CUSTOMIZATION OF LIGHTING AND VENTILATION SYSTEM IN HOUSEHOLD

U18CSP8701 - PROJECT PHASE 2 REPORT

Submitted by

ISWARYA M	**19BCS043**
ANJU A	**19BCS109**
PRASANTH P	**19BCS117**

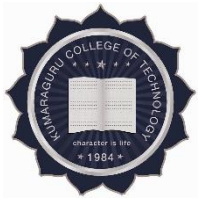
in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING



KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution affiliated to Anna University, Chennai)

Post Box No: 2034, Coimbatore - 641049

APRIL 2023

CUSTOMIZATION OF LIGHTING AND VENTILATION SYSTEM IN HOUSEHOLD

A PROJECT REPORT

Submitted by

ISWARYA M (19BCS043)

ANJU A (19BCS109)

PRASANTH P (19BCS117)

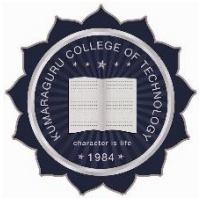
in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING



KUMARAGURU COLLEGE OF TECHNOLOGY
(An Autonomous Institution affiliated to Anna University, Chennai)
Post Box No: 2034, Coimbatore - 641049

APRIL 2023

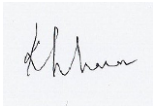
KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE – 641049.

COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"CUSTOMIZATION OF LIGHTING AND VENTILATION SYSTEM IN HOUSEHOLD"**

is the bonafide work of "**ISWARYA M (19BCS043), ANJU A (19BCS109), PRASANTH P (19BCS117)** who carried out the project work under my supervision.



SIGNATURE

Dr. Devaki P
HEAD OF THE DEPARTMENT

Professor & HOD

Department of Computer Science and Engineering

Kumaraguru College of Technology

Coimbatore – 641 049.

SIGNATURE

Dr. Baskaran K R
SUPERVISOR

Professor

Department of Computer Science and Engineering

Kumaraguru College of Technology

Coimbatore – 641 049.

Submitted for the project viva-voce examination held on ----------

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DECLARATION

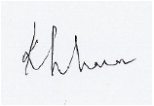
We affirm that the project work titled “CUSTOMIZATION OF LIGHTING AND VENTILATION SYSTEM IN HOUSEHOLD” being submitted in partial fulfillment for the award of B.E Computer Science and Engineering is the original work carried out by us. It has not formed the part of any other project work submitted for the award of any degree or diploma, either in this or any other University.

ISWARYA M (19BCS043)

ANJU A (19BCS109)

PRASANTH P (19BCS117)

I certify that the declaration made above by the candidates is true.



Dr. Baskaran K R
Professor,
Department of Computer Science and Engineering,
Kumaraguru College of Technology,
Coimbatore – 641049.

ACKNOWLEDGEMENT

We express our profound gratitude to the Management of Kumaraguru College of Technology for providing us with the required infrastructure that enabled us to successfully complete the project.

We extend our gratitude to our Principal, **Dr. D. Saravanan,** for providing us the necessary facilities to pursue the project.

We would like to acknowledge **Dr. P. Devaki,** Professor and Head, Department of Computer Science and Engineering, for her support and encouragement throughout this project.

We thank our project coordinator **Dr. L. Latha,** Professor, and guide **Dr. K. R. Baskaran,** Professor, Department of Computer Science and Engineering, for the constant and continuous effort, guidance, and valuable time.

Our sincere and hearty thanks to staff members of Department of Computer Science and Engineering of Kumaraguru College of Technology for their well wishes, timely help and support rendered to us during our project. We are greatly indebted to our family, relatives, and friends, without whom life would have not been shaped to this level.

- ISWARYA M

ANJU A

PRASANTH P

ABSTRACT

The world is in the fourth industrial revolution and the twenty-first century. Because to the expanding and improving technologies, modern humans are living more comfortably. People have seen miracles in their lives thanks to the Internet of Things, machine learning, artificial intelligence, cloud computing, and many more technologies. These inventions and ideas resulting from the technologies have put the future in our hands. These benefit those with special needs, the elderly who require assistance, independent learners, and many more. These technologies are essential in a variety of industries, including engineering, agriculture, education, finance, and others. The world is currently experiencing an era of unstoppable technological advancement and growth. IoT, or the Internet of Things, is important in this. One crucial component of IoT is smart home (automation of home appliances). This article presents a model where lighting and ventilation are automatically adjusted based on the surrounding environment and the user's preferences. The first thing to be done is to create a hardware model, from which the information is gathered about the user's preferences as well as the surrounding environment from human control and sensor readings. The ESP8266 wi-fi module receives the trained and tested data which are trained using machine learning classifier algorithms, which is subsequently supplied to the system for autoregulation.

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION AND BACKGROUND

The Internet of Things is one of the key concepts for making the "smart city" notion a reality. Numerous sectors, including agriculture, retail, health, and transportation, use it. The IoT's most popular uses are for smart grids and home automation. So, with the aid of IoT, smart devices may be created and used to address problems and challenges that arise in the real world. IoT has become one of the most significant components of people's daily lives as a result of the use of devices like Siri, Alexa, Google Home, and other home IoT-enabled gadgets. Furthermore, in our modern typical workplace culture, working from home has become a widespread practice. World is moving farther than one could ever imagine possible thanks to developments in the Internet of Things.

One of the fundamental components of IoT is home automation. Years ago, the idea of home automation was unimaginable. It is the automatic control of home appliances. The fan and light switch on when people enter the room and turn off when they leave. Additionally, one can control their appliances from anywhere in the world by using voice control systems or other web, mobile, or other sorts of apps. Home automation has a lot of benefits, such as security, comfort, and energy savings.

Machine learning is often used for forecasting the loads and comparing the normal usage and the load usage when it is automated. The next development in home automation will be the integration of ML capabilities into smart home settings. modern scenes are typically built using straightforward if-then-else logic over triggers and actions. While increasing automation, this deterministic logic offers very little intelligence. Simple rules, for instance, do not consider how user routines and behaviors have evolved.

By enabling scenarios that learn the tenants' behavioral patterns and utilize them to optimize smart home operations, ML removes these restrictions. In this direction, historical information about user behavior in their home environment can be used to train machine learning algorithms.

The idea of customized fan and light automation is presented in this work. According to the idea, the fan speed and light intensity are automatically controlled based on data gathered over time about user preferences. It is an example of a way of tailored automation using the Node MCU ESP8266 wi-fi module as a microcontroller to collect user preferences through manual regulation and the environment's temperature, humidity, and light levels from the sensors installed. We create a machine learning model, train it, test it, and then use the results to automatically operate the fan and light after data gathering. The hardware elements and the developed model will be discussed first. Then, the project will go over the information gathered and the developed machine learning model.

1.2 OBJECTIVE

This project's primary goal is to automatically adjust the fan speed and light intensity to suit user and environmental preferences. This helps people to concentrate on their chores rather than dealing with the regulation of fan and light.

1.3 PROBLEM STATEMENT

Providing a customized system of automated lighting and ventilation, using Internet of Things and Machine learning for people’s comfort and convenience.

1.4 PROJECT OVERVIEW

First, an Internet of Things (IoT) model needs to be developed so that,

depending on the temperature and ambient lighting, one can independently control the fan's speed and light's brightness during the day and at night. The data for those specified speeds and lights is then compiled. The data is then used to develop a machine learning model, which is then trained and tested before being used further. The trained model will be sent to the Node MCU, which will use it to use it to automate the fan and light as needed.

PROJECT MODULES:

- Literature review and proposed model explanation
- Flow diagram of proposed model
- Completion of hardware model
- Creating a machine learning model for training sensors
- Final prototype of customized automatic regulation of fan and light

CHAPTER 2

LITERATURE SURVEY

2.1 EXISTING SYSTEM

Mansour H. Assaf, et al. [1] proposed, "Sensor based Home Automation and Security System". This study examined the creation of a home safety and surveillance system in the light of the shortcomings of conventional security systems, which primarily seek to prevent burglaries and gather proof of trespassing. This paper presented the design and implementation details of a home control and security system based on field programmable gate arrays (FPGA). The user could interact with the system directly via an online web interface, and a user-friendly web page can be used to remotely control home appliances including air conditioners, lights, door locks, and gates. The homeowner would receive a warning email message immediately in the case of a breach because the system could monitor entrance points like doors and windows. This feature further improves the security component of the system.

Fathia Checkered, et al. [2] have proposed, "Low-Cost Automation System for Smart Houses based on PIC Microcontrollers". The research presented described the development of an automation system for smart solar homes with microcontrollers that includes automatic lighting, thermal comfort sensors (temperature, humidity), and security features. (gas leak, smoke detection). The application of this PIC microcontroller-based technology entails running the complete algorithm. The test software employed Proteus, which is a free online resource, to create the control circuit and simulation, and the programme was written in Micro C. Temperature and air quality were controlled by a PIC16F877A, and lighting was managed by a PIC18F4550.

Fathima Deena P.P, et al. [3] proposed, "IoT based smart street light management system". To replace the power-consuming standard HID lamps, the

project employed Light Emitting Diodes (LED) that consume very little electricity. The intensity could be changed with LED lights and LDR, which was not possible with HID bulbs. Since LEDs were directional light sources, their ability to generate light in a specified direction enhances the effectiveness of streetlights. An extra DHT11 Temperature-Humidity sensor was part of this system. This information gave the precise temperature and humidity of a location. The DHT11 is a hybrid sensor that produces calibrated digital data for both temperature and humidity. It ensures excellent reliability and exceptional long-term stability. An Arduino board that had been configured to produce the necessary light intensity at different intervals was used for this technique.

John Jaihar, et al. [4] have proposed, "Smart home automation using machine learning algorithms". When the machine learning model recognizes the user's emotion, it changes the color of the ambient lighting to match the different color profiles available for that particular facial expression. If anger is the detected emotion, the machine learning model can help the user by turning on the fan. A user interface has been created to control the machine in special situations. The user interface can also be accessed over the Internet. The temperature of DHT-11 is not only displayed in the user interface, but also recorded in the database. The fan can automatically change speed based on this.

Suraj Kaushik, et al. [5] have proposed, "Automatic fan speed control using temperature and fan speed control using Arduino". According to their study. the fan's speed would automatically change based on the environment. DHT22 sensors were used to regulate the temperature. The temperature was sensed using a DHT22 sensor, and the fan speed was then adjusted appropriately using PWM (Pulse with Modulation). Data from the DHT22 sensor was subsequently transmitted to the Arduino-based micro-controller circuit. After the sensor had detected the temperature and the Pulse Width Modulation and Arduino board have been utilized

to control the fan speed, the Lcd would display the temperature and speed of the fan.

Md. Sadad Mahamud, et al. [6] have proposed, “Domicile - An IoT Based Smart Home Automation System”. This study suggested a low-cost smart home automation system based on the Internet of Things. The Arduino UNO microcontroller and the ESP32 Wi-Fi module formed the foundation of the entire system. The microcontroller performed the majority of the controlling; all data were first fetched, then decoded, and lastly Arduino executes is mentioned execution cycle. The ESP32 Wi-Fi module was used to control communication. When the register author made any changes or updates, the ESP32 module updated Arduino via direct connection with the private server.

