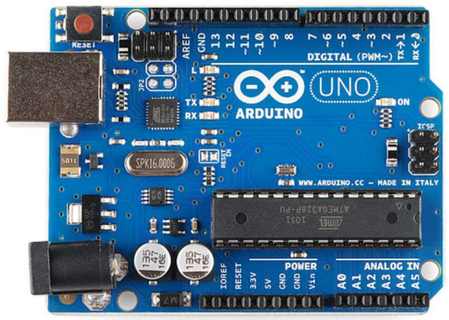
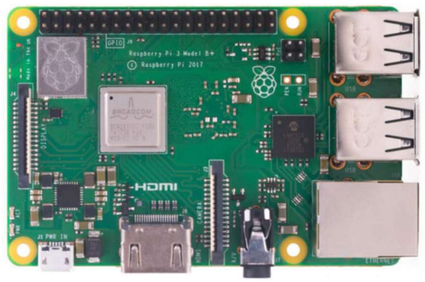
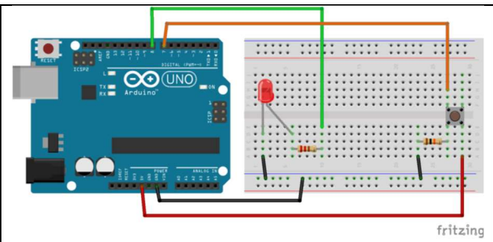
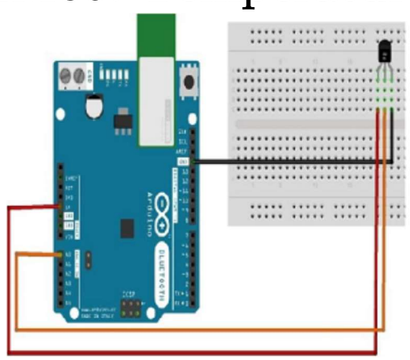
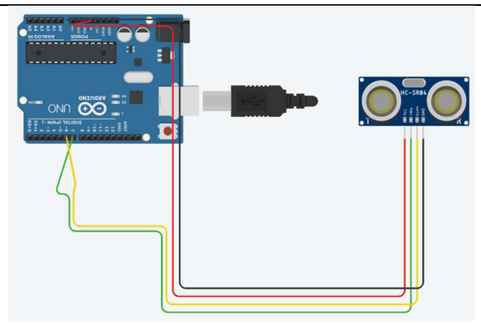
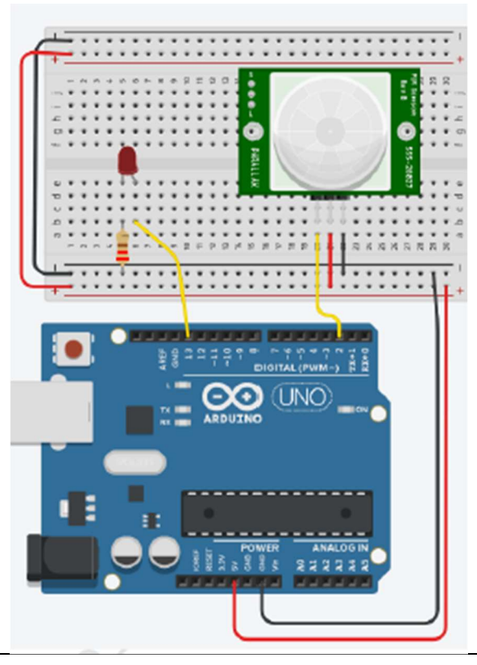
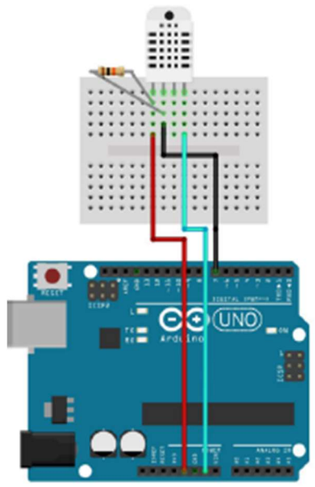
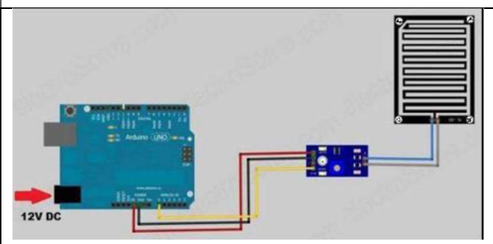
Modernized Internet of Things
Lab Practice - 2 [404187]

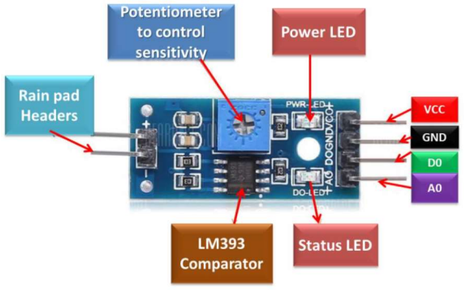
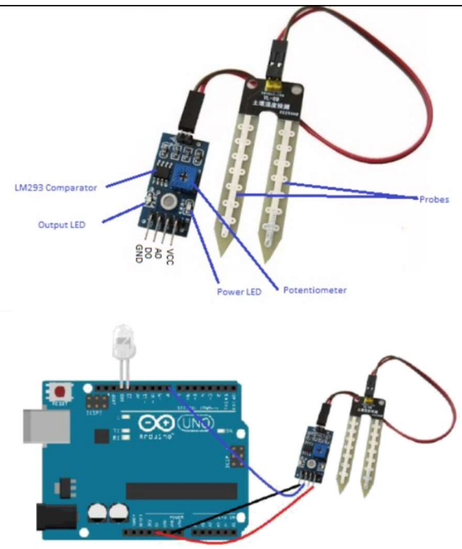
<table>
<tr><th>SN</th><th>Experiment</th><th>Circuit</th><th colspan="2">Code/Specifications</th><th>Things to remember</th></tr>
<tr>
<td>1</td>
<td>To study Raspberry Pi 4, Arduino board and operating systems and process of OS installation on the Raspberry Pi.</td>
<td> 
 
Identify the peripherals of Arduino Uno and Raspberry Pi Board</td>
<td>Arduino Uno Specifications:
o Microcontroller: ATmega328P
o Operating Voltage: 5V
o Input Voltage (recommended): 7-12V
o Inout Voltage (limit): 6-20V
o Digital I/O Pins: 14 (of which 6 provide PWM output)
o PWM Digital I/O Pins: 6
o Analog Input Pins: 6
o DC Current per I/O Pin: 20 mA
o DC current for 3.3V Pin: 50 mA
o Flash Memory: 32 KB (ATmega328P) of which 0.5 KB used by bootloader
o SRAM: 2 KB (ATmega328P)
o EEPROM: 1 KB (ATmega328P)
o Clock Speed: 16 MHz
o LED_BUILTIN: 13
