

# **SMART FAN SYSTEM**

## **INTERDISCIPLINARY PROJECT**

Submitted in partial fulfillment of the requirements for the award of  
Bachelor of Technology degree in Information Technology

by

**ARVIND B (41120030)**



**DEPARTMENT OF INFORMATION TECHNOLOGY**

**SCHOOL OF COMPUTING**

# **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**Category-1 University by UGC**

**Accredited with Grade “A++” by NAAC**

**12B Status by UGC |Approved by AICTE**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

**MAY 2024**



# SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE  
[www.sathyabama.ac.in](http://www.sathyabama.ac.in)

## DEPARTMENT OF INFORMATION TECHNOLOGY

### BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **ARVIND B (41120030)** who carried out the project entitled "**SMART FAN SYSTEM**" under my supervision from **JANUARY 2024** to **MAY 2024**.

*Dr. D. Adhimuga Sivasakthi 2/5*

Internal Guide

**Dr. D. ADHIMUGA SIVASAKTHI**

*R.W.*

Head of The Department

**Dr. R. SUBHASHINI, M.E., Ph.D.**

Submitted for viva voce Examination held on

*8/5*

*[Signature]*

Internal Examiner

*[Signature]*

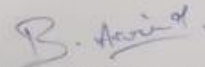
External Examiner

## DECLARATION

I, ARVIND B (41120030), hereby declare that the Project Report entitled "SMART FAN SYSTEM" done by me under the guidance of Dr. D. Adhimuga Sivasakthi, is submitted in partial fulfillment of the requirements for the award Bachelor of Technology degree in Information Technology.

DATE: 08/05/2024

PLACE: CHENNAI



SIGNATURE OF THE CANDIDATE

## ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D.**, Dean School of Computing, **Dr. R. Subhashini, M.E., Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. D. Adhimuga SivaSakthi** for her valuable guidance, suggestions and constant encouragement which paved the way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

# CERTIFICATE OF COMPLETION



149, 1C, 2nd Floor, Mount Poonamallee Rd,  
Ramapuram, Chennai - 600089.  
admin@hackwittechnologies.com  
www.hackwittechnologies.com

10.02.2024

## CERTIFICATE FOR INTERNSHIP

TO WHOMSOEVER IT MAY CONCERN

This is to Certify that **ARVIND B (Reg No: 41120030)** student of Sathyabama Institute of Science and Technology, B. Tech Information Technology, has successfully completed the Internship at Hackwit Technologies from 22<sup>nd</sup> January to 30<sup>th</sup> January 2024.

During the internship, **ARVIND B** has closely worked as a part of the **Internet of Things** by carrying out the project entitled "**Temperature-Based Fan Speed Controller Using Arduino**".

He demonstrated good developing and learning skill with self-motivated attitude to learn new things. His contribution to implement the Internet of Things was good. We wish him all the best for his future endeavors.

Best Regards,



Ravikumar S  
CEO & Managing Director  
Hackwit Technologies Pvt Ltd



## CERTIFICATE OF COMPLETION



## **ABSTRACT**

The Smart Fan System represents a groundbreaking convergence of IoT technology and environmental regulation, providing an inventive approach to maintaining optimal ambient conditions across diverse environments. Through seamless integration of the ESP32 microcontroller, DHT11 sensor, relay, DC fan, breadboard, and jumper wires, this project heralds a significant shift in how we engage with our living spaces. At its core, the system harnesses the precision of the DHT11 sensor's temperature and humidity sensing capabilities, bolstered by the computational capabilities of the ESP32 microcontroller, to offer real-time monitoring and control. Facilitated by seamless communication with the Blynk IoT platform, users are empowered with remote access to environmental data, enabling informed decision-making regarding their surroundings. The system's standout feature lies in its intelligent fan control logic, dynamically adjusting fan operation in response to predefined temperature thresholds, thereby ensuring optimal comfort and fostering energy efficiency. This sophisticated algorithm not only enhances user experience but also contributes significantly to a more sustainable future. In an era characterized by the complexities of modern living, projects like the Smart Fan System emerge as beacons of innovation, exemplifying the transformative potential of IoT technology. By leveraging the synergy of connectivity and intelligence, this project offers a glimpse into a future where our environments seamlessly adapt to our needs, creating spaces that are not only functional but genuinely intelligent. Additional features include customizable settings for personalized control, data logging capabilities for trend analysis, integration with smart home systems for enhanced automation, alert notifications for timely intervention, and compatibility with renewable energy sources for eco-conscious operation. The system's modular design facilitates expandability and ease of installation, while its user-friendly interface ensures accessibility for individuals with varying technical backgrounds. Community and developer support further fosters innovation and continuous improvement, contributing to the evolution of environmental control solutions. Through its multifaceted approach, the Smart Fan System embodies a holistic vision of intelligent living spaces, paving the way for a future where technology enhances our quality of life while promoting sustainability.

## TABLE OF CONTENTS

| Chapter No. | TITLE                                | Page No. |
|-------------|--------------------------------------|----------|
|             | <b>ABSTRACT</b>                      | vii      |
|             | <b>LIST OF ABBREVIATIONS</b>         | x        |
|             | <b>LIST OF FIGURES</b>               | xi       |
| 1           | <b>INTRODUCTION</b>                  | 1        |
|             | 1.1 Study Area                       | 2        |
|             | 1.2 Existing System                  | 3        |
|             | 1.3 Open Problems in Existing System | 5        |
|             | 1.4 Proposed System                  | 6        |
|             | 1.5 Objectives                       | 8        |
| 2           | <b>METHODOLOGY</b>                   | 9        |
|             | 2.1 Components Required              | 9        |
|             | 2.2 Architecture                     | 13       |
|             | 2.2.1 Hardware Architecture          | 14       |
|             | 2.2.2 Software Architecture          | 17       |
|             | 2.3 Modules                          | 20       |
|             | 2.4 Algorithm                        | 21       |
|             | 2.5 Project Scope                    | 23       |
|             | 2.6 Project Risks                    | 24       |
|             | 2.7 Project Mitigation               | 25       |
|             | 2.8 Testing Plan                     | 27       |

|   |                                   |    |
|---|-----------------------------------|----|
| 3 | <b>RESULTS AND DISSCUSSIONS</b>   | 28 |
| 4 | <b>CONCLUSION AND FUTURE WORK</b> | 31 |
|   | 4.1    Conclusion                 | 31 |
|   | 4.2    Future Work                | 32 |
| 5 | <b>REFERENCES</b>                 | 33 |
|   | <b>APPENDICES</b>                 | 35 |
|   | A    Sample Code                  | 35 |
|   | B    Screenshot                   | 37 |

## LIST OF ABBREVIATIONS

| ABBREVIATION | EXPANSION                            |
|--------------|--------------------------------------|
| IoT          | - Internet of Things                 |
| DHT11        | - Digital Humidity and Temperature   |
| DC           | - Direct Current                     |
| ESP32        | - Embedded System Platform           |
| WiFi         | - Wireless Fidelity                  |
| IDE          | - Integrated Development Environment |
| GPIO         | - General-Purpose Input/Output       |
| UAT          | - User Acceptance Testing            |
| VCC          | - Voltage Common Collector           |
| GND          | - Ground                             |
| NO           | - Normally Open                      |
| NC           | - Normally Closed                    |
| COM          | - Common                             |
| RTC          | - Real Time Clock                    |
| SFS          | - Smart Fan System                   |

## LIST OF FIGURES

| <b>Figure No.</b> | <b>Name of Figures</b>                                                                | <b>Page No.</b> |
|-------------------|---------------------------------------------------------------------------------------|-----------------|
| 2.1               | ESP32 Microcontroller                                                                 | 9               |
| 2.2               | DHT11 Sensor                                                                          | 10              |
| 2.3               | Relay Module                                                                          | 10              |
| 2.4               | DC Fan                                                                                | 11              |
| 2.5               | Arduino IDE                                                                           | 12              |
| 2.6               | Blynk App                                                                             | 12              |
| 2.7               | Hardware Architecture of Smart Fan System (SFS)                                       | 14              |
| 2.8               | Circuit Diagram of Smart Fan System (SFS)                                             | 16              |
| 2.9               | Software Architecture of Smart Fan System (SFS)                                       | 17              |
| 3.1               | Smart Fan System Model                                                                | 28              |
| 3.2               | The Temperature and Humidity are Being Displayed on the Serial Monitor of Arduino IDE | 29              |
| 3.3               | The Temperature and Humidity are Being Displayed on the Blynk App                     | 30              |
| B.1               | The Temperature and Humidity are Being Displayed on the Serial Monitor of Arduino IDE | 37              |
| B.2               | Blynk App Working                                                                     | 37              |

# **CHAPTER 1**

## **INTRODUCTION**

In a world increasingly driven by technology, the marriage of Internet of Things (IoT) concepts with everyday devices has opened doors to a realm of innovative solutions aimed at enhancing our daily lives. One such groundbreaking innovation is the Smart Fan System – a testament to the transformative power of IoT in revolutionizing how we interact with our living environments.