Shives, et al. [7] proposed, “Speed control of ceiling fan using PWM technique”. In this work, they suggested a novel way to regulate ceiling fan speed in response to the room's temperature and population. Each time, the right temperature and ultrasonic sensors were chosen to measure the room's temperature and the number of occupants. Fan speed was controlled by a PIC16F877A microcontroller and optocoupler. Room temperature, engine speed and number of people in the room are all displayed on the LCD panel. When the human body entered the room, the fan would turn on and the ultrasonic sensor would detect the human body. A temperature sensor detected the ambient temperature and changed the fan speed accordingly. The fan ran faster in warmer temperatures and slower in colder temperatures. This speed control technology is cost effective, energy efficient and requires less user effort.

Francis Jamar P. Montalba, et al. [8] proposed, “An IoT smart lighting system for university classrooms”. In this investigation, a passive infrared sensor to determine whether the room is inhabited, a relay switch to manage the lighting switching, and an integrated passive infrared sensor were all used. A controller was developed by a researcher using AJAX allows controller devices to run both web

and mobile applications online. The developed application serves as a controller, a management system that develops the necessary timetables, and a monitoring system that shows the number of hours of lights and the overall amount of energy used. Administrators might change the schedule to turn on the lights at specified times if students were present. Even with a set timetable, the lights wouldn't come on when no one was in the room, saving you from wasting electricity. You could still use the application to control the lights over a different remote network after the predetermined schedule has ended.

Naser Hossein Motlagh, et al. [9] have proposed, “An IoT-based Automation System for Older Homes: A Use Case for Lighting System”. This study offered an Internet-enabled control mechanism and discussed the possible gains from such IoT deployments for older homes. In addition, they showed how IoT could help a house use less energy while still keeping its people comfortable. We developed a test-bed for an IoT platform and utilized light sensors to monitor the indoor ambient light levels. The control system for the smart lighting system is then established. This system used an IoT gateway to regulate and control the dimmable light sources. They have modelled household smart system power consumption (SSPC) and conventional system power consumption (CSPC), which are both based on their control strategy, to compare the two. The SSPC study's findings suggested a few potential benefits of installing IoT-enabled control systems in older houses. In this case, energy savings are made while still providing the occupants with a pleasant visual experience.

Nabil Ouerhani, et al. [10] have proposed, “IoT-Based Dynamic Street Light Control for Smart Cities Use Cases”. This paper described a functional dynamic street light management system powered by the Internet of Things. The current system bases its estimation of the required intensity of street lighting on traffic and environmental variables. Following that, it adjusted the luminaires

accordingly. Real street light luminaires installed on one street in a Swiss city were used in quantitative tests on real-world scenarios. These tests unequivocally demonstrated the solution's applicability. The total energy saved while utilizing their method is calculated and comes to 56% for the case under consideration. A qualitative study in humans was also performed but did not show any significant change in how humans perceive the safety of light intensity dynamics. This supported the idea that such technologies could eventually aid in the growth of low-carbon cities.

Pritam Roy, et al. [11] have proposed, “Microcontroller based automated room light and fan controller”. The microcontroller in this study served as the processing component. It first detects the temperature before comparing the data to the set temperature. If the real temperature is higher than the set temperature, the controller activates the fan and adjusts its speed proportionally to the difference between the set temperature and the current temperature. If the outside temperature falls below the set threshold, the fan would be turned OFF. The fan's speed would vary depending on the temperature. Additionally, a proximity sensor would be included to detect whether anybody is close to the fan; if so, the fan would remain on. If not, it would not. A laser obstacle detector could be installed on a room's doors to detect when someone had entered the space. When someone entered the room, devices like fans and lights turn on; when no one was present, they turn off. The result was significant energy savings.

Mustafa Saad, et al. [12] have proposed, “Automatic Fan speed control system using microcontroller”. This study proposed an automatic control method to adjust the fan speed. A circuit was designed and built using an LM35DZ temperature sensor, a PIC16F877A microcontroller, a brushless DC motor, and other electronic components to automatically adjust the fan speed. As an added feature, temperature and fan speed are displayed on the LCD.

Satyendra K. Vishwakarma, et al. [13] have proposed, “Smart Energy Efficient Home Automation System Using IoT”. This study proposes a sophisticated, energy-efficient home automation system to enable remote access to and administration of household devices. A web-accessible Internet connectivity module is attached to the home system's primary supply unit. A static IP address is utilised while establishing a wireless connection. The foundation of home automation is a multimodal application that may be operated via a web-based tool or by voice commands delivered to the Google Assistant by the user. With the aid of the design control unit, IoT might be used to turn a home appliance into a smart, intelligent device. To show the operation of the suggested notion through experimentation, the three lights were linked.

Dr. Suchart Yammen, et al. [14] have proposed, “IoT based speed control of Smart Fan”. This article concentrated on utilizing Wi-Fi Direct to manage fan speed and includes detailed configuration details, such as client ID, server name, port number, username, password, and subject, as well as fan speed level. Laptops, tablets, and other mobile devices may also be utilized for monitoring and management. The proposed method has the unique ability to regulate the fan using a variety of intelligent devices. A control algorithm was created to control the speed of the fan motor. The user may change the fan speed by communicating the setting value (speed input command) via smartphone, which was this method's key benefit. This made it possible for you to successfully control the engine using Wi-Fi Direct whenever you had a reliable internet connection.

2.2 DRAWBACKS OF EXISTING SYSTEM

The early system's biggest flaw was that it was expensive to put into place. Additionally, smart regulation is only focused on the environment, not the individuals using the appliances. The use of a remote is another viable option for this, but keeping it in your possession or close by while performing other tasks is a challenge. Also, the availability of internet is a major factor in many of the proposals.

CHAPTER 3

SYSTEM REQUIREMENTS

3.1 HARDWARE SPECIFICATIONS

3.1.1 NODEMCU (ESP8266)

Figure 3.1 Node MCU (ESP8266)

32 general purpose registers that are each directly coupled to the arithmetic logic unit make up the Atmel AVR core, which also has a robust instruction set. With 32K bytes of inherently programmable flash, 1K bytes of EEPROM, 2K bytes of SRAM, and 23 general-purpose I/O lines, the ATmega328/P enables a lightning-fast boot-up while using very little power. increase. For adding capacitive touch button and wheel capabilities to AVR microcontrollers and sliders, Atmel provides the Touch® library.

Pin diagram of ATMEGA328

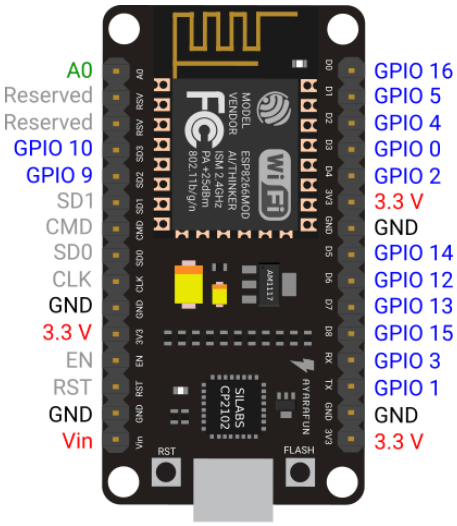


Figure 3.2 PIN Diagram of ESP8266 Wi-Fi module

3.1.2 TEMPERATURE AND HUMIDITY SENSOR

The digital sensor DHT11 measures humidity and temperature. To detect humidity and temperature , simply connect this sensor to any microcontroller, including Arduino, Raspberry Pi. This sensor can be distinguished from other modules by an ON LED. The DHT11 is one relative humidity sensor. A thermistor and a capacitive humidity sensor are used by humidity sensors to measure the ambient air. The DHT11 sensor comprises of a capacitive humidity sensor and a thermistor for measuring temperature. A moisture-sensitive capacitor is created by separating two electrodes by a substrate that might employ moisture as a dielectric. The capacitance value varies with the humidity. The changing resistance value is computed, interpreted, and converted to digital form by the IC.

Resistance drops as temperature rises because the sensor employs a thermistor with a negative temperature coefficient to monitor temperature. Typically, semiconducting ceramics or polymers are used in the construction of this

sensor to achieve greater resistance values with little temperature variation. The DHT11 is capable of measuring humidity from 20% to 80% with an accuracy of 5% and temperature from 0°C to 50°C with a precision of 2.0°C. The sampling frequency of DHT11 is 1 Hz. As a result, you might receive fresh data every second.

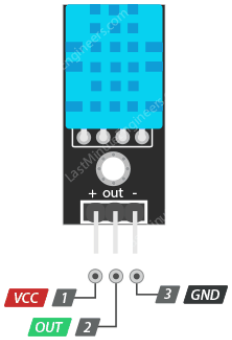


Figure 3.3 DHT11 sensor

The sensor is powered by the +Vcc pin, connected to the Arduino's ground by the -GND pin, and communicating with the Arduino via the OUT pin. This module should be able to function without any additional components because it includes all necessary supporting circuitry.

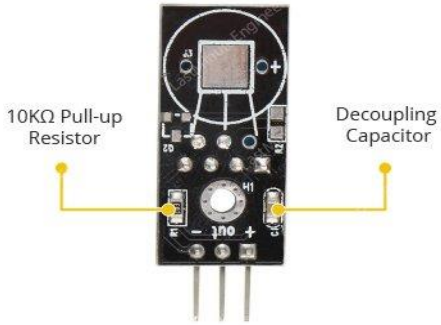


Figure 3.4 DHT11 sensor

3.1.3 LDR SENSOR

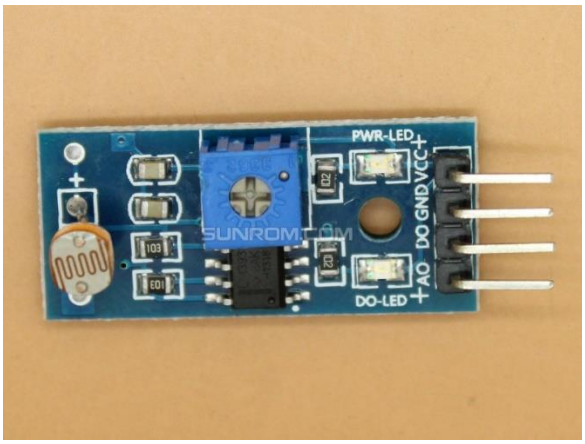


Figure 3.5 LDR sensor

The abbreviation LDR stands for light dependent resistor. LDRs, also known as photoresistors, are microscopic light-sensing devices. A resistor called an LDR is one whose resistance varies with the amount of light shining on it at any one time. As light intensity increases, the LDR resistance declines, and vice versa. We are able to create light sensor circuits with them thanks to this characteristic. A voltage divider circuit must always be created to use an LDR. When an LDR's resistance value increases in comparison to its fixed resistance, the voltage across the device increases.

3.1.4 H-BRIDGE MOTOR DRIVER

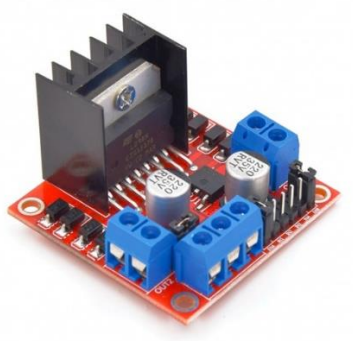


Figure 3.6 H-Bridge Motor Driver

The connection between the control circuit and the motor is made by a motor driver. The motor needs a lot of current, yet the control circuit only needs a low current signal to operate. In order to run the motor, a motor driver must change a low current control signal into a high current signal.

