DFAA Project

Department of Industrial & Systems Engineering
Indian Institute of Technology, Kharagpur
Spring Semester, 2021

Project Report

Roll No: 18QE30008	Name: Divyanshu Sheth
Title: Smart Home Switch	

1 INTRODUCTION:

Give details of the project. How the product is used, it's operating environment, and other relevant background information.

The project aims to build a smart home switch – a switch that turns on or off intelligently based on sensory data obtained from the environment. This sensory data can be of the following types: illumination (sensing the amount of light in a room using an LDR), presence of persons (using an IR/PIR sensor to detect movement), and temperature & humidity in a room (using a digital temp/humidity sensor (DHT11 sensor)). Operating conditions: regular home environment.

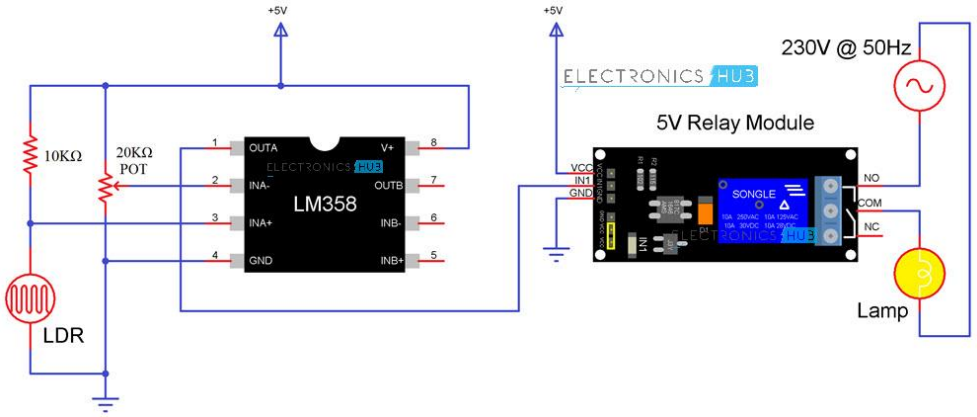
The final product functions automatically, and switches are turned on or off based on a combination of the above-mentioned sensory inputs. Switches for mainly two appliances can be controlled, a light-emitting device (bulb/tubelight/lamp, etc.) and a temperature-controlling device (fan/air-conditioner/heater, etc.)

2 SIMILAR PROJECTS

Find out about similar projects and document them

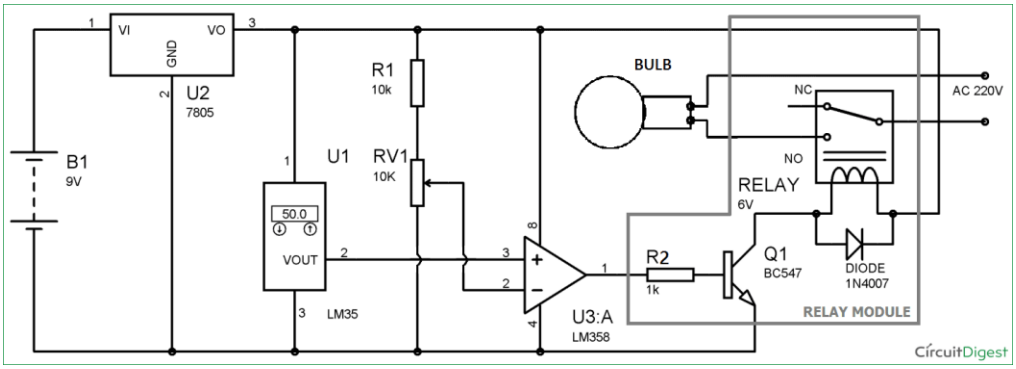
There are several projects that build smart switches using one of the sensors (either LDR, or PIR, or Temp/Humidity detection). There are also projects that use Bluetooth/Wifi modules with switches, essentially creating IoT smart switches, and some that involve speech detection and gesture detection for controlling switches. I could not find a project on the internet that uses a combination of multiple inputs (sensors) with an appropriate algorithm, however. Here are some projects that use individual sensors to control switches:

LDR - light activated switch: Circuit Diagram-



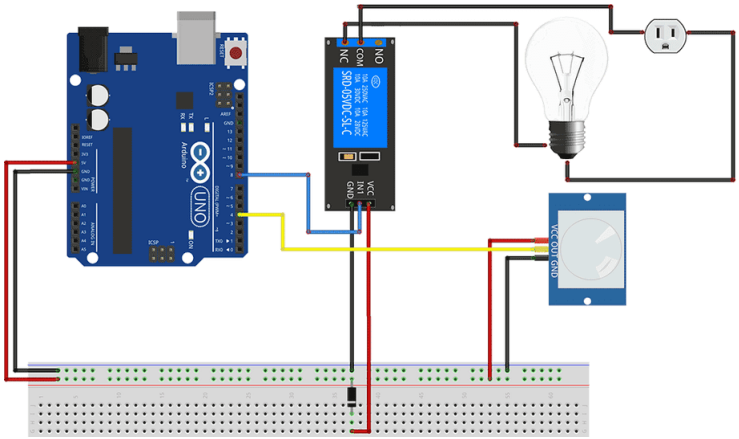
This does not involve an Arduino but uses an Op-amp IC LM358 instead. There is just the one LDR which takes inputs that control the switch.

Temperature controlled switch:



This uses the LM35 temperature sensor to control a switch. Here too, an LM358 is used.

Motion-controlled switch:



An Arduino is interfaced with a PIR sensor which detects motion and controls the switch via a relay.

3 METHODOLOGY

Give a brief description of how you plan to make the device

The three sensors (PIR/IR, LDR, DHT11) will be connected to the Arduino. An appropriate algorithm used for taking in inputs and deciding on switching behavior will be coded and run on the Arduino. The PIR/IR sensors would be placed at doors, to detect presence of people entering/leaving a room. The LDR would be placed next to lights (lamps, etc.) to sense the amount of existing light there and turn on/off those lights as needed. The temp/humidity sensor will be positioned at a convenient position in the room in question. Time of day will be taken as a factor as well, contributing towards switching on/off certain devices (eg, lights must always be on in a room where people are present after sunset). Machine learning algorithms, if possible, would be incorporated for a task like switching on an AC a few minutes prior to a person's arrival in the room (based on collected data) so the person can enter a room which has been cooled to a suitable temperature already.