At its core, the Smart Fan System represents a fusion of cutting-edge hardware components and sophisticated software algorithms, meticulously orchestrated to deliver an unparalleled experience in environmental control. With components including the ESP32 microcontroller, DHT11 sensor, relay, DC fan, breadboard, and jumper wires, this system embodies the essence of IoT innovation – seamlessly connecting devices and leveraging data to drive intelligent decision-making.

The factor that distinguishes the Smart Fan System(SFS) is its ability to address the inherent limitations of traditional environmental control systems. By harnessing the capabilities of interconnected devices and real-time data analysis, the system empowers users to exert precise control over ambient conditions. Whether it's maintaining optimal room temperature, monitoring humidity levels, or optimizing energy consumption, the Smart Fan System offers a level of customization and efficiency previously unimaginable.

In this exploration, we delve into the intricacies of the Smart Fan System – from its conception to its real-world applications. By delving into its architecture, functionality, and impact, we gain insight into how this project is reshaping the landscape of environmental control systems and paving the way for a more connected, intelligent, and sustainable future.

## **1.1. STUDY AREA**

The study area for the Smart Fan System encompasses a wide range of domains, each contributing to the development, implementation, and impact of the project. From hardware components to software architecture, and from environmental science to IoT technology, various disciplines intersect to create a comprehensive understanding of the system.

### **i) Hardware Components**

The study area begins with a focus on the hardware components used in the Smart Fan System. This includes the ESP32 microcontroller, DHT11 sensor, relay, DC fan, breadboard, and jumper wires. Understanding the specifications, capabilities, and interconnections of these components is crucial for the successful implementation of the system.

**ii) Sensor Technology:** Within the hardware domain, a specific focus is placed on sensor technology, particularly the DHT11 sensor. This involves studying the principles of temperature and humidity sensing, as well as the accuracy and reliability of the sensor in real-world applications.

### **iii) Microcontroller Programming**

Another key aspect of the study area is microcontroller programming, with a focus on the ESP32 microcontroller. This involves understanding the programming language (e.g., Arduino IDE), as well as the functionalities and libraries available for interfacing with sensors, controlling peripherals, and communicating with external platforms.

### **iv) IoT Platform**

A significant portion of the study area is dedicated to IoT platforms, particularly Blynk in the case of the Smart Fan System. This involves studying the features, functionalities, and APIs of the platform, as well as the process of connecting hardware devices, creating user interfaces, and implementing remote monitoring and control capabilities.

#### **v) Environmental Science**

In addition to technical aspects, the study area extends to environmental science, focusing on the principles of temperature regulation, humidity control, and energy efficiency. Understanding the relationship between environmental factors and human comfort is essential for designing effective control algorithms for the Smart Fan System.

#### **vi) System Integration**

Finally, the study area encompasses system integration, where all components and functionalities are brought together to create a cohesive solution. This involves testing, debugging, and optimizing the system to ensure reliable performance under various conditions.

### **1.2. EXISTING SYSTEM**

The existing system of environmental control typically involves traditional methods of temperature regulation and airflow management, often relying on manual intervention or basic automation solutions. Here's an overview of the characteristics and limitations of the existing systems:

#### **i) Manual Control**

In many cases, environmental control systems rely on manual adjustments by users, such as turning on/off fans or adjusting thermostat settings. Users need to actively monitor environmental conditions and manually intervene to maintain comfort levels, which can be inconvenient and time-consuming.

#### **ii) Basic Automation**

Some existing systems may incorporate basic automation features, such as programmable thermostats or timers for fan operation. However, these automation features are often limited in functionality and may not provide real-time monitoring or adaptive control based on changing environmental conditions.

### **iii) Limited Connectivity**

Traditional environmental control systems typically lack connectivity to external platforms or IoT networks, limiting remote monitoring and control capabilities.

Users have limited visibility into environmental conditions when away from the premises, making it challenging to respond to changes or emergencies promptly.

### **iv) Energy Inefficiency**

Without intelligent control algorithms or real-time data monitoring, existing systems may operate inefficiently, leading to unnecessary energy consumption and higher utility bills. Lack of dynamic control based on actual environmental conditions may result in overcooling or overheating, wasting energy and reducing overall efficiency.

### **v) Scalability and Adaptability**

Traditional environmental control systems may lack scalability and adaptability to accommodate changes in building layouts, occupancy patterns, or environmental requirements. Retrofitting or upgrading existing systems to meet evolving needs can be costly and disruptive, requiring extensive modifications or replacement of components.

### **vi) Limited Integration**

Integration with other smart home or building automation systems may be limited or non-existent, reducing interoperability and synergy between different subsystems. Lack of integration hampers the potential for holistic environmental control and optimization across multiple systems or devices.

### **vii) Inadequate Disaster Preparedness:**

Traditional systems may lack fail-safe mechanisms or automated responses to environmental emergencies such as fires, floods, or power outages. Absence of backup systems or emergency protocols may increase the risk of property damage or occupant safety hazards during unforeseen events.

### **1.3. OPEN PROBLEMS IN EXISTING SYSTEM**

#### **i) Limited Automation**

Many existing systems rely on manual intervention or basic automation, which may not adequately respond to changing environmental conditions in real-time. This can lead to suboptimal energy usage and comfort levels, especially in dynamic environments.

#### **ii) Lack of Granularity**

Centralized HVAC systems often lack granularity in control, leading to inefficiencies in energy usage and comfort. Different areas of a building may have varying temperature or humidity requirements, but centralized systems may not be able to adjust settings for each zone independently.

#### **iii) High Cost**

The installation and maintenance costs associated with centralized HVAC systems or proprietary solutions can be prohibitive for small businesses or residential users. This limits access to advanced environmental control technology and hinders adoption.

#### **iv) Complexity**

Proprietary solutions or advanced automation systems may be complex to install, configure, and maintain. This complexity can lead to higher training costs, longer deployment times, and increased risk of system malfunctions or errors.

#### **v) Limited Connectivity**

Some existing systems lack connectivity features, making it difficult to monitor and control environmental conditions remotely. This can be a significant drawback for users who require flexibility and accessibility in managing their systems.

#### **vi) Scalability Issues**

Existing systems may have limited scalability, making it challenging to expand or upgrade them to meet evolving needs.

#### **vii) Energy Inefficiency**

Traditional environmental control systems may not optimize energy usage based on real-time conditions, leading to unnecessary energy consumption and increased operational costs. This is particularly relevant in today's energy-conscious environment.

#### **viii) Scalability Issues**

Existing systems may have limited scalability, making it challenging to expand or upgrade them to meet evolving needs. This can result in outdated technology and suboptimal performance over time.

#### **xi) Environmental Impact**

Many conventional HVAC systems rely heavily on fossil fuels or consume large amounts of electricity, contributing to greenhouse gas emissions and environmental degradation. Finding sustainable and eco-friendly alternatives is becoming increasingly important.

### **1.4. PROPOSED SYSTEM**

#### **i) Integrated IoT Platform**

The Smart Fan System integrates seamlessly with the Blynk IoT platform, providing a centralized hub for remote monitoring and control. Users can access the system from anywhere via the Blynk app, ensuring convenience and accessibility.

#### **ii) Sensor-Based Environmental Monitoring**

The system utilizes a DHT11 sensor to continuously monitor temperature and humidity levels in the environment. This real-time data collection enables precise environmental control and ensures optimal comfort conditions. The DHT11 sensor is an affordable option for measuring temperature and humidity, making it accessible for hobbyists, educational projects, and small-scale applications.

### **iii) Intelligent Fan Control Logic**

Advanced algorithms implemented on the ESP32 microcontroller analyze the sensor data and determine optimal fan operation. The system dynamically adjusts fan speed based on environmental conditions, user preferences, and predefined thresholds.

