 1);
  lcd.print("MOVING FORWARD");
    digitalWrite(lm1, HIGH);
    digitalWrite(lm2, LOW);
    digitalWrite(rm1, HIGH);
    digitalWrite(rm2, LOW);
}

void go_back() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("TAKING REVERSE");
  lcd.setCursor(0, 1);
  lcd.print(cm);
  digitalWrite(lm1, LOW);
  digitalWrite(lm2, HIGH);
  digitalWrite(rm1, LOW);
  digitalWrite(rm2, HIGH);
}

void stop_bot() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("SOMETHING AHEAD");
  lcd.setCursor(0, 1);
  lcd.print("STOP!");
```

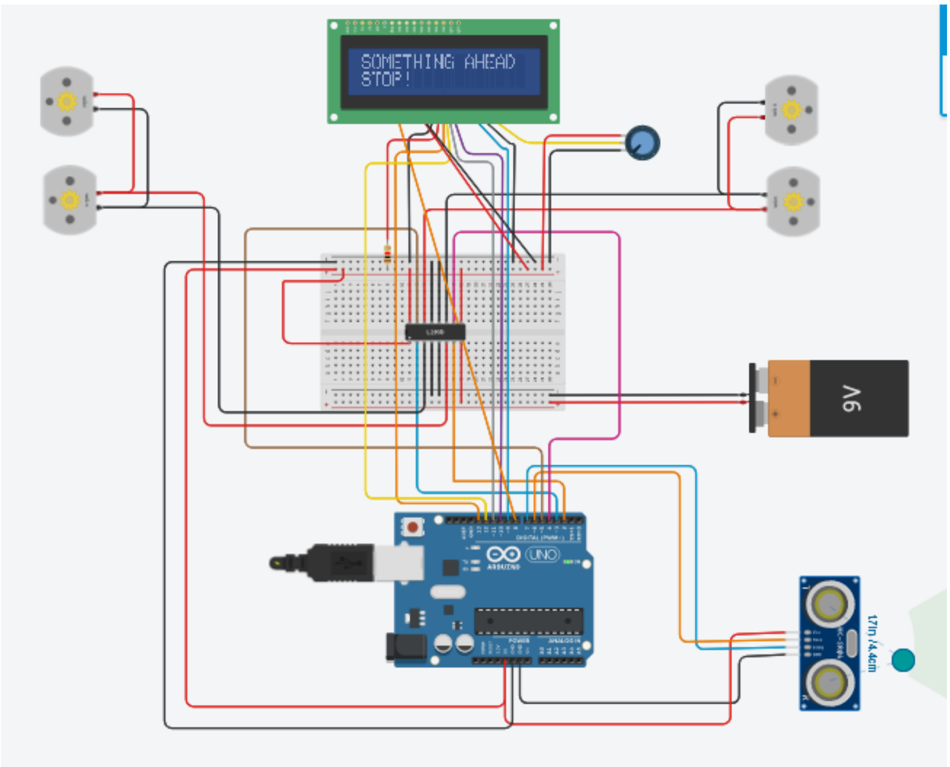
```
  digitalWrite(lm1, LOW);
  digitalWrite(lm2, LOW);
  digitalWrite(rm1, LOW);
  digitalWrite(rm2, LOW);
}

void stop_again() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("BREAK FOR TURN");
  digitalWrite(lm1, LOW);
  digitalWrite(lm2, LOW);
  digitalWrite(rm1, LOW);
  digitalWrite(rm2, LOW);
}

void go_left() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("TURNING LEFT");
  lcd.setCursor(0, 1);
  lcd.print(cm);
  digitalWrite(lm1, LOW);
  digitalWrite(lm2, LOW);
  digitalWrite(rm1, HIGH);
  digitalWrite(rm2, LOW);
}