4 THEORY:

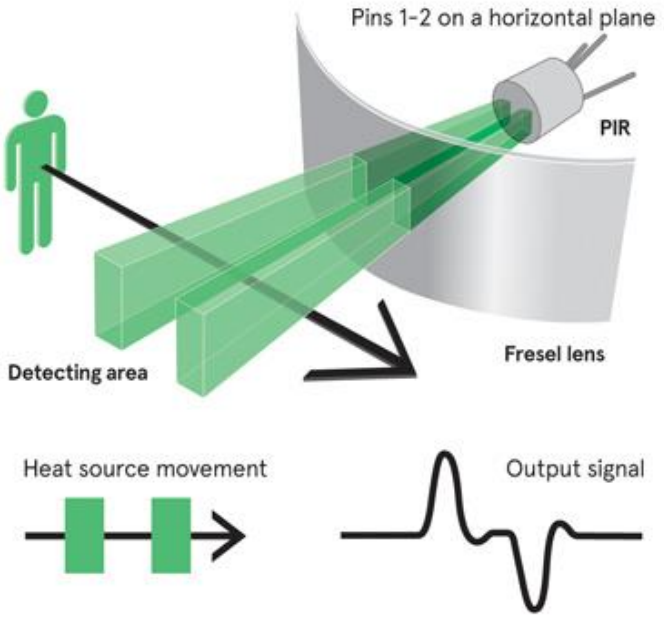
Explain key process and the theory behind your device

PIR sensor:

A passive infrared sensor is an electronic sensor that measures infrared light radiating from objects. The below image shows a typical pin configuration of the PIR sensor, which is quite simple to understand the pinouts. The PIR sensor consist of 3 pins, VCC, Output, and GND.

Generally, PIR sensors can detect animal/human movement in a requirement range. PIR is made of a pyroelectric sensor, which is able to detect different levels of infrared radiation. The detector itself does not emit any energy but passively receives it.

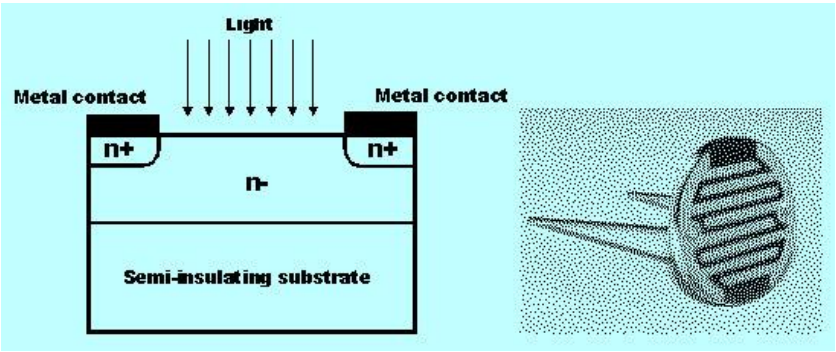
In the detection area, the lens of the detector receives the infrared radiation energy of the human body through the clothing and focused on the pyroelectric sensor. When the human body moves in this surveillance mode, it enters a certain field of view in sequence and then walks out of the field of view. The pyroelectric sensor sees the moving human body for a while and then does not see it, so the infrared radiation of human body constantly changes the temperature of the pyroelectric material. So that it outputs a corresponding signal, which is the alarm signal.



LDR:

The working principle of an LDR is photoconductivity, that an optical phenomenon. When the light is absorbed by the material then the conductivity of the material reduces. When the light falls on the LDR, then the electrons in the valence band of the material are eager to the conduction band. But, the photons in the incident light must have energy superior than the bandgap of the material to make the electrons jump from one band to another band (valence to conduction).

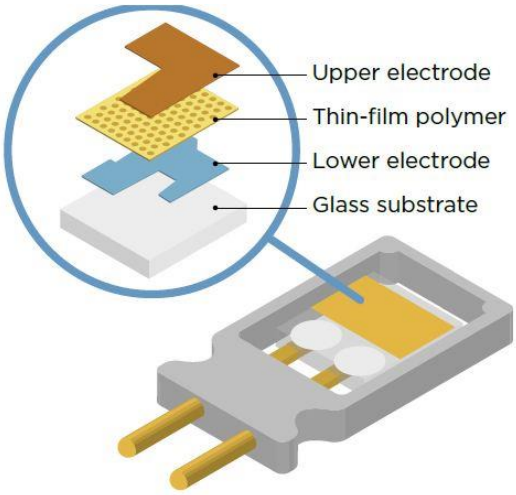
Hence, when light having ample energy, more electrons are excited to the conduction band which grades in a large number of charge carriers. When the effect of this process and the flow of the current starts flowing more, the resistance of the device decreases.



DHT11:

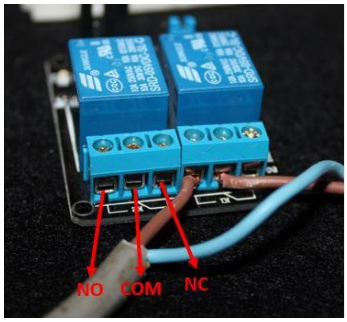
DHT11 sensor consists of a capacitive humidity sensing element and a thermistor for sensing temperature. The humidity sensing capacitor has two electrodes with a moisture holding substrate as a dielectric between them. Change in the capacitance value occurs with the change in humidity levels. The IC measure, process this changed resistance values and change them into digital form.
For measuring temperature this sensor uses a Negative Temperature coefficient thermistor, which causes a decrease in its resistance value with increase in temperature. To get larger resistance value even for the smallest change in temperature, this sensor is usually made up of semiconductor ceramics or polymers.

The temperature range of DHT11 is from 0 to 50 degree Celsius with a 2-degree accuracy. Humidity range of this sensor is from 20 to 80% with 5% accuracy.



Relay Module:

A relay is an electrically operated switch that can be turned on or off, letting the current go through or not, and can be controlled with low voltages, like the 5V provided by the Arduino pins. Relay modules often have 1, 2, 4 or 8 channels. They should be powered with 5V, which is appropriate to use with an Arduino.



Machine Learning:

MicroML is a project to bring Machine learning algorithms to microcontrollers. It was born as an alternative to Tensorflow for Microcontrollers. Here, leaner alternatives to neural networks to run inference even on 8-bit microcontrollers can be found. At the current state, it can convert Support Vector Machines to optimized C code that can be deployed on any uC of choice. Code in the scikit-learn Python library can be ported to C code and run using the MicroML library.

Regression methods can be used for predicting the time of arrival of a person in a room, based on data collected over several instances. Linear regression can be used for predicting the time. Support Vector Machines Regression can also be used. Linear regression is a linear model, e.g. a model that assumes a linear relationship between the input variables (x) and the single output variable (y). More specifically, that y can be calculated from a linear combination of the input variables (x).
When there is a single input variable (x), the method is referred to as simple linear regression. When there are multiple input variables, literature from statistics often refers to the method as multiple linear regression. Different techniques can be used to prepare or train the linear regression equation from data, the most common of which is called Ordinary Least Squares. The objective function of SVR is to minimize the coefficients, more specifically, the l2-norm of the coefficient vector. The error term is handled in the constraints, where we set the absolute error less than or equal to a specified margin, called the maximum error, ϵ (epsilon). We can tune epsilon to gain the desired accuracy of our model.

5 ACTION PLAN

Identify work steps and document time bound action plan to complete your project

	Task	Date
1.	Procure all required components, test them	03/03
2.	Create a circuit diagram	03/03
3.	Finalize code for algorithm for switching using various input parameters	07/03
4.	Test the working of the algorithm on a simulation, finalize edits if any	10/03
5.	Build the physical circuit, connect and test its working	21/03
6.	Build a machine learning implementation for certain functions (predicting certain switching actions such as AC switch-on, and performing them in advance), and test functioning	31/03

6 EQUIPMENT SELECTION:

List equipment that needs to be procured

Item	Quantity
Arduino UNO	1
Breadboard	1
Jumper Wires	-
Resistors (220 and 10K ohms)	~5
PIR/IR sensor	1
LDR sensor	1
DHT11 sensor	1
2-Channel Relay Board	1
LCD Display	1

7 FEASIBILITY STUDY

Estimate cost and time for the different options and select the best alternative for the requirement

Arduino Uno/ Arduino Nano 33 BLE Sense:

Arduino Nano 33 BLE Sense combines a powerful Arm Cortex-M4 microcontroller with onboard sensors including a microphone, color, proximity and movement - which means you can address many use cases without additional hardware or wiring. The board is also small enough to be used in end applications like wearables and the BLE Sense board over BLE or USB is an easy way to capture sensor data for your ML projects on larger computers.

However, it is scarcely available in India as of yet, having been recently released, and is priced much higher than Arduino Uno (3x the cost). So, an Arduino Uno has been chosen for the project.

A PIR sensor provides greater depth of accurate measurement of movement and is hence preferred over an IR sensor for the project. Price difference between the two exists, with the PIR being priced at INR 150 and IR sensors being priced around INR 60.

DHT22 has a greater temperature range (-40 degrees to 125 degrees) as compared to DHT11(0 degrees to 50 degrees). However, for measuring room temperature in tropical conditions, a DHT11 performs well enough and is chosen over DHT22 which is not easily available and is priced higher as well.

220 ohms and 10K ohms resistors are used as per the requirements of the components used. The other components are standard.

Costs of components:

Item	**Price**
Breadboard, Jumper Wires	269
DHT11 Sensor	199
PIR Sensor	146
LDR Sensor	171
LEDs	120
Arduino UNO	500
Relay Module	181
Resistors	91
9V Battery	150
LCD Display	205
TOTAL	**2032**

8 CONTROL SYSTEM:

Identify sensors and processing devices. Identify location of sensors and control logic to be used

Sensors:

PIR/IR – to detect presence

LDR – to detect illumination in a room

DHT11 – to detect temperature and humidity

Control logic:

Inputs – time of day, PIR input for a room, LDR input for a room, Temperature and humidity inputs for a room

Considering one light bulb and an AC unit to be controlled by the Arduino (two switches to be controlled):

PIR sensor is positioned at the door, LDR sensor near the light bulb, and DHT at a suitable position-

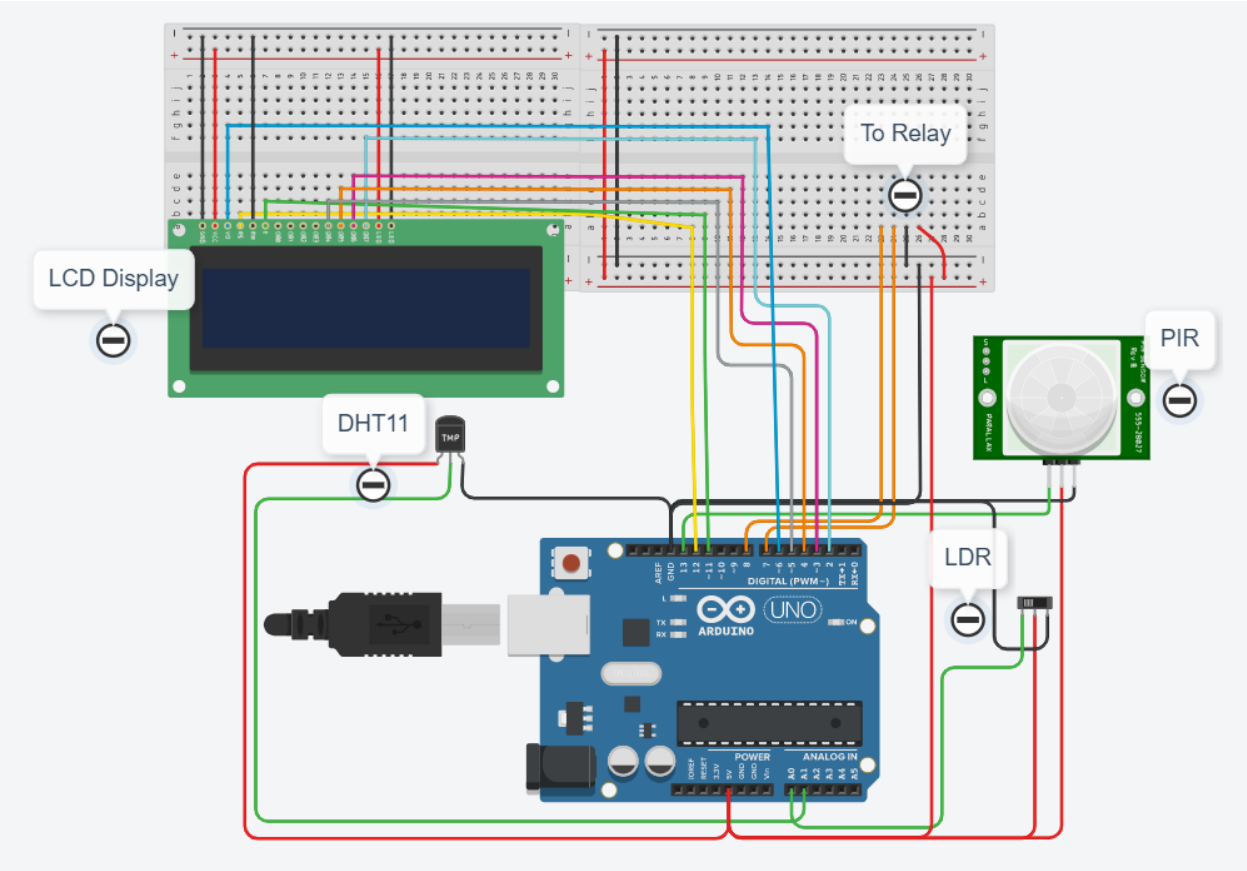
Algorithm:

1. Check for presence of people in the room (input from the PIR sensor)
2. If no one is present, keep lights switched off. Predict time of switching on the AC after arrival using the ML model. If this time is within 10 mins (configurable) of the current time, switch on the AC. Otherwise, keep it switched off.
3. If someone is present, check the time of the day – if it is past 7 pm (Configurable), turn on the lights. If it is between 4:30 pm and 7 pm, (Configurable) check LDR sensor input – if there's low light, turn on the lights. If not, keep lights switched off.
4. If someone is present, check the temperature – if it is above 30 degrees C (Configurable), turn on the AC. If not, check humidity – if it is over 75% (Configurable), turn on the AC. Otherwise, keep the AC switched off.

9 FINAL DESIGN

Give drawing of the device. Preferably 3D CAD Model, circuit diagram, PCB design, Code listing

Circuit Diagram:



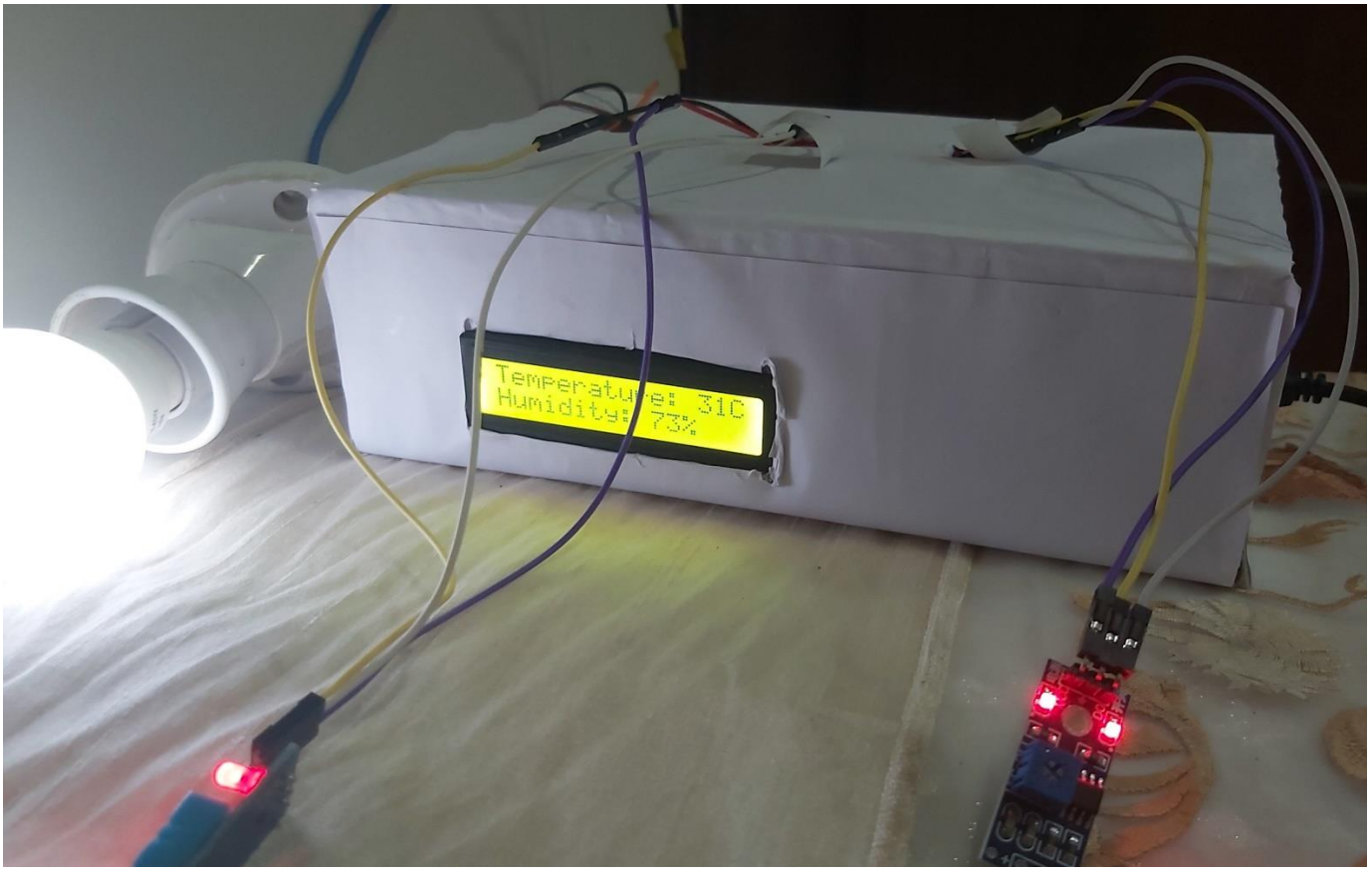
Model:

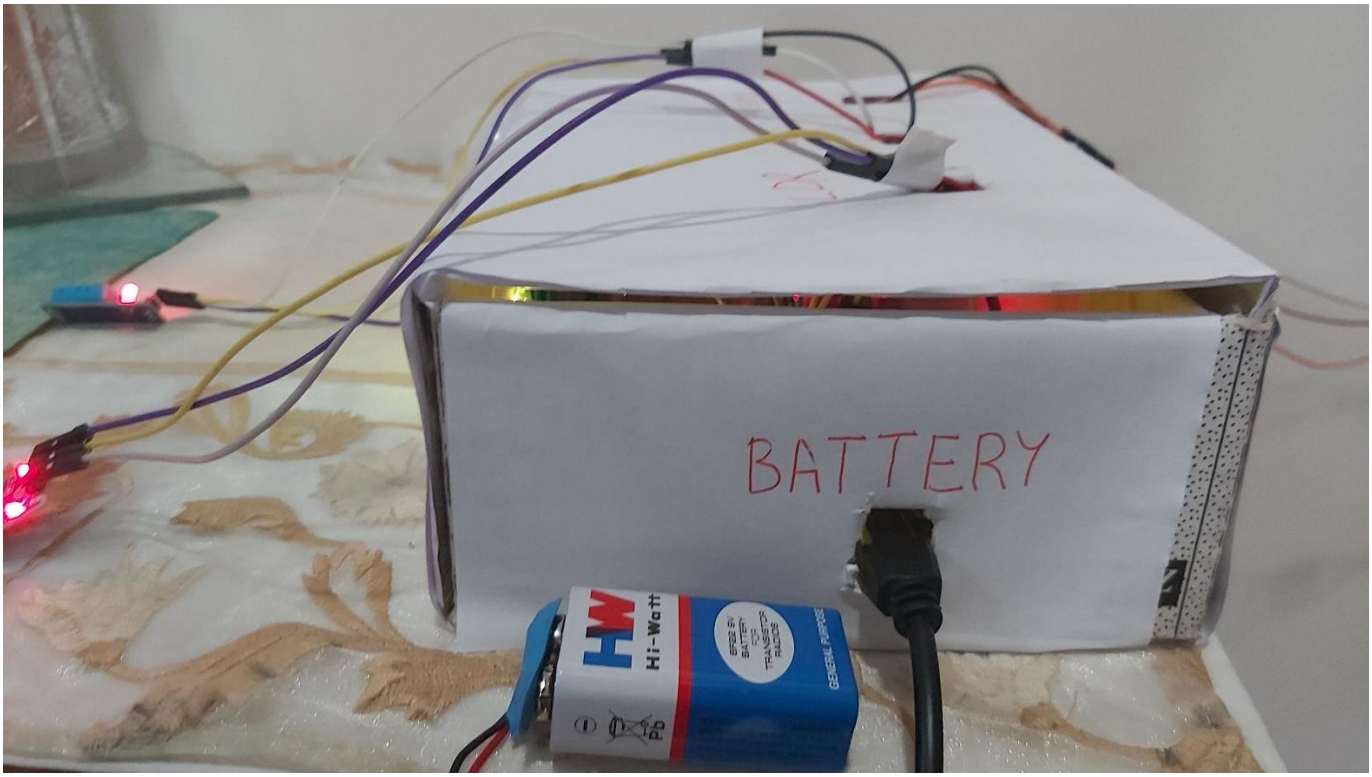
Packaging:

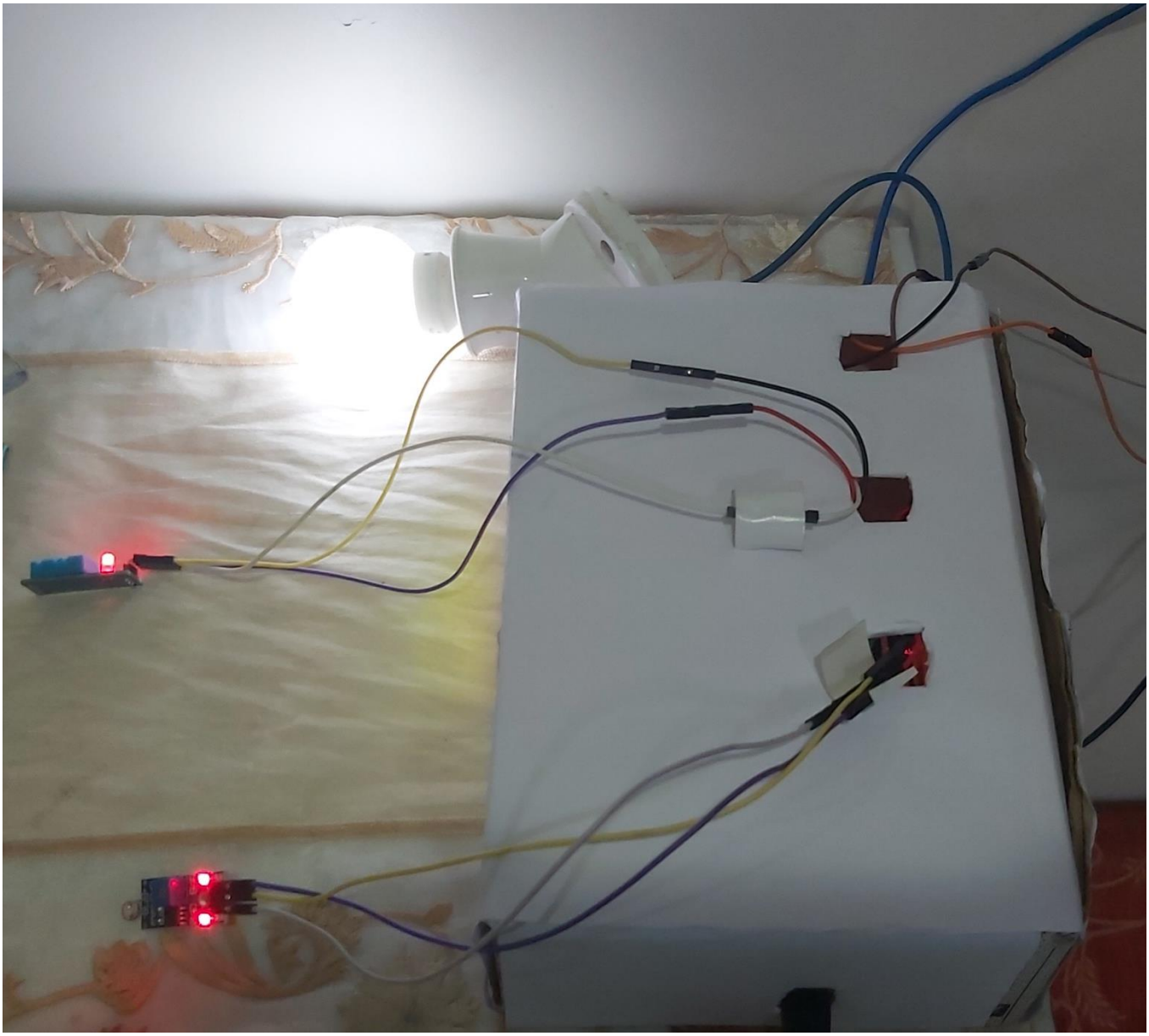
The finished product is fit in a plastic container of readily available size. The size can be further decreased by obtaining a custom-made (possibly 3D printed) container.

There are slots in the container for the three sensors, the battery, the LCD Display, and the wire inputs to the Relay. The entire thing functions as a standalone unit, and only needs to be connected to the appliance.

The product is completely modularized and individual components can be swapped out for others (e.g. Replacing sensors, batteries, appliance, etc. very easily, by just plugging in at the proper place.