o Length: 68.6 mm
o Width: 58.4 mm
o Weight: 25 g</td>
<td>Raspberry Pi model 3 B Specifications:
o SOC: Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC
o CPU: 1.4GHz 64-bit quad-core ARM Cortex-A53 CPU
o RAM: 1GB LPDDR2 SDRAM
o WIFI: Dual-band 802.11ac wireless LAN (2.4GHz and 5GHz) and Bluetooth 4.2
o Ethernet: Gigabit Ethernet over USB 2.0 (max 300 Mbps). Power-over-Ethernet support (with separate PoE HAT). Improved PXE network and USB mass-storage booting.
o Thermal management: Yes
o Video: Yes – Video Core IV 3D. Full-size HDMI
o Audio: Yes
o USB 2.0: 4 ports
o GPIO: 40-pin
o Power: 5V/2.5A DC power input
o Operating system support: Linux and Unix
o Bluetooth 4.2 Low Energy
o Faster onboard Ethernet, up to 300mbps speed</td>
<td>Procedure to install the Operating System
o Install the SD Formatter software in Computer/Laptop
o Insert the micro-SD card into the card reader and connect it to Computer/Laptop
o Select correct drive & format the SD card
o Download the OS from official web site of raspberry pi
https://www.raspberrypi.org/downloads/noobs/
o Download zip file
o Extract the zip file into SD card
o Insert the micro-SD card into the slot on the underside of the Pi
o Plug in keyboard and mouse
o Plug in monitor using the HDMI port
o The Raspberry Pi doesn’t have a power button.