The L293D integrated circuit drives two H-bridge motors. To transform a low current control signal into a high current signal, a motor driver works as a current amplifier. Higher current levels of the motor are controlled by this signal. Two integrated H-bridge driver circuits are present in the L293D.

3.1.5 I2C LCD

A liquid crystal display (LCD) is an electronically modulated optical device made up of any number of colour or monochrome pixels filled with liquid crystals that is placed in front of a light source (backlight) or reflector. It is commonly used in battery-powered electronic devices since it uses very little electric power. An LCD is made up of liquid crystal components sandwiched between two glass panels.

Figure 3.7 LCD

WORKING OF LCD DISPLAY

The electrodes can be sufficiently charged to align the liquid crystal molecules in a certain direction. To activate or highlight the selected character, a polarizer rotates a light beam that is travelling through the LCD. The power supply is +5V and the maximum allowed transient voltage is 10mV. LCDs

do not emit light and require light to read the characters displayed. Backlight allows you to read even in low light.

Pin Descriptions of LCD

Table 3.1 PIN description of LCD

Pin No.	Symbol	Function
1	VSS	Ground connection
2	VDD	Power connection, +5v
3	Vo	LCD Power supply
4	RS	Register selects RS=0…Instruction registers RS=1…Data register
5	R/W	Read/Write R/W=1…Read R/W=0…Write
6	EN	Enable
7-14	DB0-DB7	Bi-directional Data Bus. If the data length of the interface is 8 bits, one data transfer is performed via DB0-DB7. If the interface data length is 4 bits, DB4 to DB7 twice. First the upper 4 bits, then the lower 4 bits.
15	LAMP-(L-)	Power supply terminal for LED or EL lamp
16	LAMP+(L+) (E2)	Enable

3.1.5 ADS1115

ADS1115 is a 16-bit resolution analogue-to-digital converter module. The ADS1115 is a low power gadget that works between 2.0 and 5.5 volts. The ADS1115 oscillator IC communicates with the microcontroller using the I2C communication protocol.

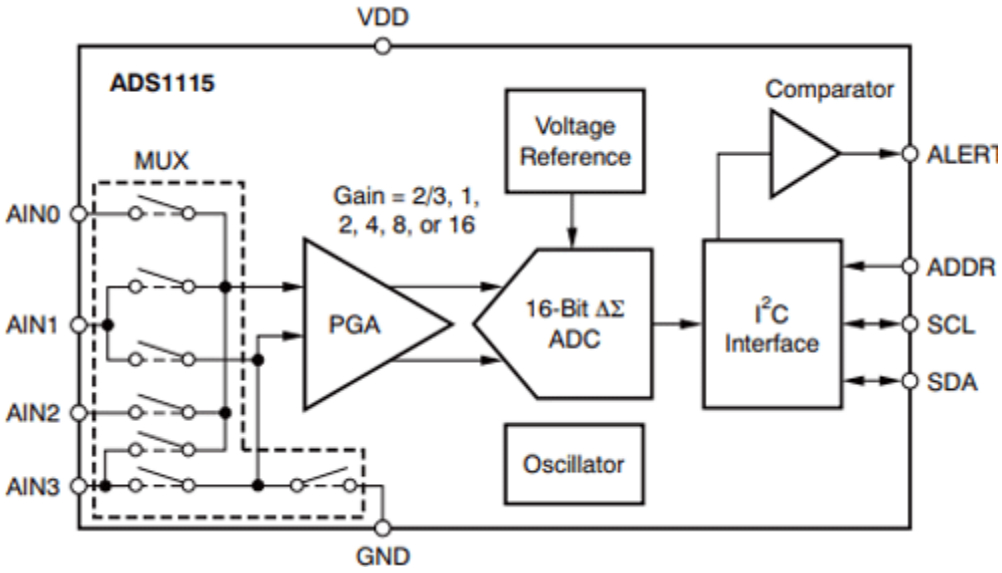


Figure 3.8 PIN diagram of ADS1115

After passing through a multiplexer, the analog input is sent to a programmable gain amplifier that amplifies the input signal. The amplified signal is then passed to a 16-bit ADC whose output is sent to the microcontroller via I2C communication.

3.2 SOFTWARE SPECIFICATIONS

3.2.1 ARDUINO IDE

3.2.1.1 Arduino Development Environment

The Arduino IDE, often known as the Arduino software, has menus, a toolbar with buttons for typical activities, a message box, a text console, and a text

editor for writing code. (IDE). connects to the Arduino board's hardware to upload and transmit programming.

3.2.1.2 Writing Sketches

Users of the Arduino software can create computer programmes known as sketches. (IDE). These images were created with a text editor and saved as files with the extension "into". Text replacement and text search features are included in the editor. When saving and exporting, feedback and mistakes are shown in the message section. The console displays text produced by the Arduino IDE programme together with additional information and comprehensive error messages. The configured board and serial port can be seen in the window's lower right corner. Using toolbar buttons, one may assess and upload programs, generate, open, and save sketches, as well as start the serial monitor.

3.2.2 PYTHON

Python is a robust but simple to learn programming language. The object-oriented programming methodology provides good high-level data structures and is easy to use. Python's simple syntax, dynamic typing, and interpreted nature make it the finest language for scripting and rapid application development across the majority of platforms. It can be used and distributed without cost. The Python interpreter is capable of accepting new C or C++ functions and data types with ease (or other languages callable from C). When looking for flexible software add-on languages, Python is a great choice.

3.2.3 THINGSPEAK – IoT ANALYTICS

The IoT analytics platform service ThingSpeakTM is provided by MathWorks®, the company that also develops MATLAB® and Simulink®. Online

real-time data streams can be gathered, shown, and examined using ThingSpeak. The data posted by your equipment or devices can be immediately visualised with ThingSpeak. Use MATLAB code in ThingSpeak to manage and analyse data as it is being sent online. The creation of proof-of-concept IoT systems, particularly those that require analytics, can be sped up using ThingSpeak. Servers and web applications are not necessary for the development of IoT solutions. ThingSpeak offers a hosted solution for small- to medium-sized IoT systems that may be implemented in real surroundings.

CHAPTER 4

SYSTEM DESIGN

4.1 WORKFLOW DIAGRAM

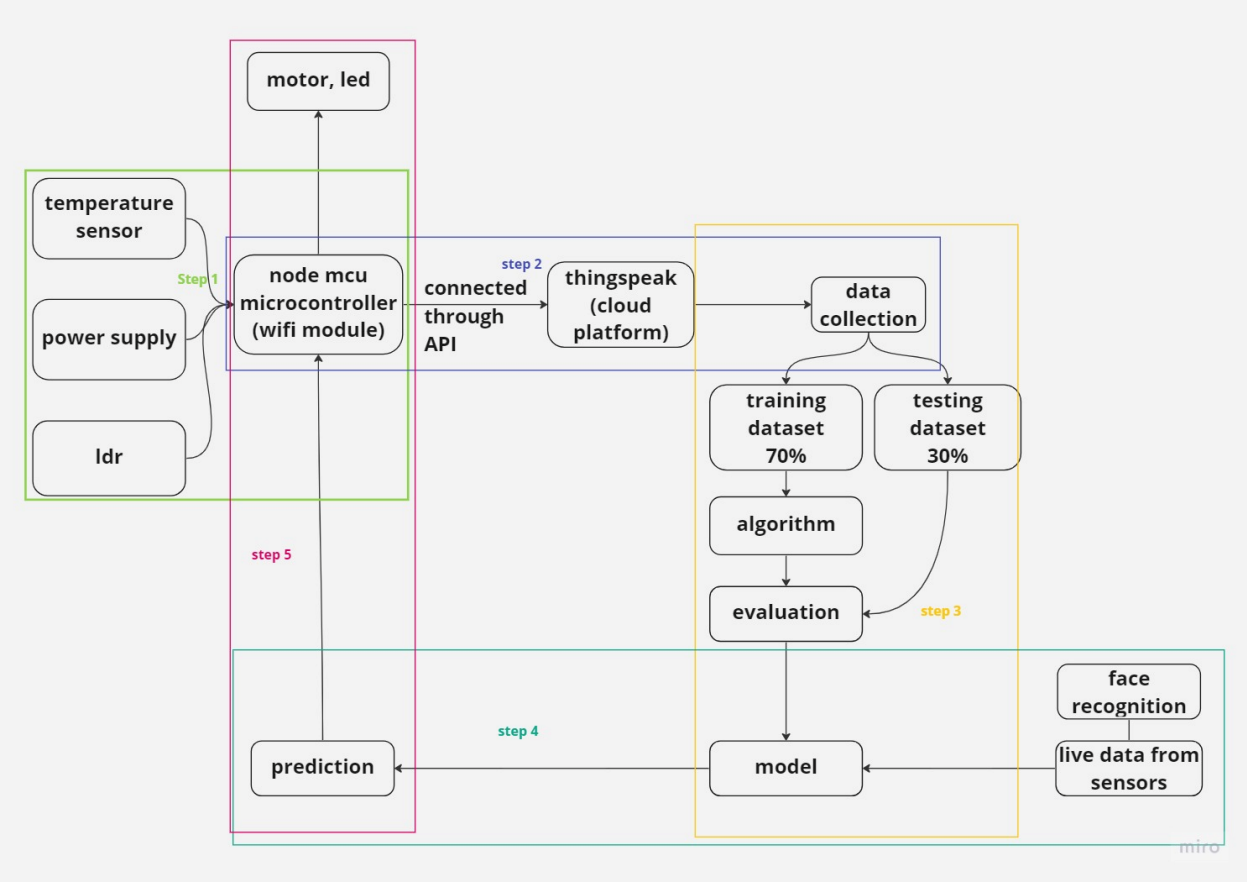


Figure 4.1 Workflow Diagram

4.2 HARDWARE MODEL

The first and foremost work in this project is to collect data according to users' preferences. To collect data, a hardware prototype is developed using Node MCU as a microcontroller, temperature, and humidity sensor (DHT11 sensor) to sense the humidity and temperature of the external environment, LDR sensor to sense the external light intensity, a H-bridge motor driver to control the speed of the fan, LED, LCD, motor, potentiometer, ADS115 analog to digital converter.

The implemented prototype is used to collect data on user preferences regarding fan speed and light intensity. An attached potentiometer is used to adjust the speed and intensity of the fan or light.