BATTERY
HW
Hi-Watt



Code:

Main C code for Arduino:

Here, the parameters for switching can be set in the configurable parameters section. Hence, the functioning of the smart home switch based on various inputs is entirely customizable.

```
#include "dht.h"
#include <TimeLib.h>
#include <LiquidCrystal.h>
#include "stdarg.h"



#define LDRInputPin A0                    // LDR sensor connected here
#define DHTInputPin A1                    // DHT sensor connected here



// SET ALL PARAMETERS HERE
const int aheadMinutes = 10;              // prior starting time for
fan/AC (for Machine Learning based switching)
const int fanTemp = 30;                   // temperature at which fan
turns on
const int fanHumidity = 75;               // humidity at which fan turns
on
const int lightOnHourLDR = 17;            // time of day (hour) at which
light switching is controlled by the LDR Input
const int lightOnMinuteLDR = 30;          // time of day (minutes) at
which light switching is controlled by LDR Input
const int lightOnHour = 19;               // time of day at which light
turns on automatically
const int LDRBoundary = 60;               // bounday LDR input value -
for LDR input less than this, lights will turn on
const int hournow = 18;                    // set the current time
(hour) while starting the system for the first time
```

```
const int minutenow = 55;                 // set the current time
(minute) while starting the system for the first time

const int secondnow = 0;                  // set the current time
(second) while starting the system for the first time

const int daytoday = 11;                  // set the current date (day)
while starting the system for the first time

const int monthtoday = 12;                // set the current date
(month) while starting the system for the first time

const int yeartoday = 2021;               // set the current date
(year) while starting the system for the first time



const int PIRInputPin = 13;             // PIR Input comes here

const int LightLEDPin = 7;              // Light connected here // IN1
on the relay

const int FanLEDPin = 8;                // Fan connected here // IN2
on the relay

const int LCDContrast = 125;            // Contrast on the LCD Display



int PIRState = LOW;

int LDRState = 0;

int PIRval = 0;

int DHTHumidity = 0;

int DHTTemperature = 0;



LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

dht DHT;


// Machine Learning - Linear Regression for predicting time of arrival
```

```
#pragma once
//namespace Eloquent {
//    namespace ML {
//        namespace Port {
            class LinearRegression {
                public:
                    /**
                    * Predict class for features vector
                    */
                    float predict(float *x) {
                        return dot(x, -7.490451107479, -
1.069948671851) + 2027.519167910283;
                    }

                protected:
                    /**
                    * Compute dot product
                    */
                    float dot(float *x, ...) {
                        va_list w;
                        va_start(w, 21);
                        float dot = 0.0;

                        for (uint16_t i = 0; i < 21; i++) {
                            const float wi = va_arg(w, double);
                            dot += x[i] * wi;
                        }

                        return dot;
```

```
                    }
                };
//              }
//          }
//      }
```

```
void setup() {

  Serial.begin(9600);
  delay(100);     // Delay to let system boot

  // LCD
  analogWrite(6, LCDContrast);
  lcd.begin(16, 2);

  // Time
  int hr = hournow, mn = minutenow, sc = secondnow, dd = daytoday, mm
= monthtoday, yyyy = yeartoday;
  setTime(hr, mn, sc, dd, mm, yyyy);

  // Inputs/Outputs
  pinMode(LightLEDPin, OUTPUT);
  pinMode(FanLEDPin, OUTPUT);

  pinMode(DHTInputPin, INPUT);
  pinMode(LDRInputPin, INPUT);
  pinMode(PIRInputPin, INPUT);
```

```
  Serial.println("Sensors:\n1. DHT11 Humidity & Temperature Sensor\n2.
LDR Light Sensor\n3. PIR Presence Sensor\n\n");


}


void loop() {
    time_t t = now();
    Serial.print("The time is: ");
    Serial.print(hour(t));
    Serial.print("hours and ");
    Serial.print(minute(t));
    Serial.print("minutes\n\n");

    // PIR
    Serial.print("PIR:\n");
    PIRval = digitalRead(PIRInputPin);  // read input value

    if (PIRval == HIGH) {            // check if the input is HIGH
    //      digitalWrite(LEDPin, HIGH);  // turn LED ON
      if (PIRState == LOW) {
        // we have just turned on
        Serial.println("Entered the room!\n");
        // We only want to print on the output change, not state
        delay(5000);
        PIRState = HIGH;
      }
      else {
        Serial.print("Left the room\n\n");
        delay(5000);
```

```
    PIRState = LOW;
  }
}

if (PIRState == LOW) {
  digitalWrite(LightLEDPin, LOW);

  // ML to predict time of arrival, and set FanLEDPin accordingly

  LinearRegression lr;
  float result;
  float input[2];
  input[0] = hour(t);
  input[1] = minute(t);
  result = lr.predict(input);

  if (result - (hour(t) * 60 + minute(t)) <= aheadMinutes) {
    digitalWrite(FanLEDPin, LOW);
  }
  else {
    digitalWrite(FanLEDPin, LOW);
  }
  delay(2000);

}

else {

  // DHT
```

```
DHT.read11(DHTInputPin);
Serial.print("DHT:\n");
Serial.print("Current humidity = ");
DHTHumidity = DHT.humidity;
Serial.print(DHTHumidity);
Serial.print("%  ");
Serial.print("Temperature = ");
DHTTemperature = DHT.temperature;
Serial.print(DHTTemperature);
Serial.print("degrees C  \n\n");
delay(200);



// Print temperature and humidity on the LCD Display
lcd.setCursor(0, 0);
lcd.print("Temperature: ");
lcd.print(DHTTemperature);
lcd.print("C");
lcd.setCursor(0, 1);
lcd.print("Humidity: ");
lcd.print(DHTHumidity);
lcd.print("%");



if (DHTTemperature >= fanTemp) {
  digitalWrite(FanLEDPin, HIGH);
}

else if (DHTHumidity > fanHumidity) {
```

```
        digitalWrite(FanLEDPin, HIGH);
      }

      else {
        digitalWrite(FanLEDPin, LOW);
      }


      if (hour(t) < lightOnHourLDR || hour(t) == lightOnHourLDR &&
minute(t) <= lightOnMinuteLDR) {
        digitalWrite(LightLEDPin, LOW);
      }

      else if (hour(t) < lightOnHourLDR) {
        // LDR
        Serial.print("LDR:\n");
        LDRState = analogRead(LDRInputPin);
        Serial.print("Input: \n");
        Serial.print(LDRState);
        Serial.print("\n\n");
        delay(200);

        if (LDRState < LDRBoundary) {
          digitalWrite(LightLEDPin, HIGH);
        }
        else {
          digitalWrite(LightLEDPin, LOW);
        }
      }
```

```
      else {
        digitalWrite(LightLEDPin, HIGH);
      }


      delay(2000);                    // Wait 2 seconds before accessing
the sensors again
    }

}
```

Python ML Code for Linear Regression (Converted to corresponding C code using MicroML)

```
! pip -q install micromlgen
from micromlgen import port
from sklearn.svm import SVR
import pandas as pd
from sklearn import preprocessing
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import LinearRegression
data = pd.read_csv('/content/microml_dataset.csv')
X = data[['day', 'month', 'holiday']]
enc = OneHotEncoder(handle_unknown='ignore')
features = enc.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(features, data['arriva
l_time'], test_size = 0.25, shuffle = True, random_state = 42)
regr = LinearRegression()
regr.fit(X_train, y_train)
print(port(regr))
```