o It boots up as soon as you plug in the power supply
o If you’ve completed all the previous steps, plug in the power supply to boot the Raspberry Pi</td>
</tr>
</table>

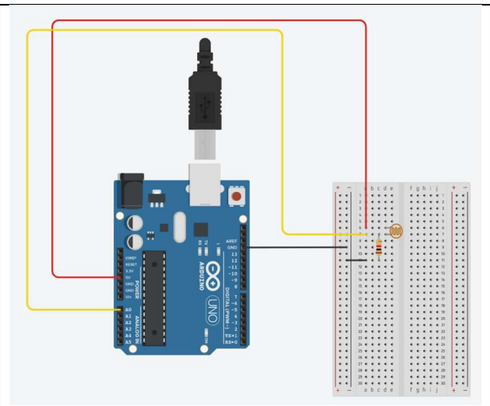
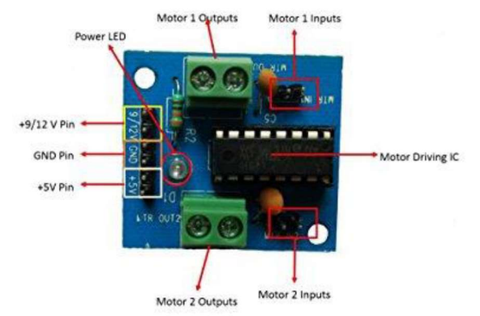
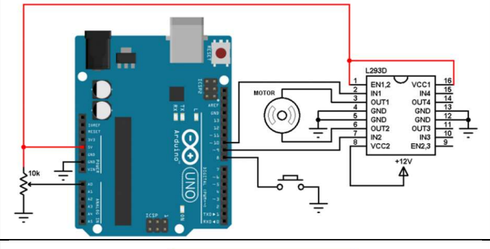
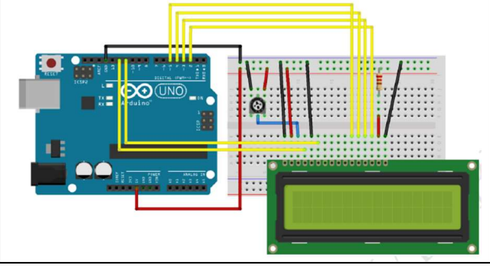
2	To study interfacing LED and Push Button with Arduino Board.		**Turn on the LED when button is pressed, turn it off otherwise** #define LED_PIN 8 #define BUTTON_PIN 7 void setup() { pinMode(LED_PIN, OUTPUT); pinMode(BUTTON_PIN, INPUT); } void loop() { if (digitalRead(BUTTON_PIN) == HIGH) { digitalWrite(LED_PIN, HIGH); } else { digitalWrite(LED_PIN, LOW); } } **Toggle LED's state with the push button** #define LED_PIN 8 #define BUTTON_PIN 7 byte lastButtonState = LOW; byte ledState = LOW; void setup() { pinMode(LED_PIN, OUTPUT); pinMode(BUTTON_PIN, INPUT); } void loop() { byte buttonState = digitalRead(BUTTON_PIN); if (buttonState != lastButtonState) { lastButtonState = buttonState; if (buttonState == LOW) { ledState = (ledState == HIGH) ? LOW: HIGH; digitalWrite(LED_PIN, ledState); } } } **Turn LED on and off with button – using debounce** #define LED_PIN 8 #define BUTTON_PIN 7 byte lastButtonState = LOW; byte ledState = LOW; unsigned long debounceDuration = 50; // millis unsigned long lastTimeButtonStateChanged = 0; void setup() {	**Step by step instructions to build the circuit** o First, make sure to power off your Arduino – remove any USB cable. o Plug a black wire between the blue line of the breadboard and a ground (GND) pin on the Arduino board. o Plug the LED. You can notice that the LED has a leg shorter than the other. Plug this shorter leg to the ground (blue line here) of the circuit. o Connect the longer leg of the LED to a digital pin (here pin no 8, you can change it). Add a 220 Ohm resistor in between to limit the current going through the LED. o Add the push button to the breadboard, like in the picture. o Connect one leg of the button to the ground and put a 10k Ohm resistor in between. This resistor will act as a "pull down" resistor, which means that the default button's state will be LOW. o Add a red wire between another leg of the button and VCC (5V). o Finally, connect a leg of the button (same side as the pull-down resistor) to a digital pin (here 7). **Read all explanation regarding all 3 cases.**

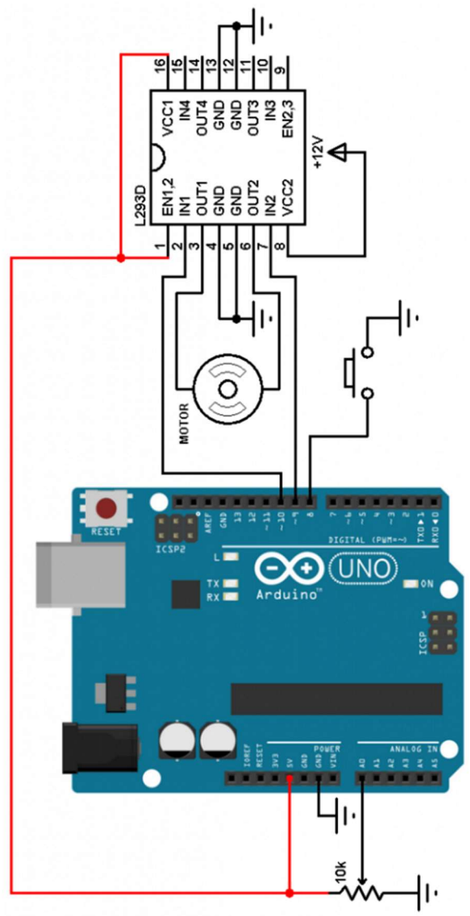
			pinMode(LED_PIN, OUTPUT); pinMode(BUTTON_PIN, INPUT); } void loop() { if (millis() - lastTimeButtonStateChanged > debounceDuration) { byte buttonState = digitalRead(BUTTON_PIN); if (buttonState != lastButtonState) { lastTimeButtonStateChanged = millis(); lastButtonState = buttonState; if (buttonState == LOW) { ledState = (ledState == HIGH) ? LOW: HIGH; digitalWrite(LED_PIN, ledState); } } } }	
3	To interface Temperature Sensor LM35	LM35: Temperature 	float temp; int tempPin = 0; void setup() { Serial.begin(9600); } void loop() { temp = analogRead(tempPin); // read analog volt from sensor and save to variable temp temp = temp * 0.48828125. // convert the analog volt to its temperature equivalent Serial.print("TEMPERATURE = "); Serial.print(temp); // display temperature value Serial.print("*C"); Serial.println(); delay(1000); // update sensor reading each one second }}	LM35 sensor has three terminals - Vs, Vout and GND. We will connect the sensor as follows 1. Connect the +Vs to +5v on your Arduino board. 2. Connect Vout to Analog0 or A0 on Arduino board. 3. Connect GND with GND on Arduino. The Analog to Digital Converter (ADC) converts analog values into a digital approximation based on the formula ADC Value = sample * 1024 / reference voltage (+5v). **https://www.tinkercad.com/things/cKDTozdqW00-copy-of-lm35-temperature-sensor/editel**
4	To interface Ultrasonic Sensor	Ultrasonic Sensor	const int pingPin = 7; // Trigger Pin of Ultrasonic Sensor const int echoPin = 6; // Echo Pin of Ultrasonic Sensor void setup() { Serial.begin(9600); // Starting Serial Terminal } void loop() { long duration, inches, cm; pinMode(pingPin, OUTPUT); digitalWrite(pingPin, LOW);	The Ultrasonic sensor has four terminals - +5V, Trigger, Echo, and GND connected as follows 1. Connect the +5V pin to +5v on your Arduino board. 2. Connect Trigger to digital pin 7 on your Arduino board. 3. Connect Echo to digital pin 6 on your Arduino board.