### **iv) Adaptive Threshold Management**

The Smart Fan System employs adaptive threshold management to ensure efficient and responsive fan control. Threshold values for temperature and humidity can be customized based on user preferences and environmental requirements, allowing for flexibility and adaptability.

### **v) Energy-Efficient Operation**

By intelligently adjusting fan operation based on real-time sensor data, the system maximizes energy efficiency while maintaining comfort levels. The use of variable fan speed control and hysteresis algorithms minimizes energy consumption and reduces operational costs.

### **vi) User-Friendly Interface**

The Blynk app provides an intuitive and user-friendly interface for interacting with the Smart Fan System. Users can easily monitor environmental conditions, adjust fan settings, and receive notifications or alerts directly on their mobile devices.

### **vii) Scalability and Integration**

The modular design of the Smart Fan System allows for easy scalability and integration with additional sensors or actuators. It can be customized to meet specific use cases or environmental requirements, making it suitable for a wide range of applications.

### **viii) Cost-Effective Solution**

Compared to centralized HVAC systems, the Smart Fan System offers a cost-effective alternative for environmental control due to its simplicity.

## **1.5. OBJECTIVES**

The objectives of the Smart Fan System are to

### **i) Monitor Environmental Conditions**

Measure temperature and humidity levels in real-time using the DHT11 sensor. Continuously monitor environmental parameters to ensure comfort and well-being.

### **ii) Implement Intelligent Fan Control**

Automatically adjust fan speed or operation based on environmental conditions. Implement intelligent algorithms to optimize fan control for energy efficiency and comfort.

### **iii) Provide Remote Monitoring and Control**

Enable users to remotely monitor environmental conditions and fan operation via the Blynk IoT platform. Allow users to adjust fan settings, thresholds, and operating modes remotely for convenience and flexibility.

### **iv) Ensure Energy Efficiency**

Optimize fan operation to minimize energy consumption while maintaining desired comfort levels. Implement features such as variable fan speed control and hysteresis to reduce energy wastage and operational costs.

### **v) Enhance User Experience**

Provide a user-friendly interface through the Blynk app for easy monitoring and control of the Smart Fan System. Ensure reliability, responsiveness, and ease of use to enhance user satisfaction and adoption.

### **vi) Ensure Safety and Reliability**

Incorporate safety features such as isolation relays to protect the system and users from electrical hazards. Design the system to operate reliably under various environmental conditions without compromising safety.

## CHAPTER 2

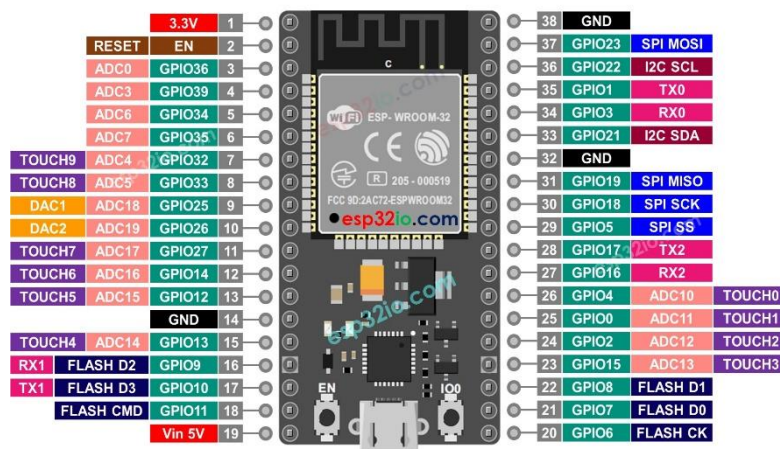
### METHODOLOGY

#### 2.1. COMPONENTS REQUIRED

##### i) HARDWARE REQUIREMENTS

- **ESP32 Development Board**

The ESP32 development board is the heart of the system, serving as the main controller. It features a powerful microcontroller unit (MCU) with built-in Wi-Fi and Bluetooth connectivity, making it ideal for IoT applications. The ESP32 board provides GPIO pins for interfacing with sensors, relays, and other peripheral devices. It can be programmed using the Arduino IDE or other development environments, offering flexibility and ease of implementation.

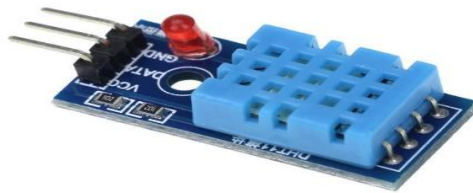


**Fig 2.1 ESP32 Microcontroller**

It communicates with other system components to adjust fan speed accordingly and facilitates remote control via IoT platforms like Blynk or MQTT, allowing users to monitor and adjust settings from anywhere via a smartphone app or web interface. Overall, the ESP32 enables intelligent fan control and remote accessibility, enhancing the system's functionality and user experience.

- **DHT11 Temperature and Humidity Sensor**

The DHT11 sensor is a digital sensor that accurately measures temperature and humidity levels in the surrounding environment. It features a calibrated digital signal output, simplifying interfacing with the ESP32 development board. The sensor is low-cost and easy to use, making it suitable for a wide range of temperature and humidity monitoring applications.



**Fig 2.2 DHT11 Sensor**

- **Relay Module**

The relay module serves as the interface between the low-power ESP32 microcontroller and the high-power DC fan. It typically consists of one or more electromechanical relays capable of switching AC or DC loads. The relay module provides isolation between the microcontroller and the fan circuit, ensuring safety and reliability.



**Fig 2.3 Relay Module**

- **DC Fan**

The DC fan is responsible for generating airflow to regulate temperature in the environment. It operates on direct current (DC) power and is available in various sizes and configurations, depending on the cooling requirements. Choose a fan with appropriate specifications (voltage, current, and airflow) to suit your application needs.



**Fig 2.4 DC Fan**

- **Breadboard and Jumper Wires**

The breadboard provides a convenient platform for prototyping and connecting the hardware components. It features interconnected metal strips that allow components to be easily plugged in and arranged without soldering. Jumper wires are used to establish electrical connections between the ESP32, sensors, relay module, and DC fan, ensuring proper functionality and signal transmission.

- **Power Supply**

The power supply provides the necessary electrical power to the ESP32, sensors, relay module, and DC fan. Depending on the power requirements of the components, you can use a USB power adapter, batteries, or a dedicated power supply unit. Ensure the power supply can deliver stable and sufficient power to all components to avoid voltage fluctuations or power failures.

## ii) SOFTWARE REQUIREMENTS

- **Arduino IDE**

The Arduino Integrated Development Environment (IDE) is used for writing, compiling, and uploading code to the ESP32 microcontroller. It provides a user-friendly interface for programming the microcontroller using the C++ programming language.



**Fig 2.5 Arduino IDE**

- **Blynk Application**

Blynk is a platform for building IoT applications that allows users to control hardware remotely over the internet. In this project, the Blynk app is used to monitor temperature and humidity readings from the DHT11 sensor and to control the fan speed.



**Fig 2.6 Blynk App**

- **Blynk Library for ESP32**

The Blynk library for ESP32 provides the necessary functions to connect the ESP32 microcontroller to the Blynk cloud server. It allows the ESP32 to send sensor data to the Blynk app and receive commands from the app to adjust fan speed.

- **DHT Sensor Library**

The DHT sensor library provides functions for interfacing with DHT series sensors, such as the DHT11 used in this project. It enables the ESP32 to read temperature and humidity values from the sensor.

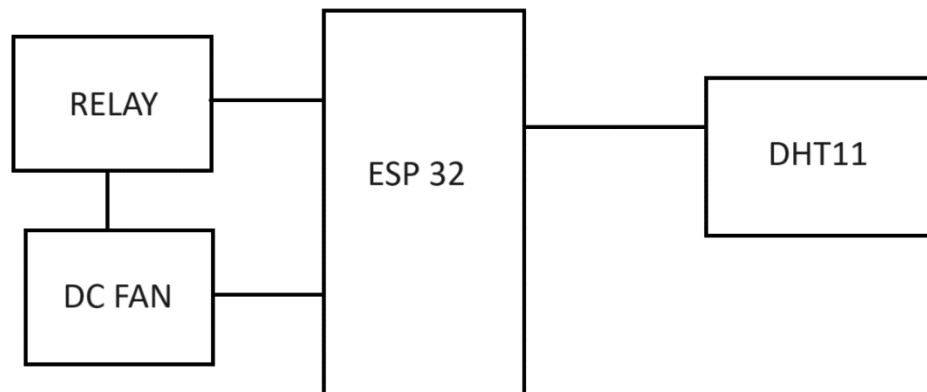
## **2.2. ARCHITECTURE**

The architecture of the Smart Fan System is built upon a foundation of both hardware and software components working in harmony to achieve its objectives. At its heart lies the ESP32 microcontroller, serving as the brain of the system. The ESP32 is responsible for interfacing with the various hardware peripherals, including the DHT11 sensor for temperature and humidity sensing and the relay for controlling the DC fan.

This firmware encompasses code to initialize the hardware components, read sensor data, implement fan control logic based on predefined thresholds, and communicate with the Blynk IoT platform. The Blynk platform, hosted on the cloud, serves as the bridge between the Smart Fan System and the user, providing a user-friendly interface for remote monitoring and control.

Communication between the ESP32 and the Blynk server is facilitated through Wi-Fi connectivity, enabling seamless integration with the Blynk platform. This allows users to monitor real-time temperature and humidity values remotely via the Blynk app and control the fan operation as needed. The data flow follows a path from sensor data collection by the ESP32, local processing to determine fan control actions, transmission of processed data to the Blynk server, and finally, user interaction through the Blynk app.

### 2.2.1. HARDWARE ARCHITECTURE



**Fig 2.7 Hardware Architecture of Smart Fan System (SFS)**

#### i) ESP32 Microcontroller

- **Central Processing Unit:** The ESP32 serves as the brain of the system, executing the firmware code and coordinating all operations.
- **Interfacing with Sensors:** It interacts with the DHT11 sensor to receive temperature and humidity readings, enabling real-time environmental monitoring.
- **Data Processing:** Once sensor data is acquired, the ESP32 processes this data using its onboard processing capabilities. It runs algorithms to analyze the temperature and humidity levels in real-time.
- **Controlling the Fan Relay:** The ESP32 sends control signals to the relay, dictating when to activate or deactivate the DC fan based on predefined thresholds or user commands.
- **Communicating with Blynk IoT Platform:** It establishes Wi-Fi connectivity to communicate with the Blynk server, facilitating remote monitoring and control of the system via the Blynk app.
- **Integration with IoT Platform:** Additionally, the ESP32 microcontroller may be integrated with an IoT platform such as Blynk or MQTT for remote monitoring and control.

## ii) DHT11 Sensor

- **Temperature and Humidity Sensing:** The DHT11 sensor accurately measures temperature and humidity levels in the environment, providing crucial data for environmental monitoring and control.
- **Analog Data Transmission:** It converts temperature and humidity readings into analog signals, which are then transmitted to the ESP32 microcontroller for processing.
- **Reliability and Accuracy:** Known for its reliability and accuracy, the DHT11 sensor ensures precise environmental data collection, essential for effective control of the Smart Fan System.

## iii) Relay

- **Controlled Fan Operation:** The relay acts as a switch, controlling the operation of the DC fan based on commands received from the ESP32 microcontroller.
- **Electromechanical Switching:** When activated, the relay allows electrical current to flow to the DC fan, causing it to spin and regulate airflow in the environment.
- **Reliability and Durability:** Designed for robustness, the relay ensures reliable fan control, even under varying environmental conditions and continuous operation.

## iv) DC Fan

- **Airflow Regulation**  
The DC fan is responsible for actuating airflow within the environment, facilitating temperature regulation and ensuring user comfort.
- **Efficient Operation**  
Designed for energy efficiency, the DC fan optimizes power consumption while providing adequate airflow to maintain optimal environmental conditions.
- **Installation and Mounting:** The DC fan is installed and mounted in a suitable location within the environment, such as on a ceiling or wall, to facilitate optimal air circulation

## v) Breadboard and Jumper Wires

- **Physical Connectivity**

The breadboard and jumper wires provide physical connections between the components on the hardware setup, facilitating ease of assembly, testing, and modification.

- **Modularity and Flexibility**

Their modular design allows for quick prototyping and experimentation, enabling developers to iterate on the system's design and functionality efficiently.

- **Organized Wiring**

By organizing wiring on the breadboard, jumper wires ensure a tidy and organized hardware setup, reducing the risk of electrical shorts or malfunctions.

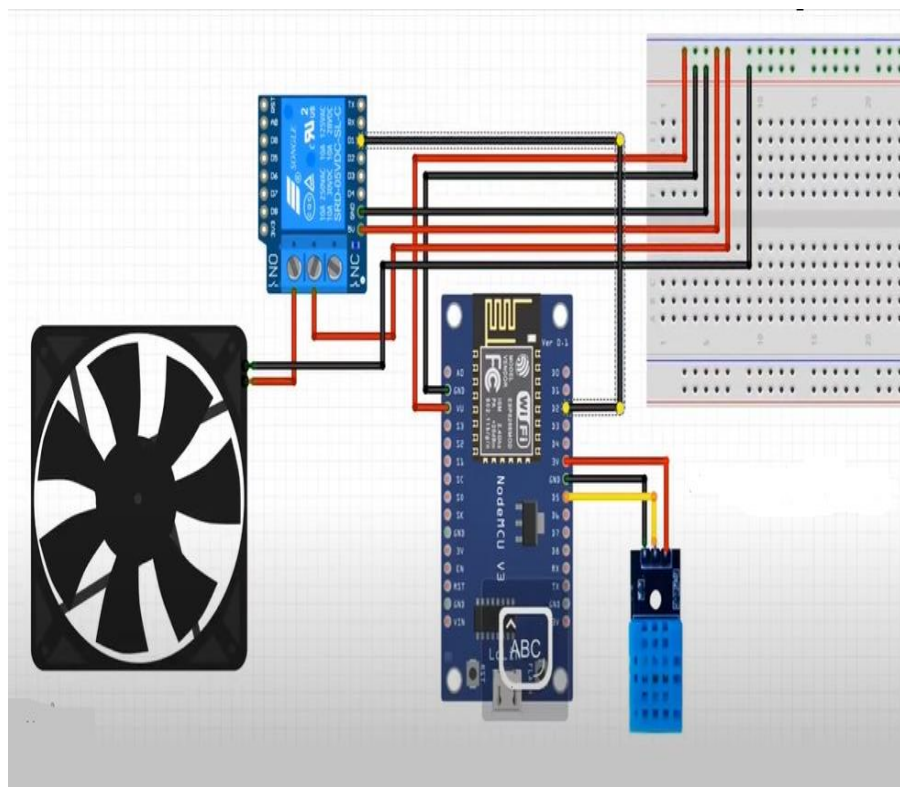


Fig 2.8 Circuit Diagram of Smart Fan System (SFS)