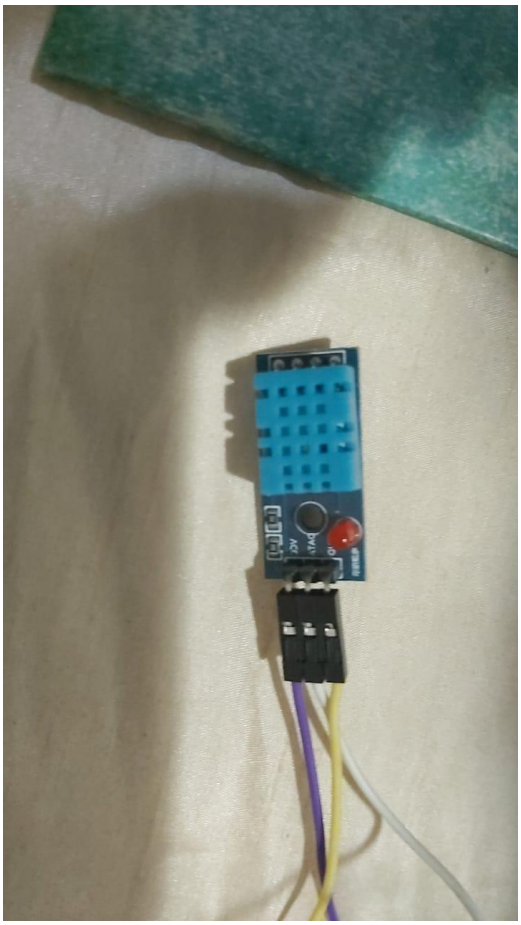
Output: (Correspondng C code):

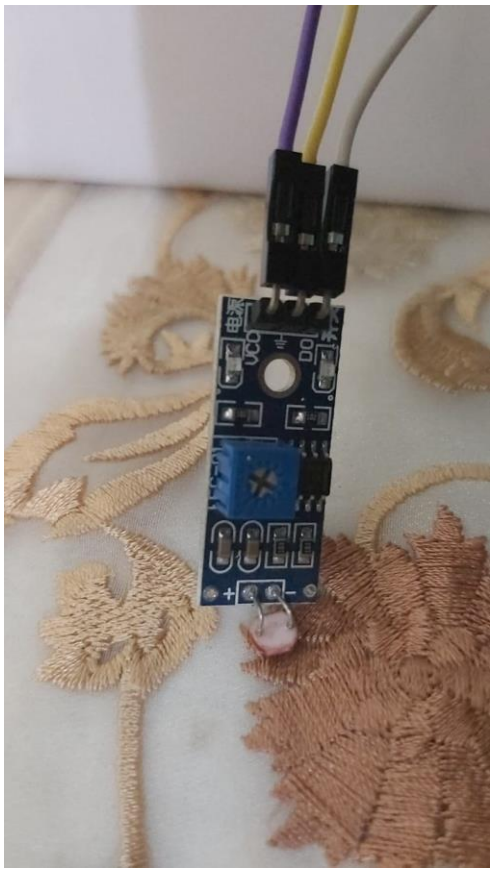
```
#pragma once
#include <cstdarg>
namespace Eloquent {
    namespace ML {
        namespace Port {
            class LinearRegression {
                public:
                    /**
                    * Predict class for features vector
                    */
                    float predict(float *x) {
                        return dot(x, -7.490451107479, -2.131955033807, -
14.864346220156, 17.857421237046, -26.076434085877, 10.428375372749,
22.277389837524, 20.985167964303, 13.878946252880, 4.158854226772,
4.060741899768, -35.252109602183, -18.319326021150, -7.546801716586,
30.012158885118, -41.273935747967, 24.133536036778, -8.490986034305,
13.653753856572, -1.069948671851, 1.069948671850) + 2027.519167910283;
                    }

                protected:
                    /**
                    * Compute dot product
                    */
                    float dot(float *x, ...) {
                        va_list w;
                        va_start(w, 21);
                        float dot = 0.0;

                        for (uint16_t i = 0; i < 21; i++) {
                            const float wi = va_arg(w, double);
                            dot += x[i] * wi;
                        }

                        return dot;
                    }
                };
            }
        }
    }
```

10 VALIDATION

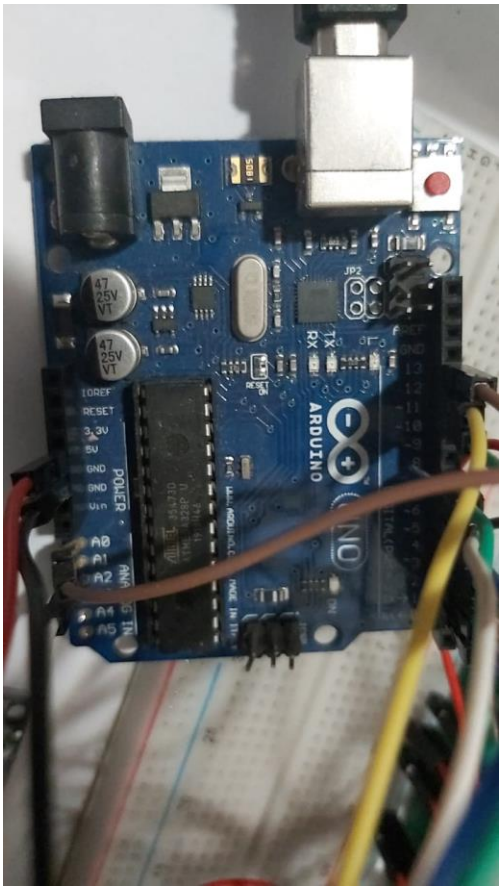
Give pictures of final part, tests done and outcome

DHT11 Sensor (Left), LDR Sensor (Right)