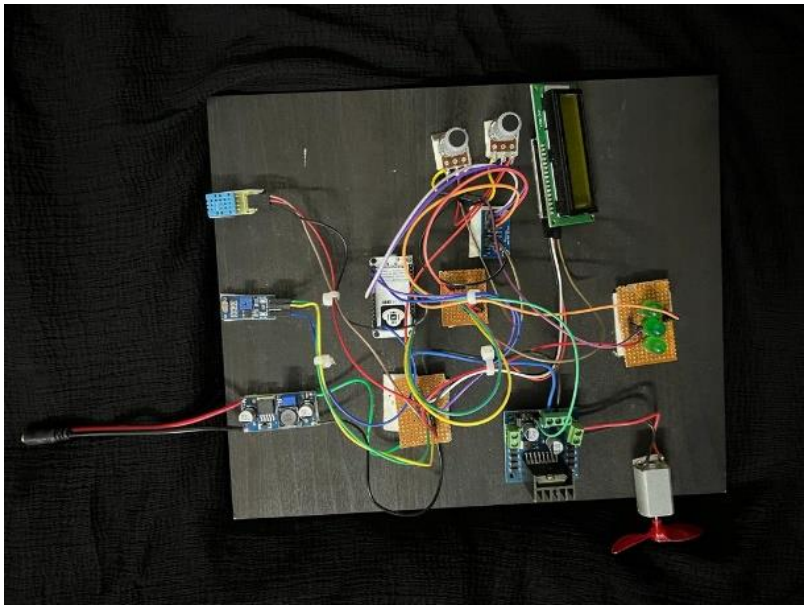


Figure 4.2 Smart light and fan hardware prototype

The DHT sensor is interfaced with D5 pin of node MCU and the LDR sensor is integrated in A0 pin of Node MCU the A0 pin represents the analog to digital converter which collects the analog value of LDR and converts to digital values and send to Node MCU. Since the Node MCU has only one analog pin additional potentiometer is integrated with ads1115 sensor, the ads module and lcd display is connected in I2C protocol D1 and D2 pin of Node MCU and the motor driver is connected to D6 and D7 the pin of node MCU. From this prototype, data such as fan speed, light intensity, external light intensity, external temperature and humidity will be collected and used for future improvements.

4.3 FACE RECOGNITION MODEL

The face recognition library is used to construct a deep learning-based face recognition system in this project. Installing and importing face recognition libraries like dlib, face_recognition, and opencv comes first. The face recognition library is then loaded with test photos, and this library swiftly finds faces on its own. The deep learning-based library face recognition offers single-shot learning, which means it only needs one image to train itself to recognise a person.

4.3.1 WORKING OF FACE RECOGNITION MODEL

• We give the model the person's name and photo.

• The model collects all of the images, encodes them numerically, and stores them in a list together with all of the labels (people's names) in another list.

• During the prediction phase, the recognition model encodes the image of the unfamiliar person when we pass it by.

• After encoding the image of an unidentified individual, it searches for the encoding that is the closest in terms of distance. The closest match will be the store encoding that is closest to the encoding of an unknown person.

• After finding the encoding with the closest match, indexing is utilized to take the index of that encoding from the list. The name of the detected person is identified.

4.4 DATA COLLECTION

The hardware model built is used in the collection of data according to the usage of fan and light in correspondence with the environmental temperature and light intensity. The microcontroller of the model i.e., NodeMCU – ESP8266, which is a Wi-Fi module, helps in collection of data. One can easily sign-in to the ThingSpeak platform, create their own channel for data collection in real-time. The Wi-Fi module is connected to ThingSpeak cloud platform with the help of Arduino

code. In the Arduino code include ThingSpeak library, personal Wi-Fi ID and password, ThingSpeak channel ID and the secret API key and the details to be collected in order to visualize and export the real-time data. The data collected here are date, time, environmental temperature and light intensity, values of fan speed and light intensity regulated by the user. For two users, a total of 900+ data are collected for this project. This data is then combined with face ID and then is finalized for training using machine learning classifier algorithms.

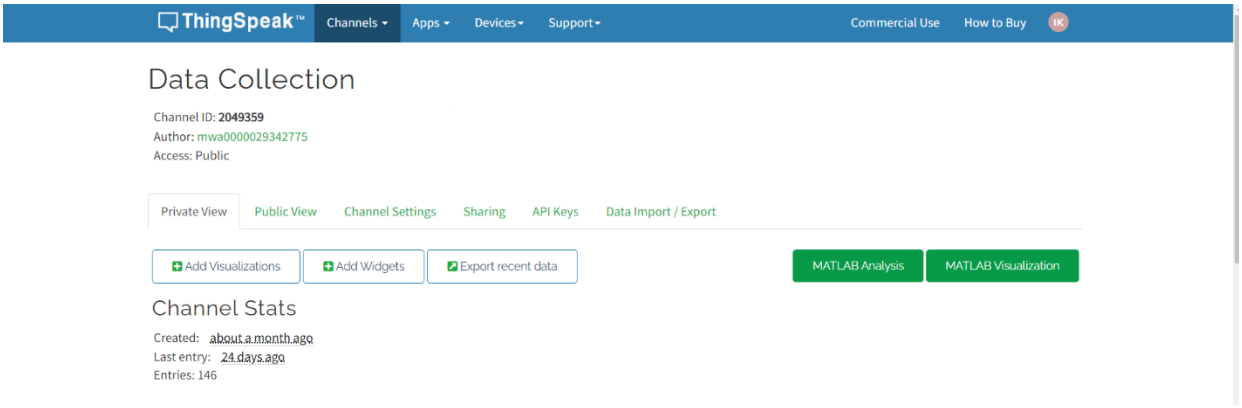


Figure 4.3 ThingSpeak channel for data collection

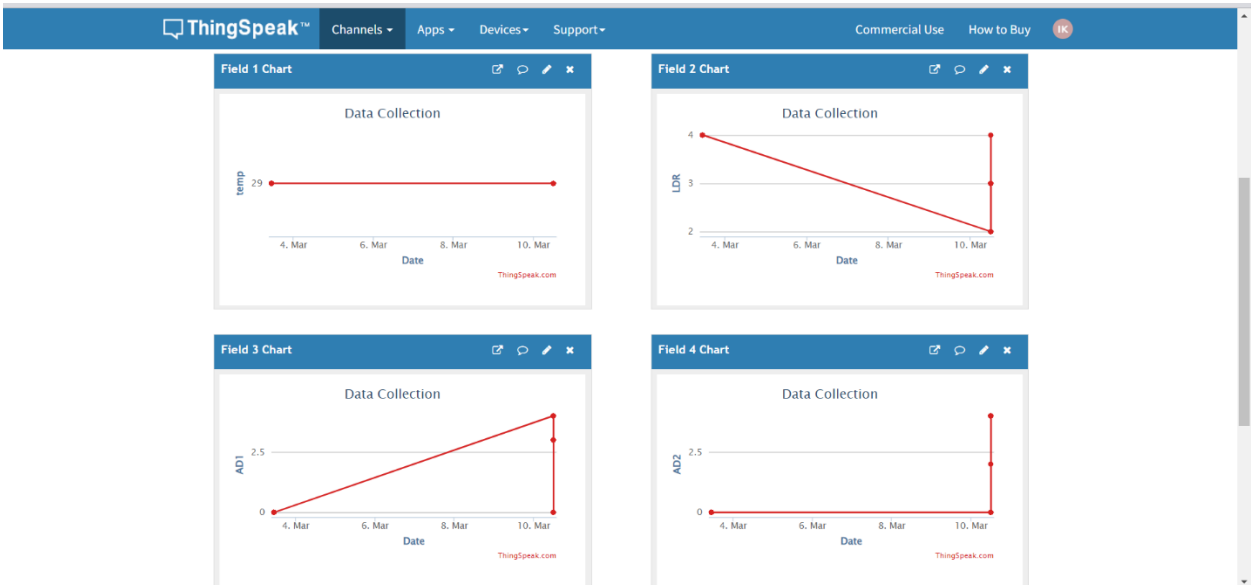
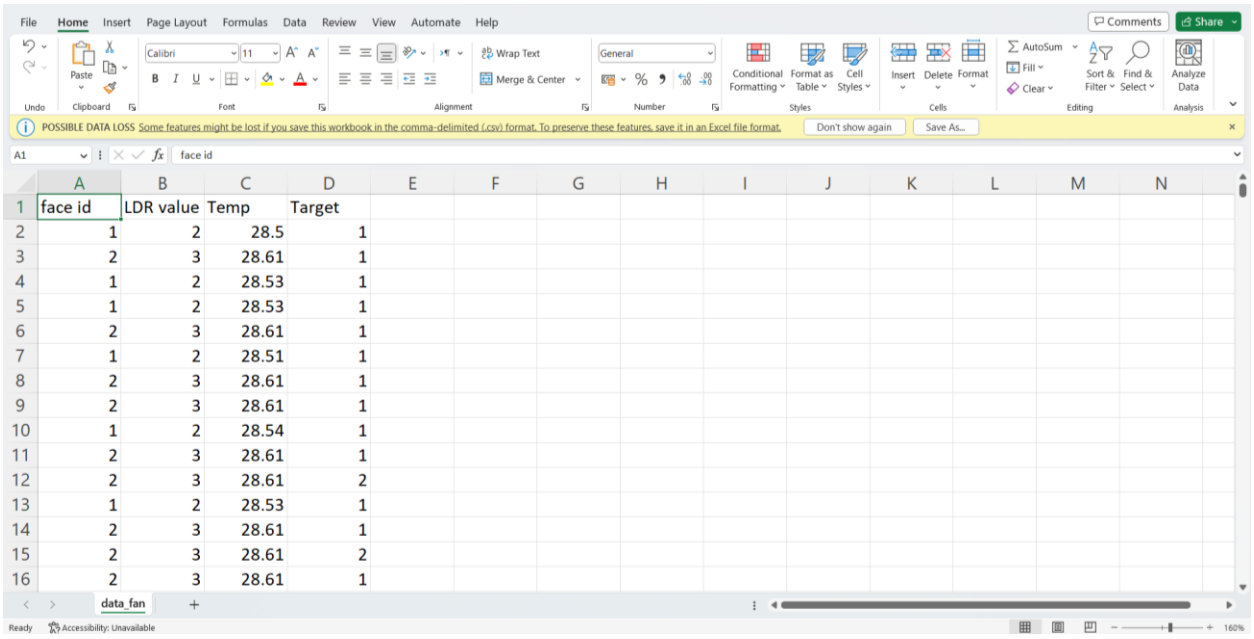


Figure 4.4 Real-time data visualizing graph in ThingSpeak



face id	LDR value	Temp	Target
1	2	28.5	1
2	3	28.61	1
1	2	28.53	1
1	2	28.53	1
2	3	28.61	1
1	2	28.51	1
2	3	28.61	1
2	3	28.61	1
1	2	28.54	1
2	3	28.61	1
2	3	28.61	2
1	2	28.53	1
2	3	28.61	1
2	3	28.61	2
2	3	28.61	1

Figure 4.5 Exported data from ThingSpeak

4.5 MACHINE LEARNING ALGORITHMS

4.5.1 K-NEAREST NEIGHBORS

The supervised machine learning method known as the k-nearest neighbour (K-NN) classifier is non-parametric. It classifies items according to their proximity to one another and the classifications of their immediate neighbours. Although K-NN can be used to address regression-related problems, classification tasks are where it is most frequently applied. The number of labelled points (neighbours) taken into account while classifying an item is indicated by the k parameter in a K-NN. Depending on the amount of k, the number of these points required to calculate the result varies. Our tasks are to determine the distance and find the categories that are closest to our unidentified entity.

The underlying principle of the k-nearest neighbour approach is as follows. We can search for the nearest points in our feature space to a point whose class is unknown. These points are the k-nearest neighbours. It is very likely that the point belongs to the same class as its neighbours because related items usually

occupy the same locations in feature space. Based on it, a new point might be assigned to one class or another. The ideal k value can be used to get the best model accuracy.