## 2.2.2. SOFTWARE ARCHITECTURE

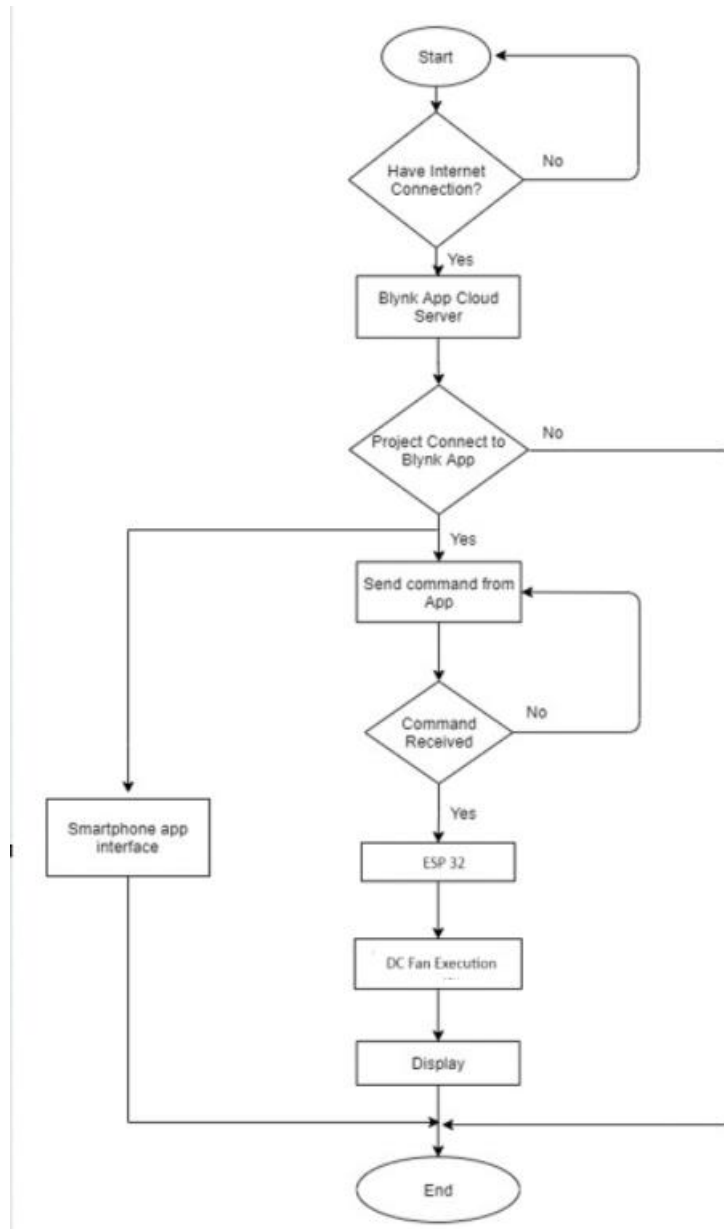


Fig 2.9 Software Architecture of Smart Fan System (SFS)

## **i) ESP32 Firmware**

- **Core Logic**

The firmware running on the ESP32 microcontroller serves as the backbone of the system, implementing core functionalities such as sensor data acquisition, fan control logic, and communication protocols.

- **Arduino IDE**

Developed using the Arduino Integrated Development Environment (IDE), the firmware is written in C/C++ programming languages, leveraging Arduino libraries and functions for hardware interfacing and data processing.

- **Sensor Data Acquisition**

The firmware includes code to interface with the DHT11 sensor, reading temperature and humidity values at regular intervals and storing them in variables for further processing.

- **Fan Control Logic**

Based on predefined temperature thresholds or user-defined settings, the firmware implements control logic to determine when to activate or deactivate the DC fan relay, ensuring optimal environmental conditions.

## **ii) Blynk IoT Platform**

- **Cloud-Based Infrastructure**

Hosted on the cloud, the Blynk IoT platform provides a centralized hub for remote monitoring and control of the Smart Fan System.

- **User Interface**

Blynk offers a user-friendly interface, accessible via the Blynk mobile app or web dashboard, allowing users to visualize real-time sensor data, set fan control parameters, and receive notifications or alerts.

- **API Integration**

The ESP32 firmware communicates with the Blynk server using Blynk's API, enabling bidirectional data exchange and synchronization between the hardware and the Blynk app.

- **Event Handling**

Blynk supports event-driven programming, allowing the firmware to respond to user inputs or system events in real-time, such as adjusting fan speed or sending notifications when temperature thresholds are exceeded.

### iii) Communication Protocols

- **Wi-Fi Connectivity**

The ESP32 firmware establishes a Wi-Fi connection to the local network, enabling communication with the Blynk server and other devices connected to the network.

- **HTTP/HTTPS**

Data exchange between the ESP32 and the Blynk server occurs over HTTP or HTTPS protocols, ensuring secure and reliable communication over the internet.

- **WebSocket**

Blynk utilizes WebSocket protocol for real-time communication between the ESP32 and the Blynk app, enabling instantaneous updates and notifications on the user interface.

### iv) Data Flow

- **Data Collection**

Sensor data (temperature and humidity readings) are collected by the ESP32 firmware from the DHT11 sensor at regular intervals.

- **Data Processing**

The firmware processes the collected data, applying control logic to determine fan operation based on predefined thresholds or user inputs.

- **Data Transmission**

Processed data is transmitted to the Blynk server over Wi-Fi using HTTP/HTTPS protocols, allowing users to monitor real-time sensor data and control the fan operation remotely via the Blynk app.

## **2.3. MODULES**

The Smart Fan System integrates hardware modules like ESP32, DHT11 sensor, relay, DC fan, and breadboard, offering intelligent environmental control. Harnessing IoT technology, it optimizes comfort and efficiency in living and working spaces. The modules are explained below

### **i) ESP32 Module**

- The ESP32 module serves as the core of the system, housing the central processing unit (CPU), memory, and a variety of input/output (I/O) pins.
- It supports both Wi-Fi and Bluetooth connectivity, enabling communication with external devices and networks.
- Programmed using development environments like Arduino IDE, it offers extensive built-in peripherals and interfaces for sensor integration and control.
- Additionally, it provides real-time clock (RTC) functionality, ensuring accurate timekeeping in applications.

### **ii) DHT11 Sensor Module**

- The DHT11 sensor module utilizes a digital sensor to measure temperature and humidity with high accuracy and reliability.
- Outputting data in digital format simplifies interfacing with microcontrollers, requiring minimal external components for operation.
- Some modules may feature a built-in calibration function to further enhance accuracy.
- Its simplicity and reliability make it a popular choice for environmental sensing applications.

### **iii) Relay Module**

- Comprising one or more electromechanical relays, it switches high-power electrical loads while providing isolation between control signals and load circuits. Available in single or multiple channel configurations, it supports both normally open (NO) and normally closed (NC) contacts for versatile switching.

#### **iv) DC Fan Module**

- Equipped with a DC motor and fan blades, the DC fan module generates airflow to regulate temperature in the environment.
- Available in different sizes and configurations, it operates on direct current (DC) power sources, typically 5V or 12V.
- Some modules may include features such as variable speed control or PWM support for precise airflow adjustment.
- Designed for quiet operation and long-term reliability, it is ideal for cooling applications in various environments.

#### **v) Breadboard Module**

- Consisting of a plastic base with interconnected metal strips, the breadboard module provides a platform for prototyping and testing electronic circuits without soldering.
- Featuring multiple power rails and distribution points, it facilitates easy connection of components and rearrangement of circuit layouts.
- Available in different sizes and configurations, it offers flexibility and convenience during circuit development and experimentation.

### **2.4. ALGORITHM**

The algorithm of the Smart Fan System orchestrates the interaction between hardware components, sensor data processing, and fan control logic. Let's outline the key steps of the algorithm:

#### **i) Initialization:**

- Initialize the ESP32 microcontroller, DHT11 sensor, relay, and other hardware components.
- Establish Wi-Fi connectivity to enable communication with the Blynk IoT platform.
- The ESP32 module serves as the core of the system, housing the central processing unit (CPU), memory, and a variety of input/output (I/O) pins.

#### **ii) Sensor Data Acquisition:**

- Periodically read temperature and humidity values from the DHT11 sensor.
- Store the sensor readings in variables for further processing.

#### **iii) Data Processing:**

- Compare the temperature reading with predefined thresholds to determine fan control actions.
- If the temperature exceeds the upper threshold, activate the relay to turn on the DC fan.
- If the temperature falls below the lower threshold, deactivate the relay to turn off the DC fan.
- Optionally, implement hysteresis to prevent rapid cycling of the fan due to small fluctuations in temperature.

#### **iv) Communication with Blynk IoT Platform:**

- Transmit processed sensor data to the Blynk server over Wi-Fi.
- Update the Blynk app interface with real-time temperature and humidity values.
- Receive user commands from the Blynk app to adjust fan control parameters or settings.

#### **v) Feedback Loop**

- Continuously monitor temperature and humidity values to maintain desired environmental conditions.
- Provide users with channels to submit feedback, report issues, and seek assistance from customer support.
- Adjust fan operation based on changes in sensor readings and user inputs from the Blynk app.
- Provide visual feedback to the user through the Blynk app interface, indicating the current fan status and environmental conditions.

#### **vi) Error Handling**

- Implement error handling mechanisms to detect and address sensor failures, communication errors, or other unexpected events.
- Notify the user of any errors or malfunctions through the Blynk app interface or other means of communication.

#### **vii) Power Management**

- Optimize power consumption by utilizing low-power modes when the system is idle or not actively processing sensor data.
- Ensure efficient utilization of battery or power source to prolong operational lifespan and reduce energy costs.

#### **viii) Logging and Analytics**

- Optionally, log sensor data and system events for historical analysis or troubleshooting purposes.
- Implement analytics to identify trends, patterns, or anomalies in environmental conditions and fan operation over time.