void go_right() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("TURNING RIGHT");
  lcd.setCursor(0, 1);
  lcd.print(cm);
  digitalWrite(lm1, HIGH);
  digitalWrite(lm2, LOW);
  digitalWrite(rm1, LOW);
  digitalWrite(rm2, LOW);
}
```

Step 3: Once after writing the code, click on the Start Simulation Button and now you can see the circuit is running

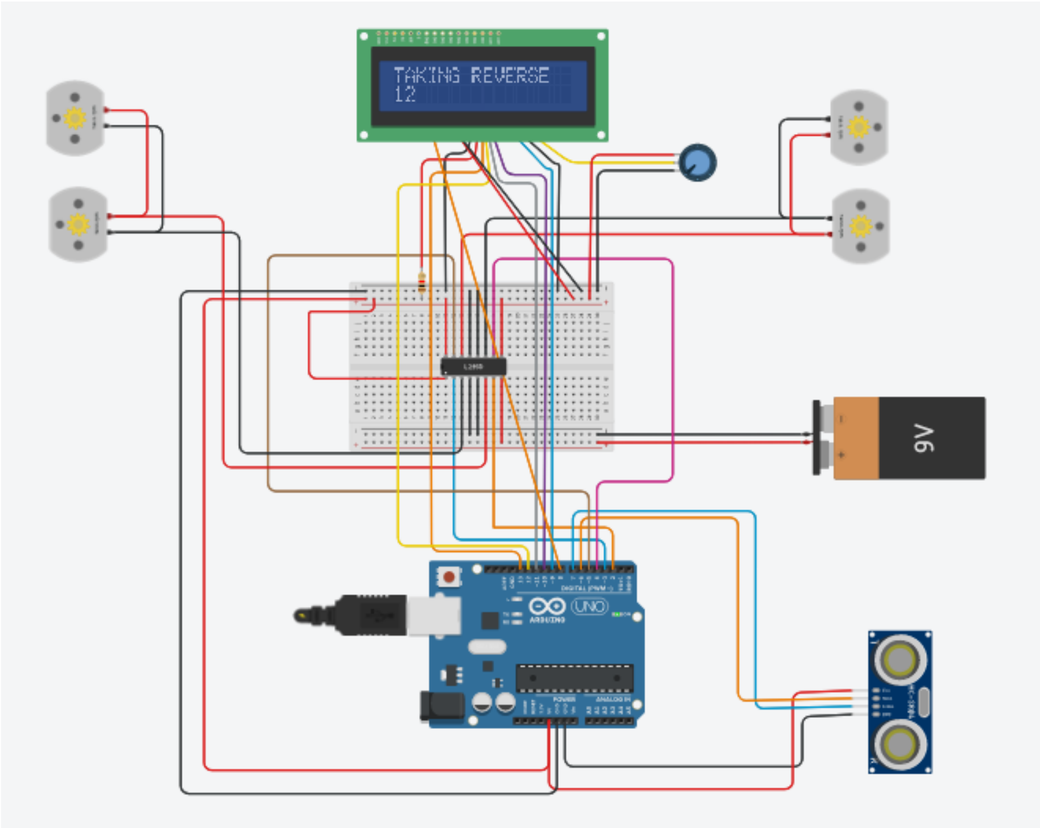
Robot Moving Ahead as there is no obstacle



Robot stops as the sensor detects objects in front.



Now the Robot is going in reverse direction



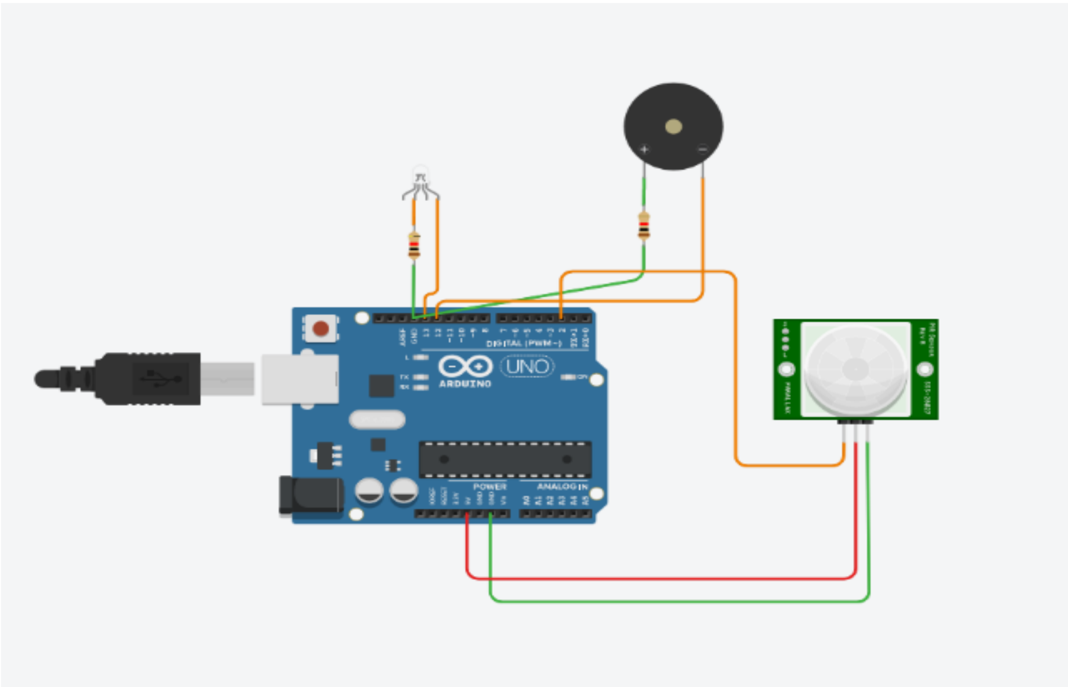
Practical 8

Aim: Develop Python code for testing the sensors.

Components:

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

Step 1: Connect all the device with wires and complete



Step 2: Code:

```
int pirsensor = 0;

void setup() {
  pinMode(2, INPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
}

void loop() {
  pirsensor = digitalRead(2); // Added semicolon here
```

```
  if (pirsensor == HIGH) {
    digitalWrite(13, HIGH);
    tone(12, 500, 500);
  } else {
    digitalWrite(13, LOW); // Added semicolon here and else block
  }
}
```

Output: