

**NAAN MUDHALVAN SKILL DEVELOPMENT PROGRAM**

**NM1075 – EBPL SOFTWARE FOR PRODUCT**

### A SKILL TRAINING REPORT

***Submitted by:***

**THANGA ABARNA.M**

**950623104102**

**SUREKA.M**

**950623104098**

in recognition of the completion of the Skill Training course as part of the curriculum

## during fourth semester

## BACHELOR OF ENGINEERING

## IN

**COMPUTER SCIENCE AND ENGINEERING**

**EINSTEIN COLLEGE OF ENGINEERING,**

**TIRUNELVELI - 12**

**ANNA UNIVERSITY : CHENNAI – 600 025**

MAY – 2025

**EINSTEIN**

**“Science is the Religion for all”**

**COLLEGE OF ENGINEERING**

Sir.C.V Raman Nagar, Tirunelveli-627012

This is to certify that, this skill training Report of work done by Mr./Ms…………….……..……………...………………Reg.No…………………….………..…. of **B.E. Computer Science and Engineering** for **NM1075 – EBPL SOFTWARE FOR PRODUCT** during the academic year **2024-2025**.

### **Staff-in-charge Head of the Department**

Submitted for University Examination held on…….…………..……………..

### **INTERNAL EXAMINER EXTERNAL EXAMINER**

***SENSOR BASED HOME AUTOMATION SYSTEM***

# *ABSTRACT*

This project presents a system that automatically controls home appliances using sensors (LDR, DHT11) based on environmental conditions. It is designed for energy saving and convenience. The system operates without manual intervention by using an intelligent controller to manage devices like fans and lights.

# *OBJECTIVES*

- To control appliances automatically based on sensor input.

- To improve energy efficiency.

- To create a smart environment with modern home conveniences.

# *COMPONENTS USED*

|  |  |
| --- | --- |
| Component | Quantity |
| Arduino UNO / NodeMCU | 1 |
| LDR Sensor | 1 |
| DHT11 Sensor | 1 |
| Relay Module | 2 |
| LED Bulb / Fan (Demo Units) | 2 |
| Jumper Wires, Breadboard | - |

# *WORKING*

The LDR sensor measures ambient light. If the light level is low, the system automatically turns ON the light. The DHT11 sensor monitors room temperature. If the temperature exceeds 30°C, the fan is turned ON. All these operations are automatic and do not require human intervention.

# *MICROCONTROLLER CODE SUMMARY*

• Analog input (LDR) → LOW light → Light ON

• Digital input (DHT11) → HIGH temp → Fan ON

***CODING***

#include <Adafruit\_GFX.h>

#include <Adafruit\_ILI9341.h>

#include <Adafruit\_FT6206.h>

#include <SPI.h>

#include <Wire.h>

#include <Servo.h>

#include <DHT.h>

#include <LiquidCrystal\_I2C.h>

#define TFT\_CS   10

#define TFT\_DC   9

#define TFT\_RST  8

#define DHTTYPE DHT22

#define DHTPIN 5

DHT dht(DHTPIN, DHTTYPE);

Adafruit\_ILI9341 tft = Adafruit\_ILI9341(TFT\_CS, TFT\_DC, TFT\_RST);

Adafruit\_FT6206 touch = Adafruit\_FT6206();

bool doorState = true;

bool lightState = true;

Servo myServo;

#define TRIG\_PIN 6

#define ECHO\_PIN 7

long duration;

float distance;

float prevDistance;

bool motionState = false;

float prevTemperature, currentTemperature;

LiquidCrystal\_I2C lcd(0x27, 16, 2);

#define MQ2\_PIN A0

int threshold = 400;

bool blinkerState = false;

void setup() {

**Serial**.begin(9600);

  tft.begin();

  tft.setRotation(0);

  tft.fillScreen(ILI9341\_WHITE);

  if (!touch.begin(10)) {

**Serial**.println("Touch screen not found");

    while (1);

  }

  introScreen();

  displayMain();

  myServo.attach(3);

  doorController();

  pinMode(TRIG\_PIN, OUTPUT);

  pinMode(ECHO\_PIN, INPUT);

  pinMode(4, OUTPUT);

  pinMode(2, INPUT);

  dht.begin();

  lcd.begin(16,2);

  lcd.init();

  lcd.backlight();

  pinMode(12, OUTPUT);

}

void loop() {

  int gasValue = analogRead(MQ2\_PIN);

  if (gasValue > threshold) {

    blinkerState = true;

  } else {

    blinkerState = false;

  }

  blinker();

  currentTemperature = dht.readTemperature();

  if (isnan(currentTemperature)) {

**Serial**.println("Failed to read from DHT sensor!");

    return;

  }

  acController();

  digitalWrite(TRIG\_PIN, LOW);

  delayMicroseconds(2);

  digitalWrite(TRIG\_PIN, HIGH);

  delayMicroseconds(10);

  digitalWrite(TRIG\_PIN, LOW);

  duration = pulseIn(ECHO\_PIN, HIGH);

  distance = duration \* 0.034 / 2;

  doorController();

  int motionDetected = digitalRead(2);

  if (motionDetected == HIGH && !motionState) {

    motionState = true;

    lightState = false;

    digitalWrite(4, HIGH);

    toggleLight();

  } else if(motionDetected == LOW && motionState) {

    motionState = false;

    lightState = true;

    digitalWrite(4, LOW);

    toggleLight();

  }

  prevTemperature = currentTemperature;

  prevDistance = distance;

  delay(200);

}

void blinker() {

  if(blinkerState) {

    digitalWrite(12, HIGH);

    delay(200);

    digitalWrite(12, LOW);

  }

}

void acController() {

  if(prevTemperature == currentTemperature) return;

  toggleAC();

  if (currentTemperature > 30.0) {

    lcd.clear();

    lcd.setCursor(0,0);

    lcd.print("AIR CONDITIONER");

    lcd.setCursor(7, 1);

    lcd.print("ON");

  } else {

    lcd.clear();

    lcd.setCursor(0,0);

    lcd.print("AIR CONDITIONER");

    lcd.setCursor(7, 1);

    lcd.print("OFF");

  }

}

void doorController() {

  if(distance == prevDistance) return;

  if(distance <= 100) {

    doorState = false;

    toggleDoor();

  }

  else {

    doorState = true;

    toggleDoor();

  }

  if(doorState) {

    myServo.write(0);

  }

  else {

    myServo.write(90);

  }

}

void displayMain() {

  tft.fillScreen(ILI9341\_WHITE);

  tft.fillRect(0, 0, 240, 30, ILI9341\_BLUE);

  tft.setTextSize(2);

  tft.setTextColor(ILI9341\_WHITE);

  tft.setCursor(90, 10);

  tft.print("Menu");

  tft.setCursor(10, 50);

  tft.setTextColor(ILI9341\_BLACK);

  tft.print("Door: ");

  toggleDoor();

  tft.setCursor(10, 100);

  tft.setTextColor(ILI9341\_BLACK);

  tft.print("Light: ");

  toggleLight();

  tft.setCursor(10, 150);

  tft.setTextColor(ILI9341\_BLACK);

  tft.print("Temperature: ");

  toggleAC();

}

void toggleAC() {

  tft.setCursor(155, 150);

  tft.setTextColor(ILI9341\_BLACK, ILI9341\_WHITE);

  tft.print(currentTemperature);

}

void toggleDoor() {

  if(doorState) {

    tft.fillRect(110, 45, 100, 25, ILI9341\_RED);

    tft.setTextColor(ILI9341\_BLACK);

    tft.setCursor(135, 50);

    tft.print("CLOSE");

  }

  else {

    tft.fillRect(110, 45, 100, 25, ILI9341\_GREEN);

    tft.setTextColor(ILI9341\_BLACK);

    tft.setCursor(135, 50);

    tft.print("OPEN");

  }

}

void toggleLight() {

  if(lightState) {

    tft.fillRect(110, 95, 100, 25, ILI9341\_RED);

    tft.setTextColor(ILI9341\_BLACK);

    tft.setCursor(145, 100);

    tft.print("OFF");

  }

  else {

    tft.fillRect(110, 95, 100, 25, ILI9341\_GREEN);

    tft.setTextColor(ILI9341\_BLACK);

    tft.setCursor(145, 100);

    tft.print("ON");

  }

}

void introScreen() {

  tft.fillRect(0, 0, 240, 50, ILI9341\_BLUE);

  tft.setTextSize(2);

  tft.setTextColor(ILI9341\_WHITE);

  tft.setCursor(10, 10);

  tft.print("Welcome to Home");

  tft.setCursor(10, 30);

  tft.print("Automation System");

  tft.setTextSize(2);

  tft.setTextColor(ILI9341\_BLACK, ILI9341\_WHITE);

  tft.setCursor(10, 160);

  for (int i = 10; i <= 100; i += 20) {

    tft.setCursor(10, 100);

    tft.print("Loading ");

    tft.print(i);

    tft.print("%   ");

    delay(300);