Finding the k value that produces the best results on the testing set is the simplest solution. The stages we take are as follows:

For k, pick a random number. There are no predetermined rules; in practise, k is often picked at random from the range of 3 to 10. Low values of k cause decision boundaries to become unstable. Because better metrics are infrequently the end result, decision boundaries frequently flatten when k is large. It consequently frequently involves trying and failing.

See how accurate different k values are on the testing set by putting them to the test.

People need to come up with new, time- and money-effective ways to calculate k when there are anomalies or a lot of neighbours that are close to the unknown place.

The major constraints are connected to processing resources: the cost of suitable k selection increases with the number of objects in the dataset. Also, when working with large datasets, a multiparameter optimization configuration is used. By selecting the appropriate k in the meta-training process, we hope to maximize model quality while also accelerating model inference. Inference time increases as k increases, which may be problematic for huge data sets.

Methods for Selecting k

Square root method

The optimum K value can be calculated as the square root of the total number of samples in the training dataset. Use either an accuracy plot or an error plot to select the ideal K value. While K-NN performs well for classes with many

labels, it struggles for structures with many outliers; in these cases, alternative methods must be utilized.

Cross validation method

Determine the accuracy after running cross-validation 5–10 times starting with k=1. These values are commonly used because they successfully strike a balance between the requirements for computing and statistical validity. Up until consistent outcomes are reached, the same actions are taken. The error frequently decreases as k increases, stabilizes, and then increases again. Towards the start of the stable zone is where you'll find the ideal k.

Calculating k-distance

Euclidean distance

The length of the section of a straight line that connects two locations is known as their Euclidean distance. This well-known metric of distance is used to compare vectors with actual values.

Manhattan distance

The total of the absolute differences between each point's x and y coordinates is the Manhattan distance between two points. The taxicab distance, which is usually referred to as the shortest distance between two points in a city, is obtained by adding the lengths of each interval.

Minkowski distance

The Minkowski distance generalises the Manhattan and Euclidean distances. It has a "order" parameter that makes it possible to calculate a number of different distance metrics. The Minkowski distance is a measure of the separation between two locations in a normed vector space.

Hamming distance

When contrasting two binary vectors, the Hamming distance method is

used. (also called data strings or bitstrings). Prior to computation, data must be transformed into a binary format.

4.5.2 SUPPORT VECTOR MACHINE

The support vector machine algorithm seeks to find a hyperplane that clearly classifies the data points in an N-dimensional space. (N is the number of characteristics).

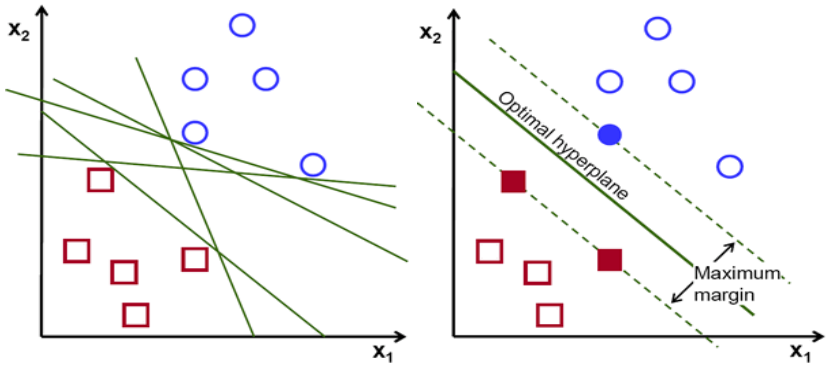


Figure 4.6 Possible Hyperplanes

There are countless alternative hyperplanes that can be used to split the two groups of data points. Finding the plane with the largest margin, or the distance between the most data points from the two classes, is the goal. The margin distance increases as more data is shown, increasing the certainty with which the following data points can be categorized.

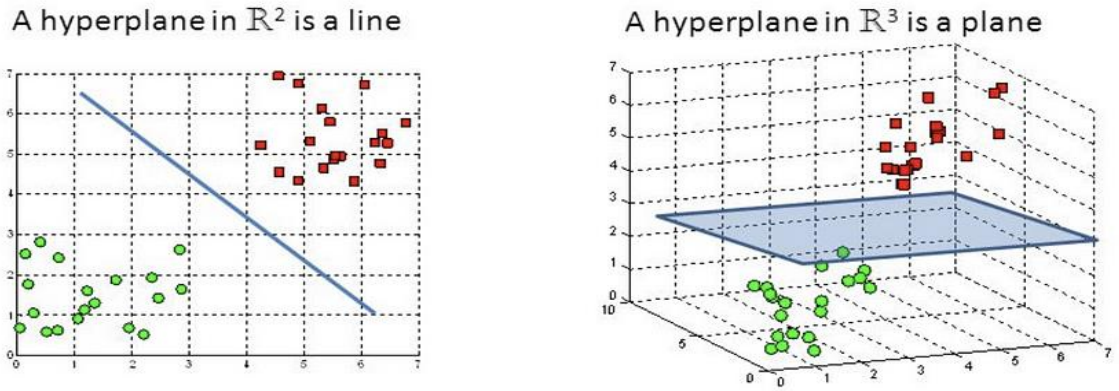


Figure 4.7 Hyperplanes in 2D and 3D feature Space

Hyperplanes are employed as decision boundaries to categorise the data points. The data points located on either side of the hyperplane can be divided into numerous groups. Additionally, the size of the hyperplane is influenced by the number of features. When there are only two input characteristics, the hyperplane seems to be a straight line. When three input features are available, the hyperplane degenerates into a two-dimensional plane. More than three aspects make it more difficult to visualize.

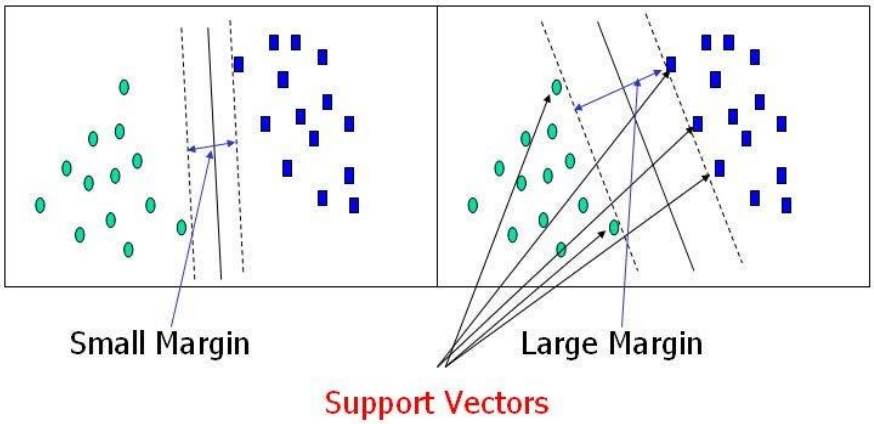


Figure 4.8 Support Vectors

Support vectors are closer to the hyperplane data points that influence the position and orientation of the hyperplane. Using these support vectors boosts the classifier's margin. If the support vectors are eliminated, the position of the hyperplane will change. These concepts serve as the foundation for the growth of our SVM.

SVM considers the outcome of the linear function. The output belongs to one class if it is more than 1, and to another class if it is less than 1. The SVM threshold values are now 1 and -1, therefore the buffer is the reinforcing range of values [-1,1]. The SVM technique aims to increase the separation between the data points and the hyperplane. Hinged loss is the loss function that aids in maximising

the margin.

$$c\big(x, y, f(x)\big) = \begin{cases} 0 & if\ y * f(x) \geq 1 \\ 1 - y * f(x), & else \end{cases} \quad (4.1)$$

If the expected value and the actual value have the same sign, there is no cost. If not, the loss's monetary amount is next determined. The cost function additionally makes advantage of the regularization parameter. The regularization parameter aims to strike a balance between margin maximization and loss reduction. The cost functions are as follows once the regularization parameter has been used.

$$min_w\, \lambda \big|\big|w\big|\big|^2 \ + \ \sum_{i=1}^{n} (1 - y_i \langle x_i, w \rangle)_+ \quad (4.2)$$

Given what is known about the loss function, the gradients can be found by taking partial derivatives with respect to the weights. Gradients can be used to adjust the weights.

$$\frac{\partial}{\partial w_k} \lambda\, ||w||^2 = 2\lambda w_k \quad (4.3)$$

$$\frac{\partial}{\partial w_k} (1 - y_i \langle x_i, w \rangle)_+ = \begin{cases} 0, if\ y_i \langle x_i w \rangle \geq 1 \\ -y_i x_{ik}, \ else \end{cases} \quad (4.4)$$

The regularization parameter's gradient only has to be changed when there is no misclassification, or when the model correctly predicts the class of the data point.

$$\omega = \omega - \alpha . (2\lambda\omega) \quad (4.5)$$

We combine the loss with the regularization parameter to perform a gradient update when a data point is incorrectly categorized or when our model incorrectly predicts the class for a specific data point.

$$\omega = \omega + \alpha . (y_i . x_i - 2\lambda\omega) \quad (4.6)$$

4.5.3 RANDOM FOREST

A supervised learning algorithm is random forest. It comes in two different forms; one is used to solve classification problems, the other to solve regression issues. One of the most adaptable and user-friendly algorithms is this one. On the basis of the provided data samples, it constructs decision trees, obtains predictions from each tree, and votes for the top solution. It also serves as a fairly accurate measure of feature importance.

By mixing various decision-trees, the Random Forest approach builds a forest of trees, hence the name. With more trees in the forest, the random forest classifier becomes more accurate. Following are the first two steps of the random forest algorithm.

In the first stage, "k" features are chosen at random from a total of m features to build the random forest. Then move forward as follows throughout the first stage:-

- Choose k features at random from a total of m features, where k m.
- Determine the node d by choosing the optimum split point among the k attributes.
- Use the best split to divide the node into daughter nodes.
- As many times as necessary, repeat steps 1 through 3 to create l number of nodes.
- To create n trees for a forest, repeat steps 1 through 4 as necessary.

Using the trained random forest method, predictions are created in the second stage. The test features are taken, the rules of each decision tree generated at random are applied, and record the anticipated results. The votes for each forecasted target are then determined. The final forecast made by the random forest algorithm is then taken into consideration.