### **2.5. PROJECT SCOPE**

#### **i) Functionality**

The Smart Fan System will automatically adjust the speed of a DC fan based on real-time temperature readings from a DHT11 sensor. When the temperature rises above a predefined threshold, the fan speed will increase to enhance cooling, ensuring a comfortable environment. Conversely, if the temperature decreases below the threshold, the fan speed will decrease to conserve energy while maintaining optimal conditions.

#### **ii) Hardware Implementation**

The project will utilize an ESP32 microcontroller to interface with the DHT11 sensor and control the DC fan. The components will be interconnected on a breadboard, providing a flexible and modular setup for prototyping and testing.

### **iii) Software Development**

Programming of the ESP32 microcontroller will be performed using the Arduino IDE. Blynk, a mobile application platform, will be integrated to develop a user-friendly interface for remotely monitoring temperature readings and adjusting fan speed settings. The software will implement algorithms to interpret sensor data, determine fan speed adjustments, and communicate with the Blynk app.

### **iv) Testing and Validation**

Comprehensive testing will be conducted to validate the functionality and performance of the Smart Fan System. This includes unit testing of individual hardware and software components, integration testing to ensure seamless communication and operation, functional testing to verify system behavior under various temperature conditions, and performance testing to assess reliability and responsiveness in real-world scenarios.

## **2.6. PROJECT RISKS**

### **i) Technical Complexity**

The intricate nature of interfacing hardware components and developing firmware may introduce technical challenges, potentially leading to delays in project milestones.

### **ii) Supply Chain Disruptions**

Interruptions or delays in the availability of essential hardware components, such as the ESP32 microcontroller or DHT11 sensor, could impact project timelines and delivery schedules.

### **iii) Compatibility Issues**

Compatibility issues between firmware code, development tools like the Arduino IDE, and the Blynk platform might arise, causing software glitches or unexpected system behaviour.

#### **iv) Data Security Vulnerabilities**

Inadequate security measures in data transmission or storage mechanisms may expose sensitive user data to potential breaches, compromising user privacy and system integrity.

#### **v) User Adoption Challenges**

User dissatisfaction stemming from issues with system performance, usability, or reliability could hinder user acceptance and adoption of the Smart Fan System, affecting overall project success.

#### **vi) Regulatory Compliance Risks**

Failure to adhere to regulatory standards and industry guidelines pertaining to IoT devices, data privacy, or electrical safety may lead to legal repercussions, including fines or penalties.

### **2.7. PROJECT MITIGATION**

#### **i) Technical Complexity**

Conduct thorough research and feasibility studies before commencing the project to identify potential technical challenges. Allocate sufficient time for prototyping, testing, and troubleshooting. Engage with experts or consultants to provide guidance and support in overcoming technical hurdles. Additionally, develop contingency plans and establish flexible project schedules to accommodate unforeseen technical challenges or delays.

#### **ii) Supply Chain Disruptions**

Diversify the supply chain by sourcing components from multiple vendors to reduce dependency on a single supplier. Maintain open communication with suppliers to stay informed about potential disruptions and proactively address any issues. Consider stocking up on critical components to mitigate the impact of supply chain delays.

### **iii) Compatibility Issues**

Regularly update firmware code, development tools, and software libraries to ensure compatibility and stability. Conduct thorough testing and validation of software components across different versions to identify and address compatibility issues early in the development process. Implement version control and rollback procedures to mitigate the impact of compatibility issues on project timelines.

### **iv) Data Security Vulnerabilities**

Implement robust encryption protocols and secure communication channels to protect sensitive data transmitted between the Smart Fan System and external devices or platforms. Adhere to industry best practices for data handling, storage, and access control. Conduct regular security audits and penetration testing to identify and address potential vulnerabilities in the system.

### **v) User Adoption Challenges**

Involve end users in the design and testing phases of the project to gather feedback and address user preferences. Prioritize usability and user experience in the design of the Smart Fan System interface. Provide comprehensive training and support to users to ensure they are comfortable with the system and understand its benefits. Continuously monitor user adoption metrics, such as usage patterns, feedback, and support requests, to identify areas for improvement. Use this data to iterate on the project, making refinements and enhancements that address user needs and preferences.

### **vi) Regulatory Compliance Risks**

Stay informed about relevant regulatory standards and industry guidelines applicable to IoT devices, data privacy, and electrical safety. Incorporate compliance requirements into the design and development process from the outset. Seek legal counsel or regulatory experts to ensure adherence to applicable regulations and standards. Conduct regular compliance audits to identify and address any compliance gaps proactively.

## **2.8. TESTING PLAN**

### **i) Unit Testing**

Test individual software components, such as sensor data reading, fan speed control logic, and communication protocols, in isolation. Use simulated inputs to verify the behaviour of each component under different conditions. Ensure that each unit functions as expected and meets specified requirements.

### **ii) Integration Testing**

Test the interaction and integration of hardware and software components. Verify that sensor data is accurately read by the ESP32 firmware and transmitted to the Blynk app. Ensure proper communication between the ESP32 microcontroller and the Blynk server.

### **iii) Functional Testing**

Test the overall functionality of the Smart Fan System in a real-world environment. Verify that temperature readings are accurately displayed on the Blynk app in real-time. Test the responsiveness and reliability of fan speed adjustments based on temperature changes.

### **iv) Performance Testing**

Test the system's performance under various load conditions, such as high temperatures or rapid temperature changes. Measure response times for sensor data acquisition, fan speed adjustments, and remote control operations. Assess system stability, reliability, and resource utilization under stress conditions.

### **v) User Acceptance Testing (UAT)**

Involve end users in the testing process to evaluate the usability, intuitiveness, and effectiveness of the Smart Fan System. Gather feedback on user experience, interface design, and overall satisfaction with the system. Incorporate user feedback to make iterative improvements and enhancements to the system.