  }

}

# *OUTPUT*

# 

# *ADVANTAGES*

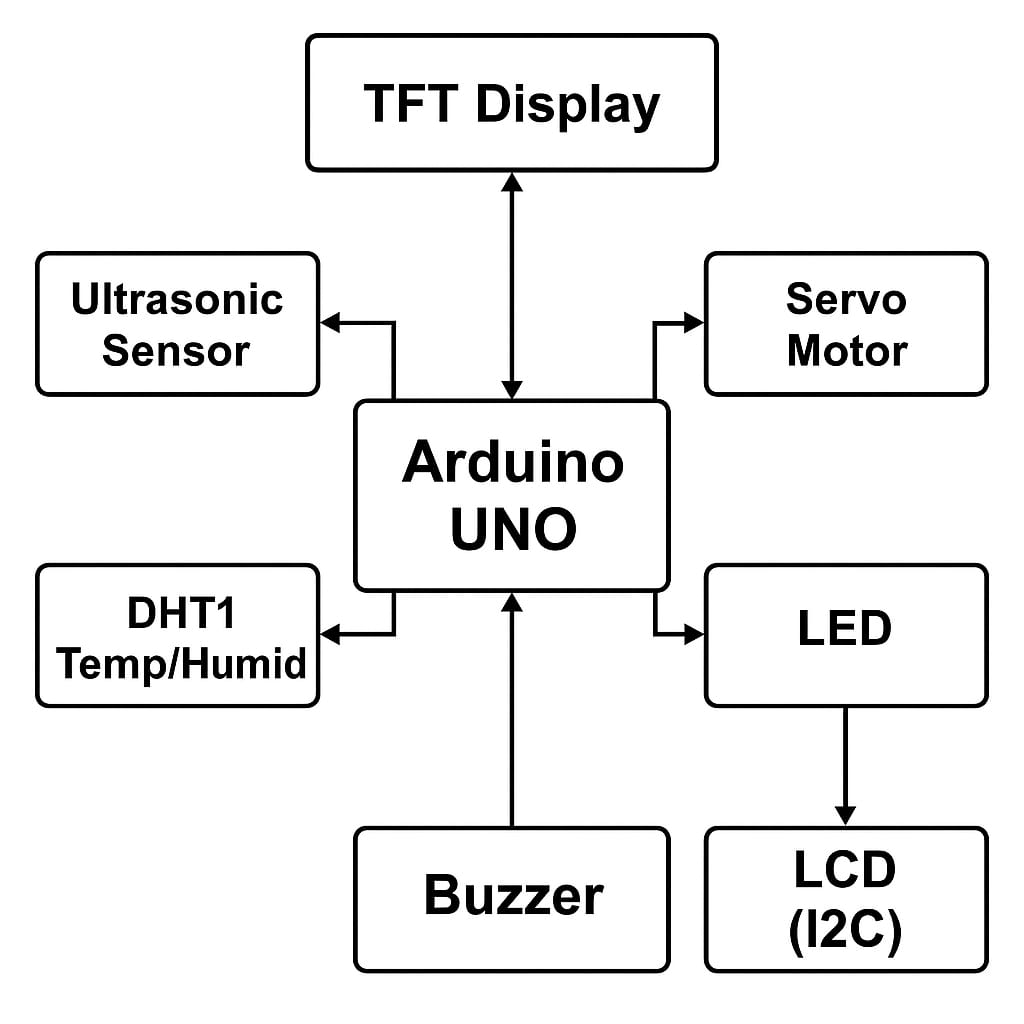
• Energy efficient

• No need for manual switch

• Can be extended to a complete smart home

• Low cost and easy to implement

# *BLOCK DIAGRAM*



# *BLOCK DIAGRAM EXPLAINATION*

Sensors (LDR, DHT11) provide environmental data. The microcontroller (Arduino UNO / NodeMCU) processes the data and triggers the relay module. The relay modules then control the light and fan accordingly. Optionally, a PIR sensor can be used for motion detection.

# *CONCLUSION*

This project serves as a great starting point for smart home applications. By using sensors to operate devices automatically based on environment, it creates a convenient and safe atmosphere. In the future, this system can be enhanced with IoT and AI for more intelligent automation.

**SUBITTED BY:**

**M.THANGA ABARNA**

**(950623104102),**

**M.SUREKA**

**(950623104098)**