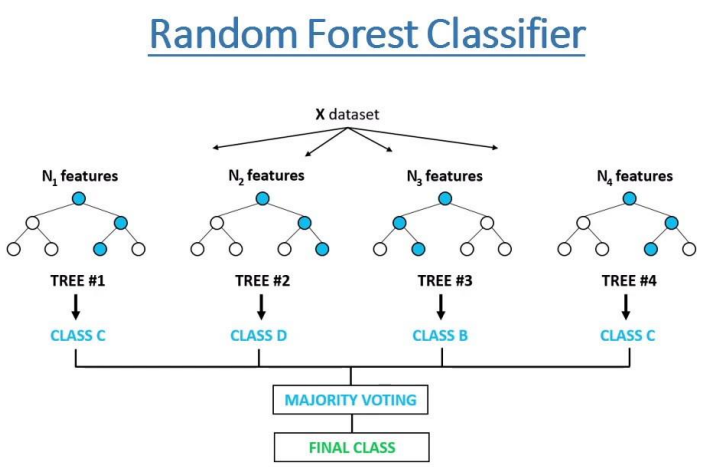


Figure 4.9 Random Forest Classifier

The random forests algorithm can be used to implement the feature selection approach. In classification or regression issues, this method can be used to prioritize variables. To assess the data and determine the statistical importance of each variable in a dataset, utilize the random forest approach. The out-of-bag error for each data point is noted and averaged over the forest during the fitting procedure. After training, the j-th feature's significance was assessed. The out-of-bag error was once again calculated using this perturbed dataset, which had been altered to reflect the values of the j-th feature in the training dataset. By averaging the out-of-bag error between the permutation's before and after conditions over all trees, the relevance score for the j-th feature is determined. The standard deviation of these differences is used to standardize the score.

The importance of characteristics is valued higher than that of features that produce tiny values for this score. The most crucial attributes and exclude the less crucial ones for the model development process are selected based on this score.

4.5.4 NAÏVE BAYES

Finding the class of the observation given the values of the attributes is the aim of the supervised learning technique using naive bayes. (data point). A naive bayes classifier (p(yi | x1, x2,..., xn)) determines the likelihood of a class given a set of feature values.

$$\boldsymbol{p(yi|x1, x2, \dots, xn) = \frac{p(x1, x2, \dots, xn|yi).p(yi)}{p(x1,x2,\dots xn)}} \quad (4.7)$$

p(x1, x2,..., xn | yi) represents the likelihood that a certain feature combination will receive a class label. We need very large datasets in order to estimate the probability distribution for every possible feature value combination. Because it is assumed that each feature is independent of every other feature, the naive Bayes method overcomes this problem. The denominator (p(x1,x2,..., xn)), which merely normalizes the conditional probability value for a class given an observation (p(yi | x1,x2,..., xn)), can be removed to simplify the equation.

It's simple to determine a class's probability (p(yi)):

$$\boldsymbol{p(yi) = \frac{number\ of\ observations\ with\ class\ yi}{number\ of\ all\ observations}} \quad (4.8)$$

When features are thought to be independent, the expression p(x1, x2,..., xn | yi) can be written as follows:

$$\boldsymbol{p(x1, x2, \dots, xn|yi) = p(x1|yi).p(x2|yi) \dots . p(xn|yi)} \quad (4.9)$$

Calculating the conditional probability for a single feature given the class label (p(x1 | yi)) using the data may be easier. Every class's characteristic probability distributions must be independently recorded by the procedure. For instance, 50 distinct probability distributions need to be recorded if there are 5 classes and 10 characteristics. The qualities of features determine the type of distributions:

- Binary features include Y/N, True/False, and 0/1. Distribution using Bernoulli
- Regarding distinct properties, such as word counts: Multiple node distribution

- About recurring features: Normative (Gaussian) distribution

The distribution of features model is another name for the naive Bayes model. (i.e. Gaussian naive bayes classifier). It can be required to use a different sort of distribution for different attributes in mixed-type datasets. Using the naive bayes method, it was simple to combine the probability of recognizing a class given the values of the features (p(yi | x1, x2,..., xn)).

4.5.5 DECISION TREE

Decision Trees use the dataset features to generate yes/no questions and attempt to repeatedly split the dataset until all the data points that correspond to each class are isolated. This procedure organises the data into a tree-like structure. The tree grows by one node each time a question is asked. Moreover, the root node is the very first node. Depending on how relevant a feature is as a result of a question's answer, the dataset is segmented into nodes. Leaf nodes are the last nodes produced if the process concludes in a split. One branch of the tree's facts all corroborate the idea that the query indicated by the preceding node's rule has a favorable outcome. At a node on the opposing branch, a few more data points can be found.

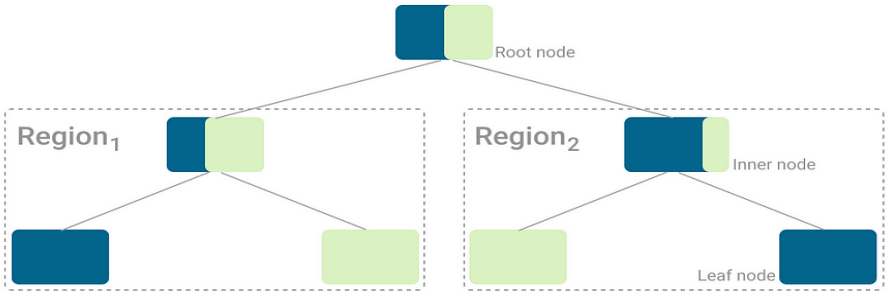


Figure 4.10 Regions and types of nodes in a tree

The method attempts to assign each leaf node a node that doesn't further separate the data into a single class by thoroughly separating the dataset. They are referred to as pure leaf nodes.

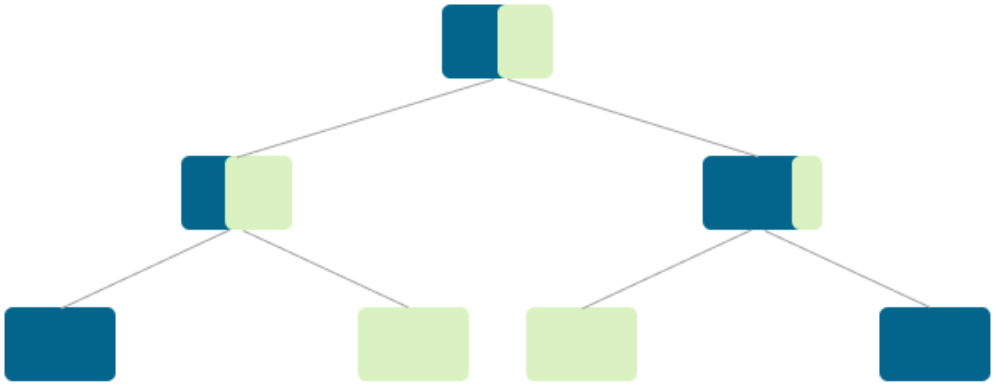


Figure 4.11 Tree with only pure leaf nodes

Mixed leaf nodes, on the other hand, are generally found when not all of the data points belong to the same class.

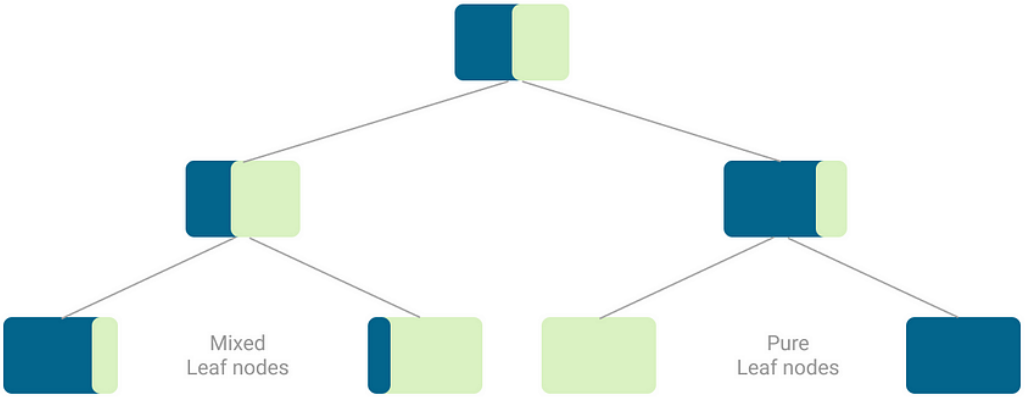


Figure 4.12 Tree with pure and mixed leaf nodes (before assigning final class)

At the end of the process, each leaf node's data points can only be divided into one class. This issue has already been resolved because every data point in a pure leaf node belongs to the same class. Out of all the data points collected at mixed leaf nodes, the approach assigns the class that has been seen the most frequently.

4.6 IMPLEMENTATION OF THE PROPOSED SYSTEM

Firstly, a face recognition model is created using face_recognition library which is built using deep learning methods and is based on single shot learning which means the model works using a single image for a person. This is done for the identification of known and unknown users. As a next step, from the hardware model designed, data such as environmental temperature and light intensity, fan speed and light brightness according to the user along with date and time are collected. Date and time are excluded from the file for this project. The data is collected for two known users and hence their face IDs are included along with the collected data. These data are split into 70 percent and 30 percent for training and testing respectively. The training set is trained by various machine learning classifiers such as K-Nearest Neighbor, Support Vector Machine, Random Forest, Naïve Bayes and Decision Tree. These classification models are used in order to select the most suitable one to make the system work more accurately. Training of dataset is done by importing specific classifier models and accuracy of each model is calculated in order to choose the best fit. Accuracy is a measure of how much test data is classified or predicted in their most suitable class correctly. Accuracy of each algorithm is calculated and the best fit is found for the proposed system among the mentioned classifiers.

CHAPTER 5
RESULTS AND DISCUSSION

5.1 TESTING THE PROTOTYPE

The first main step to be done in this project is data collection, for which a hardware model consisting of motor and LED is built. The data is collected with the help of microcontroller (NodeMCU-ESP8266) and the data are stored live in ThingSpeak cloud platform from where those data are retrieved in .csv format. The following is the number of values in various range of speed of fan and intensity of light from the data collected.

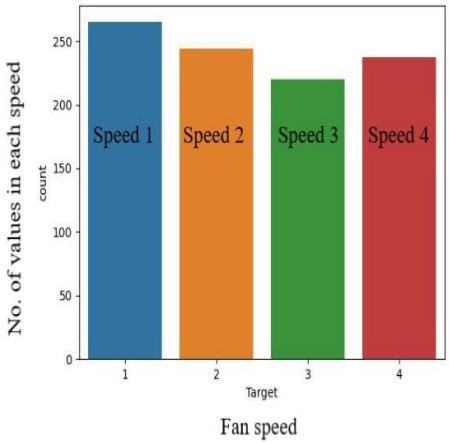


Figure 5.1 Fan Speed

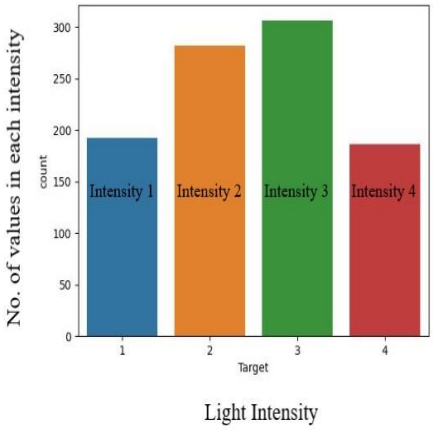


Figure 5.2 Light Intensity

The data collected are trained and tested using various machine learning classifier models such as k-nearest neighbors, Support Vector Machine, random forest, Naïve bayes and decision trees. From these models, the most accurate ones are chosen and used to regulate the motor and LED. Each classifier has its own advantages and disadvantages. The following is the comparison of each classifier according to the

accuracy obtained while testing the data collected.