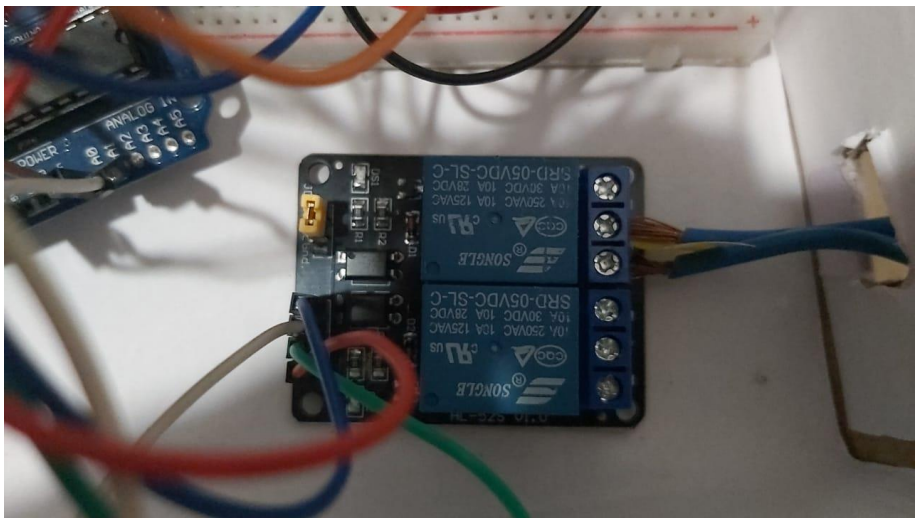




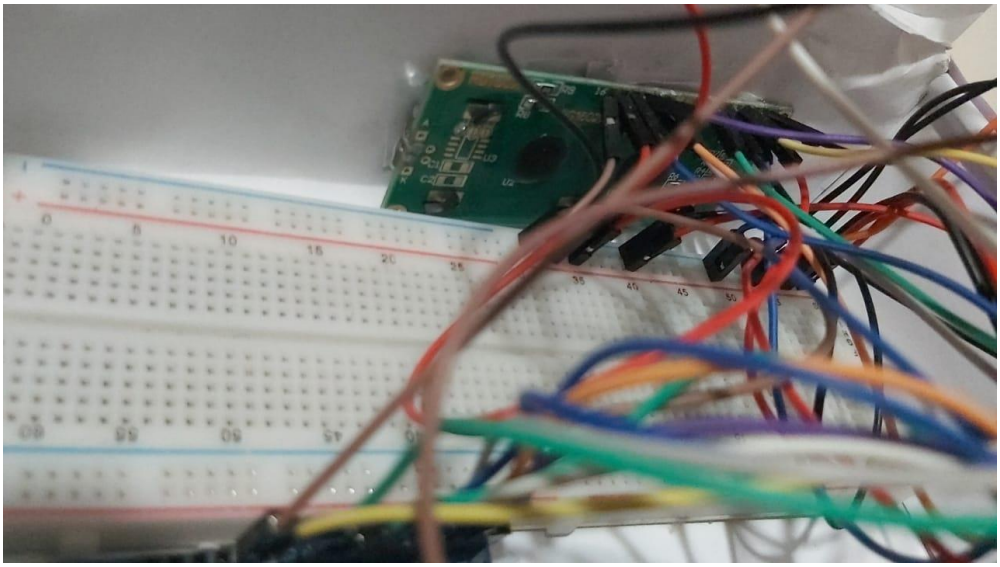
Bulb. Arduino UNO



Relay Module:

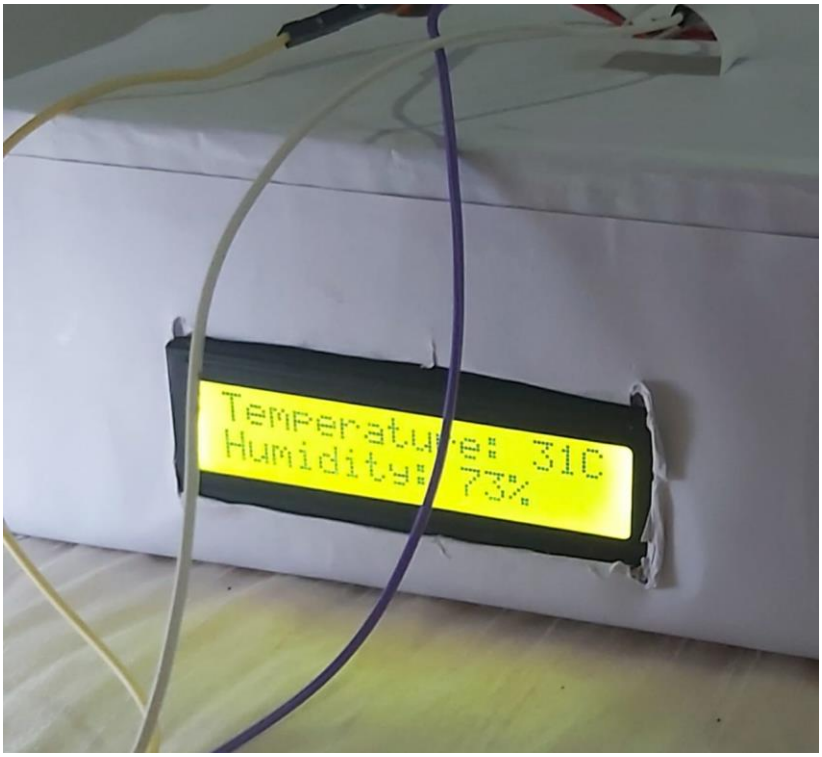


LCD Display, Breadboard



PIR Sensor

LCD Display:



Tests done –

Person enters a room – Light switches on/off correctly based on time of day and LDR Input. For simulating fan/ac, the same lightbulb was connected to the other channel of the relay. The fan/AC functioning worked just as expected, taking in the temperature and humidity data.

Noone in the room – Light stays off, correctly. Fan/AC switch turns on after a certain amount of time, predicting arrival time using ML (Linear Regression). This time varied quite a bit because the dataset generated to train the model was randomized, not based on actual data.

This is what the dataset used for ML looked like:

	A	B	C	D
1	day	month	holiday	arrival_time
2	Wed	Sep	TRUE	1951
3	Thu	Dec	FALSE	1955
4	Wed	Aug	TRUE	1944
5	Wed	Apr	TRUE	1933
6	Thu	Feb	TRUE	1932
7	Wed	Jul	TRUE	1956
8	Wed	Sep	TRUE	1959
9	Fri	Dec	FALSE	1957
10	Mon	Nov	FALSE	1927
11	Fri	Jul	TRUE	1929
12	Thu	Nov	FALSE	1933
13	Mon	Jan	FALSE	1926
14	Tue	Dec	FALSE	1907
15	Tue	Feb	FALSE	1943

Dataset created for ML (Linear Regression Model) – features are day, month, holiday, and the true output is the arrival time (generated randomly)

11 Conclusion and Discussion

Give details of your learning and scope of any further work.

I learnt a lot about the functioning of all the components used, and the challenges involved in building a ready electronics product. Actually working on the real sensors and equipment is vastly different from working on a simulator, and there are numerous more challenges involved in working on actual components. Also learnt about the scope of running ML on Arduino, given the limited hardware capacities. All in all, it was a wholesome learning experience.

Future scope:

The finished product is placed in a plastic container of readily available size. The size can be further decreased by obtaining a custom-made (possibly 3D printed) container.

A breadboard is used for the wiring, due to inconveniences in learning soldering remotely without guidance. A PCB can definitely be used for a more robust and compact product.