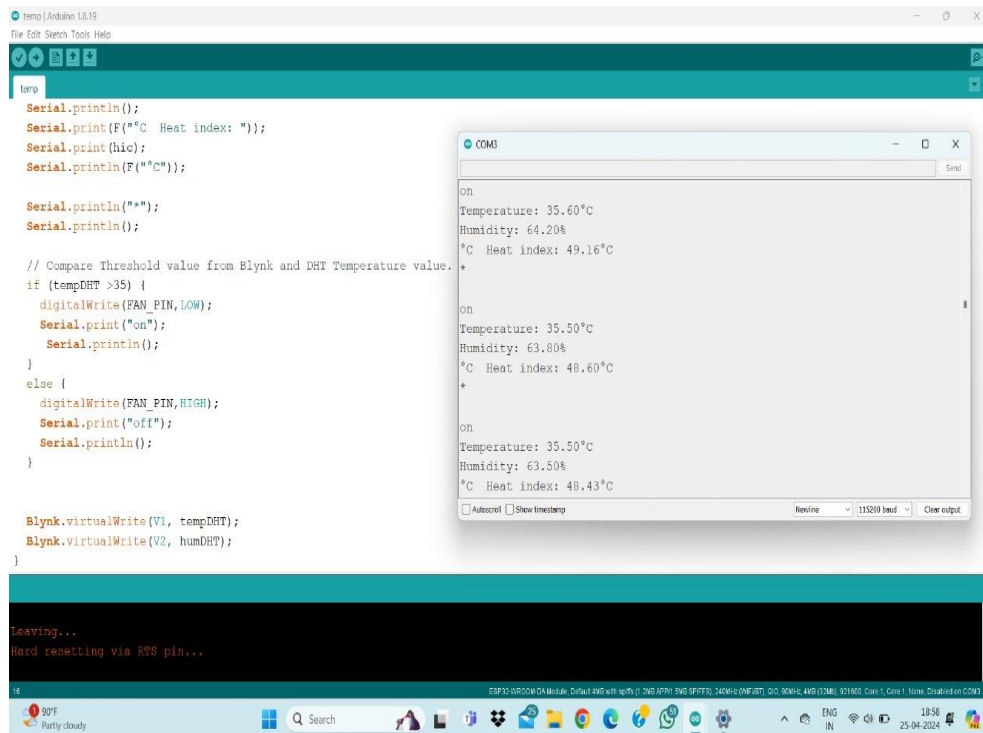
## CHAPTER 3

### RESULTS AND DISCUSSIONS



**Fig 3.1 Smart Fan System Model**

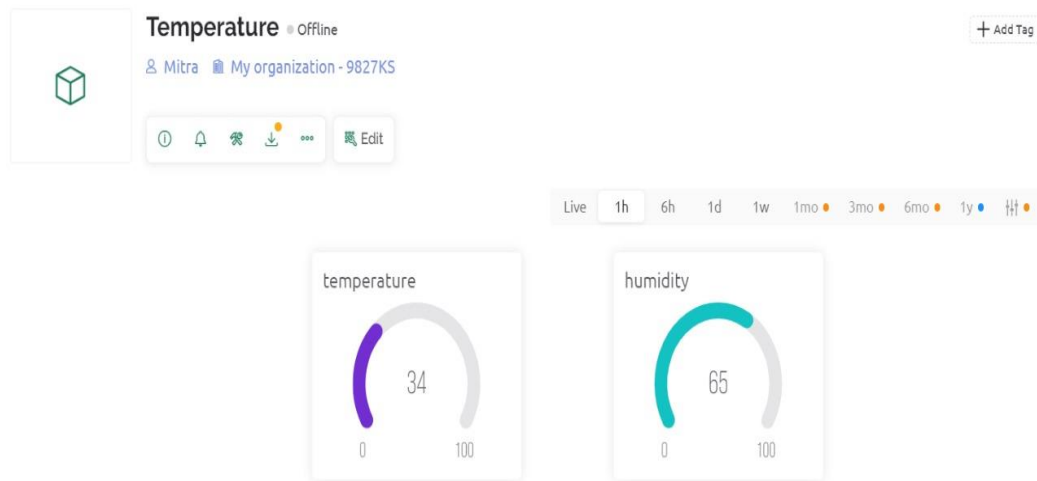
The Smart Fan System operates by automatically adjusting the speed of a DC fan based on real-time temperature readings from a DHT11 sensor. When the temperature exceeds a predefined threshold, the system activates the fan gets on environment. Conversely, if the temperature drops below the threshold, the fan gets off while ensuring optimal conditions. This functionality is achieved through the ESP32 microcontroller, which interprets sensor data and controls the fan's operation accordingly. By continuously monitoring temperature levels and dynamically adjusting fan speed, the system optimizes comfort and energy efficiency without requiring manual intervention. This transparency empowers users to optimize energy usage and reduce their environmental footprint while maintaining comfort levels.



**Fig 3.2 The Temperature and Humidity are Being Displayed on the Serial Monitor of Arduino IDE**

In the Arduino IDE, the process of executing code begins with the written sketch being compiled into machine language, which the microcontroller can understand and execute. This compilation process involves several steps, starting with preprocessing, where any preprocessor directives (such as `#include` or `#define`) are expanded. Next, the code is compiled, where each line of code is translated into machine language instructions specific to the microcontroller architecture being used. Finally, the compiled code is linked, where all the compiled modules are combined into a single binary file containing the executable code.

Throughout the execution of the code, the microcontroller may send output messages to the Serial Monitor via the USB connection. These output messages can include debug information, sensor readings, or any other relevant data that the developer has programmed the microcontroller to output. The Serial Monitor serves as a tool for monitoring these messages in real-time, allowing developers to observe the behaviour of the code and diagnose any issues that may arise.



**Fig 3.3 The Temperature and Humidity are Being Displayed on the Blynk App**

In the Smart Fan System, the Blynk app serves as a central hub for remote monitoring and control, offering users a seamless interface to interact with the system from anywhere with an internet connection. Through the app, users can access real-time temperature readings and humidity levels captured by the DHT11 sensor, providing valuable insights into their indoor environment. Additionally, the app allows users to adjust fan settings, such as fan speed or operating mode, with just a few taps on their smartphone or tablet.

This level of convenience empowers users to customize their comfort preferences on the go, whether they're at home, at work, or away on vacation. Furthermore, the Blynk app can send notifications or alerts to users based on predefined triggers, such as when the temperature exceeds a certain threshold or when the fan is turned on remotely. This proactive communication ensures that users stay informed about changes in their environment and enables them to take timely action as needed. Overall, the Blynk app enhances the functionality and accessibility of the Smart Fan System, providing users with greater control and peace of mind over their indoor climate management.

## **CHAPTER 4**

### **CONCLUSION AND FUTURE WORK**

#### **4.1. CONCLUSION**

The Smart Fan System represents a groundbreaking innovation in the realm of climate control, harnessing the power of IoT technology to revolutionize how we manage environmental conditions. At its core, the system seamlessly integrates hardware components such as the ESP32 microcontroller and DHT11 sensor, leveraging their capabilities to monitor real-time temperature fluctuations with precision. This data serves as the foundation for intelligent decision-making, allowing the system to dynamically adjust the speed of the DC fan in response to changing conditions. By doing so, the Smart Fan System not only enhances comfort but also promotes energy efficiency, effectively aligning with modern sustainability goals.

Through meticulous programming using platforms like the Arduino IDE, the system's firmware orchestrates the intricate dance between hardware components, ensuring seamless communication and precise control. Additionally, the integration of the Blynk app provides users with a user-friendly interface for remote monitoring and adjustment of fan settings. This convergence of hardware and software engineering represents the pinnacle of technological innovation, empowering users with unprecedented control over their indoor environment.

However, the journey towards realizing the Smart Fan System's full potential is not without its challenges. From navigating technical complexities to mitigating supply chain disruptions, the path to implementation requires careful planning and execution. Yet, with proactive risk management strategies and rigorous testing protocols in place, these obstacles can be overcome.

## **4.2. FUTURE WORK**

### **i) Advanced Control Algorithms**

Implementing more sophisticated control algorithms could further optimize the system's performance. Machine learning techniques, for example, could enable the system to adaptively adjust fan speed based on historical data and user preferences, leading to even greater energy savings and comfort.

### **ii) Integration with Smart Home Ecosystems**

Integrating the Smart Fan System with existing smart home ecosystems and platforms could unlock new possibilities for automation and interoperability. Compatibility with popular platforms like Amazon Alexa or Google Home would enable voice control and seamless integration with other smart devices in the home.

### **iii) Enhanced Environmental Monitoring**

Expanding the sensor capabilities to include additional environmental parameters beyond temperature and humidity, such as air quality or occupancy sensing, could provide users with a more comprehensive understanding of their indoor environment. This data could be leveraged to further optimize fan operation and promote overall well-being.

### **iv) Energy Harvesting and Sustainability**

Exploring energy harvesting technologies, such as solar or kinetic energy harvesting, could reduce the system's reliance on external power sources and enhance its sustainability. Additionally, integrating renewable energy sources into the system's power supply could further reduce its environmental footprint.

### **v) User Experience Improvements**

Continuously refining the user interface and experience of the Smart Fan System through iterative design and user feedback could enhance usability and satisfaction.

## REFERENCES

### i) IEEE PAPERS

[1] Alheraish(2004) "Design and implementation of home automation system," in IEEE Transactions on Consumer Electronics, vol. 50(4), pp. 1087-1092.

[2] A. Gupta (2023) M. Gupta and K. Markam, "Temperature Control Smart Fan: Model Analysis and Working Analysis," IEEE Conference on Innovations in High Speed Communication and Signal Processing (IHCSPP), BHOPAL, India, pp. 527-530.

[3] H. Hao, Y. Lin, A. S. Kowli, P. Barooah and S. Meyn (2014) "Ancillary Service to the Grid Through Control of Fans in Commercial Building HVAC Systems," in IEEE Transactions on Smart Grid, vol. 5(4), pp. 2066-2074,

[4] S. Yammen, S. Tang and M. K. R. Vennapusa (2019 )"IoT based speed control of Smart Fan," IEEE Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT-NCON), Nan, Thailand, pp. 17-20.

[5] V. Barral Vales, O. C. Fernández, T. Domínguez-Bolaño, C. J. Escudero and J. A. García-Naya (2022) "Fine Time Measurement for the Internet of Things: A Practical Approach Using ESP32," in IEEE Internet of Things Journal, vol. 9(19), pp. 18305-18318.

[6] W. Yang, S. Qiao, Q. Song, Z. Liu and J. Yang (2015)"The design and implementation of wireless temperature and humidity control system based on nRF905," IEEE 10th Conference on Industrial Electronics and Applications (ICIEA), Auckland, New Zealand, pp. 753-756.