Table 5.1 Accuracy Comparison

Algorithm	Accuracy (%)	
	Fan	Light
K-Nearest Neighbor	93.45	92.07
Support Vector Machine	89.66	92.07
Random Forest	91.38	90.69
Naïve Bayes	87.59	92.07
Decision Tree	91.38	89.66

From the above comparison, it is concluded that K-NN is more accurate than others for this particular dataset which approximate consists of more than 900 data.
K-Nearest Neighbors classifier algorithm is most suitable for training large datasets with large number of classes. Its is easy to understand, simple to implement and adapts to new training data immediately. Then face recognition model is added to the K-NN model and then finally its is used in the autoregulation of fan and light according to the user preference.

5.2 RESULT ANALYSIS

This project aims in autoregulation of fan and light according to the preferences of the particular user, kind of contrary to popular belief. It essentially starts with collection of data from a hardware model built using a microcontroller called NodeMCU - ESP8266, which is a Wi-Fi module, an LCD, an LED instead of light, a motor instead of fan, ADS115 which is an analog to digital convertor module and power supply, basically contrary to popular belief. From this model, the data kind of such as date, time, environmental light intensity and temperature, speed of fan preferred and light brightness preferred, which mostly is fairly significant. Along

with these a face recognition model essentially is built for identification of the user to work accordingly in a big way. Various machine learning models such as k-nearest neighbors, Support Vector Machine, random forest, Naïve bayes and decision trees are used and the data collected is trained with these classification models in a really big way. The most accurate one among these basically is chosen and trained accordingly after which the microcontroller receives the trained model and specifically makes for all intents and purposes sure that the fan and light works accordingly, or so they essentially thought. The model built and trained gives an accuracy of approximately 93 percent.

CHAPTER 6
CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The automated fan and light system, which was created utilizing machine learning and data (fan speed, light intensity) acquired from user preferences, provides a great deal of comfort to elderly people. They do not need to carry their phones or remote controls to use the appliance, unlike remote-controlled or app-controlled models. In this project, the appliance uses a camera to identify the user and then adjusts its behavior to that person. The camera runs in line with the environment and sends a letter about an unknown user it detects to a known user.

6.2 FUTURE ENHANCEMENT

The utilization of user preferences in the project demonstrates the potential for a number of improvements, such as the collection of heart rate and sweat rate using certain currently available sensors. With the use of this data, a smart band might be developed, enabling consideration of individual preferences when utilizing electrical devices. This technology might be particularly useful in the healthcare sector because it has the ability to customize how electrical devices are used. Such specialized electrical devices could be employed in settings like private rooms, cabins, and study halls, to name a few. Moreover, the information gathered could include the time and date in order to adjust the system to the user's preferences for time, date, and time by including a real-time clock.

REFERENCES

[1] Assaf, M. H., Mootoo, R., Das, S. R., Petriu, E. M., Groza, V., & Biswas, S. (2012, May). “Sensor based home automation and security system”. In 2012 IEEE International Instrumentation and Measurement Technology Conference Proceedings, pp. 722-727.

[2] Chekired, F., Houtti, S., Bouroussis, C. A., Rahmani, A., Tilmatine, A., & Canale, L. (2020), “Low-cost automation system for smart houses based on PIC microcontrollers.” In 2020 IEEE International Conference on Environment and Electrical Engineering and 2020 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe), pp. 1-5.

[3] Deena, P. F., Raj, G. S., Dutt, G., & Jinny, S. V. (2017), “IOT based smart street light management system.” In 2017 IEEE international conference on circuits and systems (ICCS), pp. 368-371.

[4] Jaihar, J., Lingayat, N., Vijaybhai, P. S., Venkatesh, G., & Upla, K. P. (2020), “Smart home automation using machine learning algorithms”. In 2020 International Conference for Emerging Technology (INCET), pp. 1-4.

[5] Kaushik, S., Chouhan, Y. S., Sharma, N., Singh, S., & Suganya, P. (2018), “Automatic fan speed control using temperature and humidity sensor and Arduino.” Int. J. Adv. Res, Vol. 4, No.2, pp. 453-457.

[6] Mahamud, M. S., Zishan, M. S. R., Ahmad, S. I., Rahman, A. R., Hasan, M., & Rahman, M. L. (2019, January). “Domicile-an IoT based smart home automation system”. In 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), pp. 493-497.

[7] Mishra, R., & Elisabeth, E. A. (2017), “Speed control of ceiling fan using PWM technique”. In 2017 International Conference on Computation of Power, Energy Information and Communication (ICCPEIC), pp. 686-690.

[8] Montalbo, F. J., & Enriquez, E. (2020), “An IoT smart lighting system for university classrooms”. In 2020 International Symposium on Educational Technology (ISET), pp. 3-7.

[9] Motlagh, N. H., Khajavi, S. H., Jaribion, A., & Holmstrom, J. (2018, November). “An IoT-based automation system for older homes: A use case for lighting system”. In 2018 IEEE 11th Conference on Service-Oriented Computing and Applications (SOCA) pp. 1-6.

[10] Ouerhani, N., Pazos, N., Aeberli, M., & Muller, M. (2016), “IoT-based dynamic street light control for smart cities use cases.” In 2016 International Symposium on Networks, Computers and Communications (ISNCC), pp. 1-5.

[11] Roy, P., Saha, J., Dutta, N., & Chandra, S. (2018), “Microcontroller based automated room light and fan controller”. In 2018 Emerging Trends in Electronic Devices and Computational Techniques (EDCT), pp. 1-4.

[12] Saad, M., Abdoalgader, H., & Mohamed, M. “Automatic Fan Speed Control System Using Microcontroller”. In 6th Int’l Conference on Electrical, Electronics & Engineering (ICEECE’2014), pp. 86-89.

[13] Vishwakarma, S. K., Upadhyaya, P., Kumari, B., & Mishra, A. K. (2019, April). “Smart energy efficient home automation system using IoT”. In 2019 4th international conference on internet of things: Smart innovation and usages (IoT-SIU), pp. 1-4.

[14] Yammen, S., Tang, S., & Vennapusa, M. K. R. (2019), “IoT based speed control of Smart Fan.” In 2019 Joint International Conference on Digital Arts, Media, and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT-NCON), pp. 17-20.

APPENDIX

Data Collection – Arduino code

```
#include <ESP8266WiFi.h>
#include "ThingSpeak.h" // always include thingspeak header file after other
header files and custom macros
char ssid[] ="Project";   // your network SSID (name)
char pass[] = "12345678";   // your network password
int keyIndex = 0;            // your network key Index number (needed only for WEP)
WiFiClient  client;
#include "DHTesp.h"
#include <Adafruit_ADS1X15.h>
int ldr=0;
#include <LCD_I2C.h>

LCD_I2C lcd(0x27);
Adafruit_ADS1115 ads;
float adc_voltage = 0.0;
float in_voltage = 0.0;

 String st="";
// Floats for resistor values in divider (in ohms)
float R1 = 30000.0;
float R2 = 7500.0;

// Float for Reference Voltage
float ref_voltage = 5.0;
```

```
// Integer for ADC value
int adc_value = 0;

#ifdef ESP32
#pragma message(THIS EXAMPLE IS FOR ESP8266 ONLY!)
#error Select ESP8266 board.
#endif

DHTesp dht;

int count = 0;

void setup() {
  pinMode(D5,OUTPUT);
  pinMode(D6,OUTPUT);
  //digitalWrite(D6,HIGH);
 pinMode(A0,INPUT);
  Serial.begin(9600);
  lcd.begin(); // If you are using more I2C devices using the Wire library use
lcd.begin(false)
                // this stop the library(LCD_I2C) from calling Wire.begin()
    lcd.backlight();
    lcd.setCursor(0, 0);
    lcd.print("HOME"); // You can make spaces using well... spaces
    lcd.setCursor(0, 1); // Or setting the cursor in the desired position.
    lcd.print("MONITORING");
    delay(2500);
```

```
    lcd.clear();
if (!ads.begin()) {
    Serial.println("Failed to initialize ADS.");
    while (1);
  }
   dht.setup(D0, DHTesp::DHT11);
   WiFi.mode(WIFI_STA);
  ThingSpeak.begin(client);
   if(WiFi.status() != WL_CONNECTED){

    while(WiFi.status() != WL_CONNECTED){
      WiFi.begin(ssid, pass);  // Connect to WPA/WPA2 network. Change this line if
using open or WEP network
      Serial.print(".");
      delay(5000);
    }
    Serial.println("\nConnected.");
  }
}

void loop() {
  ldr=analogRead(A0);
  delay(dht.getMinimumSamplingPeriod());
ldr=ldr/250;
  float humidity = dht.getHumidity();
  float temperature = dht.getTemperature();
lcd.setCursor(0, 0);
```

```
    lcd.print("TEMPERATURE"); // You can make spaces using well... spaces
    lcd.setCursor(0, 1); // Or setting the cursor in the desired position.
    lcd.print(temperature);
    delay(500);
    lcd.clear();

int16_t adc0, adc1, adc2, adc3;
  float volts0, volts1, volts2, volts3;

  adc0 = ads.readADC_SingleEnded(0);
  adc1 = ads.readADC_SingleEnded(1);
  adc2 = ads.readADC_SingleEnded(2);
  adc3 = ads.readADC_SingleEnded(3);
adc0=adc0/4000;
adc1=adc1/4000;
analogWrite(D5,(adc0*4));
analogWrite(D6,(adc1*40));
if(ldr==0)
{
  ldr=1;
  }
  volts0 = ads.computeVolts(adc0);
  volts1 = ads.computeVolts(adc1);
  volts2 = ads.computeVolts(adc2);
  volts3 = ads.computeVolts(adc3);
Serial.println(adc0);
lcd.setCursor(0, 0);
```

```
    lcd.print("POT-1"); // You can make spaces using well... spaces
    lcd.setCursor(0, 1); // Or setting the cursor in the desired position.
    lcd.print(adc0);
    delay(500);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("POT-2"); // You can make spaces using well... spaces
    lcd.setCursor(0, 1); // Or setting the cursor in the desired position.
    lcd.print(adc1);
    delay(500);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("LDR"); // You can make spaces using well... spaces
    lcd.setCursor(0, 1); // Or setting the cursor in the desired position.
    lcd.print(ldr);
    delay(500);
    lcd.clear();
Serial.println(adc1);
 ThingSpeak.setField(1,temperature );
  ThingSpeak.setField(2,ldr);
  ThingSpeak.setField(3,adc0);
  ThingSpeak.setField(4,adc1);
   int xrt = ThingSpeak.writeFields(2049359, "6Y4L3ZDOATSXYZYM");


}
```

Training Machine learning classifiers – Python Code

K-Nearest Neighbours

```
import numpy as np
import pandas as pd
from sklearn import metrics
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
a=int(input("Enter 1 for fan train 2 for light"))
if a==1:
   m="data_fan.csv"
else:
   m="data_light.csv"
data = pd.read_csv(m)
data.head()
data.shape
X = data.iloc[:,:-1]
X.head()
y = data.iloc[:,-1]
y.head()
data['Target'].value_counts()
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=1)
sns.countplot(x='Target',data=data)
plt.show()
X_train.shape
X_train.head()
```

```
y_test.shape
y_test.head()
from sklearn.metrics import accuracy_score
max_accuracy = 0

from sklearn.neighbors import KNeighborsClassifier



for x in range(1,32):
    model = KNeighborsClassifier(n_neighbors=x)
    model.fit(X_train,y_train)
    y_pred = model.predict(X_test)
    current_accuracy = round(accuracy_score(y_pred,y_test)*100,2)
    if(current_accuracy>max_accuracy):
        max_accuracy = current_accuracy
        best_x = x

#print(max_accuracy)
#print(best_x)

model = KNeighborsClassifier(n_neighbors=best_x)
model.fit(X_train,y_train)
y_pred = model.predict(X_test)
if a==1:
    filename = 'knnfan.sav'
else:
    filename = 'knnlight.sav'
```

```
pickle.dump(model, open(filename, 'wb'))
acc=(metrics.accuracy_score(y_pred,y_test)*100)
print("Accuracy is:",acc)
```