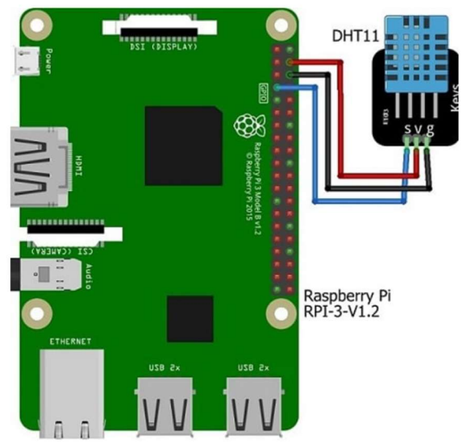
			delayMicroseconds(2); digitalWrite(pingPin, HIGH); delayMicroseconds(10); digitalWrite(pingPin, LOW); pinMode(echoPin, INPUT); duration = pulseIn(echoPin, HIGH); inches = microsecondsToInches(duration); cm = microsecondsToCentimeters(duration); Serial.print(inches); Serial.print("in, "); Serial.print(cm); Serial.print("cm"); Serial.println(); delay(100); } long microsecondsToInches(long microseconds) { return microseconds / 74 / 2; } long microsecondsToCentimeters(long microseconds) { return microseconds / 29 / 2; }	4. Connect GND with GND on Arduino. In the program, it has displayed the distance measured by the sensor in inches and cm via the serial port. **https://www.tinkercad.com/things/jtbM7IEgHW3-copy-of-arduino-ultrasonic-distance-sensor/editel**
5	To interface PIR Sensor	PIR Sensor 	int pirsensor = 0; void setup() { pinMode(2, INPUT); pinMode(13, OUTPUT); Serial.begin(9600); } void loop() { pirsensor = digitalRead(2); if (pirsensor == 1) { digitalWrite(13, HIGH); Serial.println("Motion detected , presence of object"); } else { digitalWrite(13, LOW); Serial.println("Motion not detected"); } delay(1000); // Wait for 1000 millisecond(s)	These pins are connected to pins 2, 5V and GND of Arduino respectively. An LED whose anode terminal is connected to Arduino's output pin 13 and the cathode terminal is connected to the ground pin of Arduino and a resistor is placed between the LED anode terminal and pin number 13 which help us to limit the current and prevent LED from burning it. In this circuit when the PIR sensor detects any infrared signal then the LED will Light up otherwise it will be off. **https://www.tinkercad.com/things/2w8BLOs3kcO-copy-of-pir-motion-sensor-with-arduino-blocks/editel**

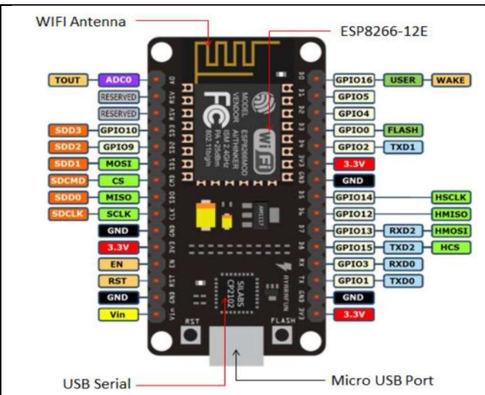
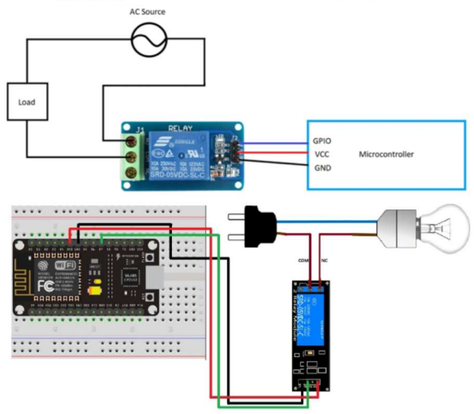
6	To interface DHT11: Humidity & Temperature Sensor	DHT11: Humidity & Temperature Sensor 	#include <dht.h> dht DHT; #define DHT11_DPIN2 Void setup() { Serial.begin(9600); } Void loop() { int chk=DHT.read11(DHT11_DPIN); serial.print("Temperature = "); serial.println(DHT.temperature); serial.print("Humidity = "); serial.println(DHT.humidity); delay(1000); } **VCC – 5V Pin** **DATA – Digital Pin 7** **NC- No Contact to Arduino** **Ground- GND Pin**	DHT 11 sensor is a tiny and cheap sensor. DHT11 is a three-wire sensor which has ground, Vcc, and a data pin. and all the information you will get both temperature and humidity from a single data line. There are two sensors inside the module one is for humidity and another is for temperature. 1. DATA pin accepts digital data to be transmitted. from this data pin, all the data will be collected by the arduino. 2. VCC supplies power to the transmitter. This should be any positive DC voltage from 3.5V to 12V. 3. GND is a ground pin. 4. N/C pin will not be connected The DHT11 uses relative humidity by calculating the electrical resistance between two electrodes used in the sensor. In DHT 11 sensor the humidity sensing component is a moisture-holding substrate with the electrodes applied. When the substrate absorbs the water vapor, ions are released by the same which can increase the conductivity between the two electrodes. The change in resistance between these electrodes should be proportional to the relative humidity. As the fact, Higher relative humidity can decrease the resistance and the resistance between electrodes can increase with the decrease in relative humidity. To program the DHT11 with Arduino you will need the libraries. DHT11 sensor connected to the Arduino digital pin. All the output is dependent on code. There are two sensors inside the module one is a humidity sensor, and another is temperature sensor. The sensor always continuously sends the data by the data pin. Arduino collects that and then Arduino process this data and shows this data on the serial monitor.