## **ii) BOOKS**

- [1] A. Kurniawan, "ESP32 for IoT Applications." Packt Publishing, 2018.
- [2] L. Wang and J. Xi, Smart Devices and Machines for Advanced Manufacturing. London: Springer, 2008.
- [3] S. F. Barrett, "Embedded Systems Design with the Atmel AVR Microcontroller." Morgan & Claypool Publishers, 2010.
- [4] S. M. Sivakumar, V. Buravalla, and A. R. Srinivasa, Smart Devices: Modeling of Material Systems: An International Workshop, IIT Madras, Chennai, India, 10-12 January 2008. New York: American Institute of Physics, 2008.

## **iii) INTERNET ARTICLES**

- [1] S. Gupta, "IOT based smart fan control using ESP8266 and Blynk," Circuit Digest - Electronics Engineering News, Latest Products, Articles and Projects, <https://circuitdigest.com/microcontroller-projects/iot-based-smart-fan-control-using-esp8266-and-blynk>
- [2] A. Parajuli, "IOT temperature & Humidity Monitoring & control system using ESP32 & Blynk 2.0," IoT Projects Ideas, <https://iotprojectsideas.com/iot-temperature-humidity-monitoring-control-system-using-esp32-blynk-2-0>
- [3] A. Newton, "IOT smart exhaust fan: ESP32 based monitoring & control," How To Electronics of Smart Exhaust Fan, <https://how2electronics.com/iot-smart-exhaust-fan-esp32-based-monitoring-control/>
- [4] "IOT project using ESP32 blynk with fan speed control and Sensor," Fan Speed Control with ESP32, <https://www.instructables.com/IoT-Project-Using-ESP32-Blynk-With-Fan-Speed-Contr/>

## APPENDICES

### A. SAMPLE CODE

```
#define BLYNK_TEMPLATE_ID "TMPL3EkRbywVv"
#define BLYNK_TEMPLATE_NAME "temperature controlled fan"
#define BLYNK_AUTH_TOKEN "jhVEtp1PH30_hfBPqmdI7NBYbBHqrFHj"
#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <DHT.h>

#define DHTPIN 5      // Digital pin connected to the DHT sensor
#define DHTTYPE DHT11 // DHT 11

#define FAN_PIN 2      // FAN RELAY

char ssid[] = "speaker";
char pass[] = "kanna123";

float humDHT = 0;
float tempDHT = 0;

// Initialize DHT sensor.
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(115200);
```

---

```

    pinMode(FAN_PIN, OUTPUT);
    digitalWrite(FAN_PIN, LOW);
    Serial.println(F("DHTxx test!"));
    dht.begin();
    Blynk.begin(BLYNK_AUTH_TOKEN, ssid, pass);
}

void loop() {
    Blynk.run();

    // Wait a few seconds between measurements.
    delay(2000);

    // Reading temperature or humidity takes about 250 milliseconds!
    // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
    humDHT = dht.readHumidity();
    // Read temperature as Celsius (the default)
    tempDHT = dht.readTemperature();

    // Check if any reads failed and exit early (to try again).
    if (isnan(humDHT) || isnan(tempDHT)) {
        Serial.println("Failed to read from DHT sensor!");
        return;
    }

    Serial.print(F("Temperature: "));
    Serial.print(tempDHT);
    Serial.print(F("°C "));
    Serial.println();
    Serial.print(F("Humidity: "));
    Serial.print(humDHT);
    Serial.print(F("%"));
    Serial.println();

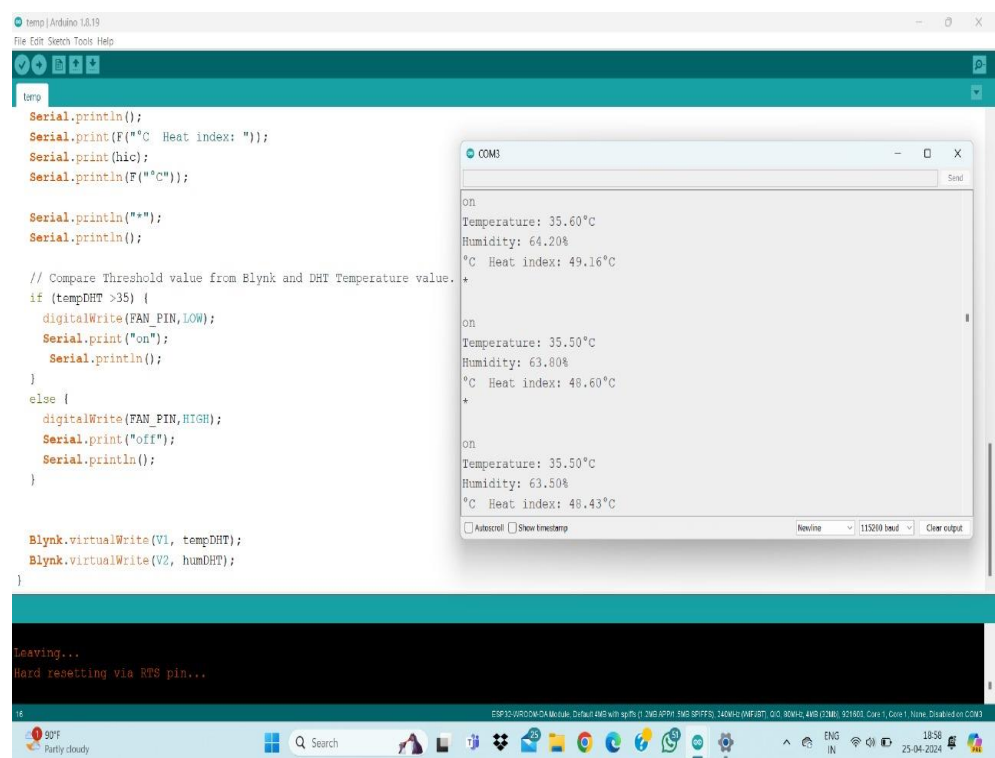
    Serial.println("*");
    Serial.println();

    // Compare Threshold value from Blynk and DHT Temperature value.
    if (tempDHT > 31) {
        digitalWrite(FAN_PIN, HIGH);
    }
    else {
        digitalWrite(FAN_PIN, LOW);
    }

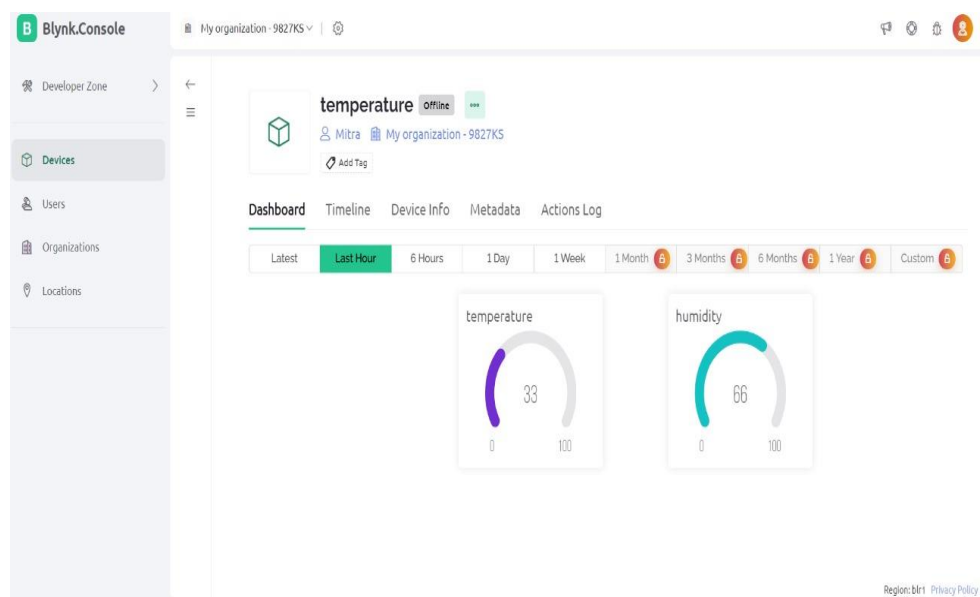
    Blynk.virtualWrite(V1, tempDHT);
    Blynk.virtualWrite(V2, humDHT);
}

```

## B. SCREENSHOTS



**B.1 The Temperature and Humidity are Being Displayed on the Serial Monitor of Arduino IDE**



**B.2 Blynk App Working**