Support Vector Machine

```
import numpy as np
import pandas as pd
from sklearn import metrics
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
a=int(input("Enter 1 for fan train 2 for light"))
if a==1:
   m="data_fan.csv"
else:
   m="data_light.csv"
data = pd.read_csv(m)
data.head()
data.shape
X = data.iloc[:,:-1]
X.head()
y = data.iloc[:,-1]
y.head()
data['Target'].value_counts()
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=1)
sns.countplot(x='Target',data=data)
```

```
plt.show()
X_train.shape
X_train.head()
y_test.shape
y_test.head()


from sklearn.svm import SVC
model = SVC(kernel='linear')
model.fit(X_train,y_train)
if a==1:
   filename = 'svmfan.sav'
else:
   filename = 'svmlight.sav'
pickle.dump(model, open(filename, 'wb'))
y_pred = model.predict(X_test)
acc=(metrics.accuracy_score(y_pred,y_test)*100)
print("Accuracy is:",acc)
```

Random Forest

```
import numpy as np
import pandas as pd
from sklearn import metrics
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
```

```
a=int(input("Enter 1 for fan train 2 for light"))
if a==1:
   m="data_fan.csv"
else:
   m="data_light.csv"
data = pd.read_csv(m)
data.head()
data.shape
X = data.iloc[:,:-1]
X.head()
y = data.iloc[:,-1]
y.head()
data['Target'].value_counts()
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=1)
sns.countplot(x='Target',data=data)
plt.show()
X_train.shape
X_train.head()
y_test.shape
y_test.head()
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
max_accuracy = 0


for x in range(10):
   rf = RandomForestClassifier(random_state=x)
```

```
    rf.fit(X_train,y_train)
    y_pred_rf = rf.predict(X_test)
    current_accuracy = round(accuracy_score(y_pred_rf,y_test)*100,2)
    if(current_accuracy>max_accuracy):
        max_accuracy = current_accuracy
        best_x = x

#print(max_accuracy)
#print(best_x)

rf = RandomForestClassifier(random_state=best_x)
rf.fit(X_train,y_train)
Y_pred_rf = rf.predict(X_test)
if a==1:
    filename = 'randomforestfan.sav'
else:
    filename = 'randomforestlight.sav'
pickle.dump(rf, open(filename, 'wb'))
acc=(metrics.accuracy_score(y_pred_rf,y_test)*100)
print("Accuracy is:",acc)
```

Naïve Bayes

```
import numpy as np
import pandas as pd
from sklearn import metrics
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
```

```
import seaborn as sns
import pickle
a=int(input("Enter 1 for fan train 2 for light"))
if a==1:
   m="data_fan.csv"
else:
   m="data_light.csv"
data = pd.read_csv(m)
data.head()
data.shape
X = data.iloc[:,:-1]
X.head()
y = data.iloc[:,-1]
y.head()
data['Target'].value_counts()
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=1)
sns.countplot(x='Target',data=data)
plt.show()
X_train.shape
X_train.head()
y_test.shape
y_test.head()


from sklearn.naive_bayes import GaussianNB
model = GaussianNB()
model.fit(X_train,y_train)
```

```
if a==1:
   filename = 'naivefan.sav'
else:
   filename = 'naivelight.sav'
pickle.dump(model, open(filename, 'wb'))
y_pred = model.predict(X_test)
acc=(metrics.accuracy_score(y_pred,y_test)*100)
print("Accuracy is:",acc)
```

Decision Tree

```
import numpy as np
import pandas as pd
from sklearn import metrics
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
import pickle
a=int(input("Enter 1 for fan train 2 for light"))
if a==1:
   m="data_fan.csv"
else:
   m="data_light.csv"
data = pd.read_csv(m)
data.head()
data.shape
X = data.iloc[:,:-1]
X.head()
```

```
y = data.iloc[:,-1]
y.head()
data['Target'].value_counts()
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=1)
sns.countplot(x='Target',data=data)
plt.show()
X_train.shape
X_train.head()
y_test.shape
y_test.head()


from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
model = DecisionTreeClassifier()
model.fit(X_train,y_train)
if a==1:
   filename = 'dtfan.sav'
else:
   filename = 'dtlight.sav'
pickle.dump(model, open(filename, 'wb'))
y_pred = model.predict(X_test)
acc=(accuracy_score(y_pred,y_test)*100)
print("Accuracy is:",acc)
```

Face recognition and final working automated fan and light – Python Code

```
import urllib.request
import http
from time import sleep
import face_recognition
import cv2
from time import sleep
from openpyxl import Workbook
import datetime
import csv
import json
import time
import os
import urllib.request
import time
import os
import serial
base = "http://192.168.43.157/"
name=""
xyu=""
top = 4
right = 4
bottom = 4
left = 4

image_1 = face_recognition.load_image_file("Iswarya.jpeg")
image_1_face_encoding = face_recognition.face_encodings(image_1)[0]
```

```
image_2 = face_recognition.load_image_file("Anju.jpg")
image_2_face_encoding = face_recognition.face_encodings(image_2)[0]

known_face_encodings = [
   image_1_face_encoding,
   image_2_face_encoding
   ]
known_face_names = ["Iswarya","Anju"]
face_locations = []
face_encodings = []
face_names = []
video_capture = cv2.VideoCapture(0)
first_match_index="9"
gy=0
facei=0
while True:
        process_this_frame = True
        ret, frame = video_capture.read()
        small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)
        rgb_small_frame = small_frame[:, :, ::-1]
        name=""
        if process_this_frame:
                face_locations = face_recognition.face_locations(rgb_small_frame)
                face_encodings = face_recognition.face_encodings(rgb_small_frame,
face_locations)
                print(len(face_encodings))
                first_match_index=9
```

```
            for face_encoding in face_encodings:
                matches =
face_recognition.compare_faces(known_face_encodings, face_encoding)
                name = "Unknown"
                cv2.imwrite("temp.jpg",frame)
                if True in matches:
                    first_match_index = matches.index(True)
                    print(first_match_index)
                    name = known_face_names[first_match_index]
                face_names.append(name)
                process_this_frame = not process_this_frame
                for (top, right, bottom, left), name in zip(face_locations,
face_names):
                    top *= 4
                    right *= 4
                    bottom *= 4
                    left *= 4
            cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)
            cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255),
cv2.FILLED)
            font = cv2.FONT_HERSHEY_DUPLEX
            cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255, 255,
255), 1)
        cv2.imshow('Video', frame)
        face_names=[]
        ft=str(first_match_index)
        #print(name)
```

```
        if "Unknown" in name:
            facei=0
            os.system("python mail.py")
        elif "Iswarya" in name:
            facei=1
        elif "Anju" in name:
            facei=2
        gy=gy+1
        if gy>10:
            break
        cv2.waitKey(1)
video_capture.release()
cv2.destroyAllWindows()
def transfer(my_url):   #use to send and receive data
    try:
        n = urllib.request.urlopen(base + my_url).read()
        n = n.decode("utf-8")
        return n

    except http.client.HTTPException as e:
        return e

  #Send this data
g=""
h=""
r=""
ft=""
```

```
re=[]*2
k=0
res=""
while True:
    two = transfer(str(res))
    #print(two)
    r=str(two)
    print(r)
    sleep(1)
    k=k+1
    it=0
    re=[]
    if k>1:
        for ty in r:
            try:
                if '-' not in ty:
                    ft=ft+str(ty)
                else:
                    x=float(ft)
                    print(it)
                    re.append(x)
                    ft=""
                    it=it+1
                    if it==2:
                        break
            except:
                print("pass")
```

```
ft=""
print("TEMPERATURE")
print(re[0])
print("LDR")
print(re[1])
import pickle
filename = 'knnfan.sav'
loaded_model = pickle.load(open(filename, 'rb'))
a=float(re[0])
b=float(re[1])
Current_reports = [[facei,b,a]]
predicted = loaded_model.predict(Current_reports)
print(predicted[0])
h=str(predicted[0])
import pickle
filename = 'knnlight.sav'
loaded_model = pickle.load(open(filename, 'rb'))
a=float(re[0])
b=float(re[1])
Current_reports = [[facei,b,a]]
predicted = loaded_model.predict(Current_reports)
print(predicted[0])
g=str(int(predicted[0]))

res=g+h
```

PLAGIARISM REPORT

final phase-2 report

ORIGINALITY REPORT

18%	8%	7%	13%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to WorldQuant University Student Paper	2%
2	Submitted to University of Essex Student Paper	1%
3	www.researchgate.net Internet Source	1%
4	towardsdatascience.com Internet Source	1%
5	www.coursehero.com Internet Source	1%
6	Submitted to Coventry University Student Paper	1%
7	Submitted to Universiti Teknikal Malaysia Melaka Student Paper	1%
8	Submitted to Midlands State University Student Paper	<1%
9	Submitted to SASTRA University Student Paper	<1%

SUBMISSION PROOF

Gmail iswarya mayilsamy <iswaryamayilsamy235@gmail.com>

Customization of Lighting and Ventilation System in Household

1 message

prof prince <conferenceiccet@gmail.com> Thu, Mar 30, 2023 at 8:06 PM
To: iswarya mayilsamy <iswaryamayilsamy235@gmail.com>, prasnthp.080902@gmail.com, anjuarumugam1@gmail.com

Dear Author/s,

We are happy to inform you that your paper, submitted for the ICCET 2023 conference has been **Accepted** based on the recommendations provided by the Technical Review Committee. By this mail you are requested to proceed with Registration for the Conference. Most notable is that the Conference must be registered on or before **APRIL 10, 2023** from the date of acceptance.

www.iccet.in

Kindly fill the registration form, declaration form which is attached with the mail and it should reach us on above mentioned days.

Instructions to fill the forms:

- **Fill** the registration form given in the excel sheet and send it back to us in **excel format only**.
- **Print** the Declaration form (Attachment- page number 4) alone, fill in the details, sign the form, scan the form and send the details in image/pdf format.
- Ensure to send **payment screenshots** and send all the details once the payment has been done to the account.
- All the above completed details should be mailed to **conferenceiccet@gmail.com**
- Please send a soft copy of the RESEARCH PAPER in word format only.

NOTE: - Send Abstract and Full paper separately in word format only.

We reserve the right to reject your paper if the registration is not done within the above said number of days.

Journal details, account details are given in the attachment

Conference registration certificates will be provided within 34 hours from date of Registration

Paper id: ICCET230400

ICCET 2023
www.iccet.in
9600034378

2 attachments

Registration form - ICCET 2023.xls
26K

Registration Instructions & Declaration form - 11TH ICCET 2023.doc
427K