7	To interface Raindrop Sensor		const int sensorMin = 0; const int sensorMax = 1024; void setup() { Serial.begin(9600); } void loop() {	**https://lastminuteengineers.com/rain-sensor-arduino-tutorial/**

			int sensorReading = analogRead(A0); int range = map(sensorReading, sensorMin, sensorMax, 0, 3); switch (range) { case 0: Serial.println("RAINING"); break; case 1: Serial.println("RAIN WARNING"); break; case 2: Serial.println("NOT RAINING"); break; } delay(1000); }	
8	To interface Soil Moisture Sensor		**Analog Mode** int sensor_pin = A0; int output_value ; void setup() { Serial.begin(9600); Serial.println("Reading From the Sensor ..."); delay(2000); } void loop() { output_value= analogRead(sensor_pin); output_value = map(output_value,550,0,0,100); Serial.print("Mositure : "); Serial.print(output_value); Serial.println("%"); delay(1000); } **Digital Mode** int led_pin =13; int sensor_pin =8; void setup() { pinMode(led_pin, OUTPUT); pinMode(sensor_pin, INPUT); } void loop() { if(digitalRead(sensor_pin) == HIGH) {	**Probes Connected to** **VCC-5V Pin** **A0-Analog Pin A0** **D0-Digital Pin 8** **GND-GND Pin**

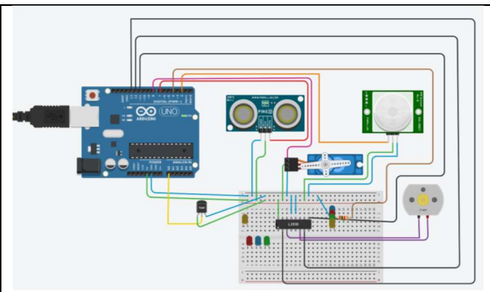
			digitalWrite(led_pin, HIGH); } else { digitalWrite(led_pin, LOW); delay(1000); }}}	
9	To interface LDR Sensor		int sensorPin = A0; // select the input pin for LDR int sensorValue = 0; // variable to store the value coming from the sensor void setup() { Serial.begin(9600); //sets serial port for communication } void loop() { sensorValue = analogRead(sensorPin); // read the value from the sensor Serial.println(sensorValue); //prints the values coming from the sensor on the screen delay(100); }	https://www.tinkercad.com/things/iJeegt2cCRW - copy-of-ldr-sensor/editel?tenant=circuits
10	To interface DC MOTOR	 	int motorpin1 = 6; //define digital output pin no. int motorpin2 = 7; //define digital output pin no. void setup () { pinMode(motorpin1,OUTPUT); //set pin 3 as output pinMode(motorpin2,OUTPUT); // set pin 4 as output } void loop () { digitalWrite(motorpin1,LOW); digitalWrite(motorpin2,HIGH); }	https://simple-circuit.com/arduino-dc-motor-speed-direction-control-l293d/
11	To interface LCD		#include <LiquidCrystal.h> LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // sets the interfacing pins void setup() { lcd.begin(16, 2); // initializes the 16x2 LCD }	**PIN CONNECTIONS** o LCD RS pin to digital pin 12 o LCD Enable pin to digital pin 11 o LCD D4 pin to digital pin 5 o LCD D5 pin to digital pin 4 o LCD D6 pin to digital pin 3 o LCD D7 pin to digital pin 2

			void loop() { lcd.setCursor(0,0); lcd.print("16x2 LCD MODULE"); lcd.setCursor(2,1); lcd.print("hello,world!"); }	
12	To interface DC MOTOR – SPIN & DIRECTION CONTROL		// Arduino DC motor speed and direction control #define button 8 #define pot 0 #define pwm1 9 #define pwm2 10 boolean motor_dir = 0; int motor_speed; void setup() { pinMode(button, INPUT_PULLUP); pinMode(pwm1, OUTPUT); pinMode(pwm2, OUTPUT); } void loop() { motor_speed = analogRead(pot) / 4; if(motor_dir) analogWrite(pwm1, motor_speed); else analogWrite(pwm2, motor_speed); if(!digitalRead(button)){ // If direction button is pressed while(!digitalRead(button)); // Wait until direction button released motor_dir = !motor_dir; // Toggle direction variable if(motor_dir) digitalWrite(pwm2, 0); else digitalWrite(pwm1, 0); } }	https://simple-circuit.com/arduino-dc-motor-speed-direction-control-l293d/

13	Sending sensor data to cloud using Raspberry-Pi		import thingspeak import time import Adafruit_DHT channel_id = 1391845 # put here the ID of the channel you created before write_key = 'TNXXJJII892UHJ1C' # update the "WRITE KEY" pin = 4 sensor = Adafruit_DHT.DHT11 def measure(channel): try: humidity, temperature = Adafruit_DHT.read_retry(sensor, pin) if humidity is not None and temperature is not None: print('Temperature = {0:0.1f}*C Humidity = {1:0.1f}%'.format(temperature, humidity)) else: print('Did not receive any reading from sensor. Please check!') # update the value response = channel.update({'field1': temperature, 'field2': humidity}) except: print("connection failure") if __name__ == "__main__": channel = thingspeak.Channel(id=channel_id, write_key=write_key) while True: measure(channel) #free account has a limitation of 15sec between the updates time.sleep(15)	https://www.iotstarters.com/how -to-send-sensor-data-to-thingspeak-using-raspberry-pi/ https://www.iotstarters.com/wp-content/uploads/2021/05/thingspeak_DHT11_code.zip IMPORTANT: READ all the configuration mentioned on the website. 1. update the packages installed in Raspberry Pi 2. install the library to read the DHT11 or DHT22 sensors. 3. Installing Raspberry Pi Thingspeak Library 4. Update the code with Channel ID and write key values from Thingspeak portal.

14	Get the status of a bulb at a remote place (on the LAN) through web.	 	Code #include <ESP8266WiFi.h> const char* ssid = ""; const char* password = ""; int ledPin = 13; // GPIO13 WiFiServer server(80); void setup() { Serial.begin(115200); delay(10); pinMode(ledPin, OUTPUT); digitalWrite(ledPin, LOW); // Connect to WiFi network Serial.println(); Serial.println(); Serial.print("Connecting to "); Serial.println(ssid); WiFi.begin(ssid, password); while (WiFi.status() != WL_CONNECTED) { delay(500); Serial.print("."); } Serial.println(""); Serial.println("WiFi connected"); // Start the server server.begin(); Serial.println("Server started"); // Print the IP address Serial.print("Use this URL to connect: "); Serial.print("http://"); Serial.print(WiFi.localIP()); Serial.println("/"); void loop() { // Check if a client has connected WiFiClient client = server.available(); if (!client) { return; } // Wait until the client sends some data Serial.println("new client"); while(!client.available()){ delay(1); } // Read the first line of the request String request = client.readStringUntil('\r');	**STEP- 1 Installing Arduino Core for NodeMCU ESP-12E Using Arduino Boards Manager** **Step 2: Insert Link for. json NodeMCU Package Files into Arduino IDE** **Step 3: Tools - Boards Manager** **Step 4: Selecting NodeMCU Board in Arduino IDE** **Circuit Connection** o Attach NodeMCU on Breadboard. o Connect VCC of Relay with Pin 3.3V of NodeMCU. o Connect GND of Relay with GND of NodeMCU o Connect Pin Signal of Relay with Pin D7 (GPIO 13) of NodeMCU o Connect one terminal of Bulb (Blue Wire) with Pin NC (Normally Close) of Relay. o Connect one terminal of Adapter with Pin C (Common) of Relay. o Connect the other terminal of Bulb (Red Wire) with another terminal of Adapter.

```
Serial.println(request);
client.flush();
// Match the request
int value = LOW;
if (request.indexOf("/LED=ON") != -1) {
digitalWrite(ledPin, LOW);
value = LOW;
}
if (request.indexOf("/LED=OFF") != -1) {
digitalWrite(ledPin, HIGH);
value = HIGH;
}
// Set ledPin according to the request
//digitalWrite(ledPin, value);
// Return the response
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println(""); // do not forget this one
client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.print("RELAY pin is now: ");
if(value == LOW) {
client.print("On");
} else {
client.print("Off");
}
client.println("<br><br>");
client.println("<a href=\"/LED=ON\"\">
<button>Turn On </button></a>");
client.println("<a href=\"/LED=OFF\"\">
<button>Turn Off </button></a><br />");
client.println("</html>");
delay(1);
Serial.println("Client disonnected");
Serial.println("");
}
```

15	IoT based small project implementation on the topics based on small problem statements of the fields like smart home (Home Automation This project can be built on any IoT simulation platform like Tinkercad, etc.		CODE #include <Servo.h> const int pingPin = 7; int servoPin = 8; Servo servo1; void setup() { // initialize serial communication: Serial.begin(9600); servo1.attach(servoPin); pinMode(2,INPUT); pinMode(4,OUTPUT); pinMode(11,OUTPUT); pinMode(12,OUTPUT); pinMode(13,OUTPUT); pinMode(A0,INPUT); digitalWrite(2,LOW); digitalWrite(11,HIGH); } void loop() { long duration, inches, cm; pinMode(pingPin, OUTPUT); digitalWrite(pingPin, LOW); delayMicroseconds(2); digitalWrite(pingPin, HIGH); delayMicroseconds(5); digitalWrite(pingPin, LOW); // The same pin is used to read the signal from the PING))): a HIGH pulse // whose duration is the time (in microseconds) from the sending of the ping // to the reception of its echo off of an object. pinMode(pingPin, INPUT); duration = pulseIn(pingPin, HIGH); // convert the time into a distance inches = microsecondsToInches(duration); cm = microsecondsToCentimeters(duration); //Serial.print(inches); //Serial.print("in, "); //Serial.print(cm); //Serial.print("cm"); //Serial.println(); //delay(100); servo1.write(0);	https://www.tinkercad.com/things/4amSeg2H3Vi-copy-of-homeautomation-system/editel?tenant=circuits **Important:** Check the pin connections of 1. Ultrasonic Sensor 2. PIR Sensor 3. DC Motor with driver 4. Servo Motor 5. LEDs 6. LM35 Temperature Sensor To Arduino Uno Board. Simulation Project. Available at: **https://www.tinkercad.com/things/4amSeg2H3Vi-copy-of-home-automation-system**

			if(cm < 40) { servo1.write(90); delay(2000); } else { servo1.write(0); } // PIR with LED starts int pir = digitalRead(2); if(pir == HIGH) { digitalWrite(4,HIGH); delay(1000); } else if(pir == LOW) { digitalWrite(4,LOW); } //temp with fan float value=analogRead(A0); float temperature=value*0.48; Serial.println("temperature"); Serial.println(temperature); if(temperature > 20) { digitalWrite(12,HIGH); digitalWrite(13,LOW); } else { digitalWrite(12,LOW); digitalWrite(13,LOW); } } long microsecondsToInches(long microseconds) { return microseconds / 74 / 2; } long microsecondsToCentimeters(long microseconds) { return microseconds / 29 / 2; }	