GROUP ID: GE05, B08
A PROJECT REPORT ON

"CLOUD INTEGRATED AIR PURIFICATION SYSTEM"

SUBMITTED TO
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING AN AUTONOMOUS INSTITUTE, PUNE
IN THE FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

BACHELOR OF TECHNOLOGY
COMPUTER ENGINEERING (REGIONAL LANGUAGE)

SUBMITTED BY

MRUNAL GAIKWAD	**121B1D025**
JAYATI WAJIRE	**121B1D037**
ANWAY ALAMWAR	**122B2E207**
GAYATRI GORE	**122B2E210**
SNEHA LONI	**122B2E215**

DEPARTMENT OF COMPUTER ENGINEERING (REGIONAL LANGUAGE)

And

ELECTRONICS & TELECOMMUNICATION

PCET'S PIMPRI CHINCHWAD COLLEGE OF ENGINEERING

Sector No. 26, Pradhikaran, Nigdi, Pimpri-Chinchwad, PUNE 411044

CERTIFICATE

This is to certify that the project report entitles

"CLOUD INTEGRATED AIR PURIFICATION SYSTEM"

Submitted by

MRUNAL GAIKWAD	**121B1D025**
JAYATI WAJIRE	**121B1D037**
ANWAY ALAMWAR	**122B2E207**
GAYATRI GORE	**122B2E210**
SNEHA LONI	**122B2E215**

is a bonafide student of this institute and the work has been carried out by him/her under the supervision of **Prof. Sujata Kolhe** and **Dr. P. V. Sontakke** and it is approved for the partial fulfilment of the requirement of Pimpri Chinchwad College of Engineering an autonomous institute, for the award of the B. Tech. degree in Computer Engineering (Regional Language).

Dr. Sujata Kolhe	**Dr. P. V. Sontakke**	**Dr. Rachana Y. Patil**	**Dr. K. S. Kinage**
Project Guide	**Project Guide**	**HOD, CE(RL)**	**HOD, E&TC**

Dr. G. N. Kulkarni
Director,
Pimpri Chinchwad College of Engineering Pune –411044

Place: Pune
Date: /04/2025

CERTIFICATE

"CLOUD INTEGRATED AIR PURIFICATION SYSTEM"

Sponsored By

Arklite Speciality Lamps Private Limited

Submitted for partial fulfillment of the Bachelor of Technology degree requirements in the Department of Electronics & Telecommunication Engineering and Computer Engineering (Regional Language), Pimpri Chinchwad College of Engineering, Savitribai Phule University of Pune, Pune.

By

MRUNAL GAIKWAD	**121B1D025**
JAYATI WAJIRE	**121B1D037**
ANWAY ALAMWAR	**122B2E207**
GAYATRI GORE	**122B2E210**
SNEHA LONI	**122B2E215**

Mr. Mandar Sahasrabudhe	**Dr. Sujata Kolhe**	**Prof. Rohini Y. Sarode**	**Dr. Rachana Y. Patil**	**Dr. K. S. Kinage**
External Project Guide	**Project Guide**	**Project Coordinator**	**HOD, CE(RL)**	**HOD, E&TC**

Dr. G. N. Kulkarni
Director,
Pimpri Chinchwad College of Engineering Pune – 411044

PIMPRI CHINCHWAD EDUCATION TRUST's

PIMPRI CHINCHWAD COLLEGE OF ENGINEERING

(An Autonomous Institute)

TÜV INDIA ISO 9001 Certified

- N.B.A. Accredited Courses - Civil Engg, Comp. Engg., E&TC Engg., IT. Engg, Mech. Engg.
- NAAC Accredited with 'A' Grade
- An ISO 9001:2015 Certified Institute
- Approved by A.I.C.T.E., New Delhi, Govt. of Maharashtra, D.T.E., Mumbai
- Permanently Affiliated to Savitribai Phule Pune University, Pune

Dnyaneshwar P. Landge	Padmatai Bhosale	Vitthalrao S. Kalbhor	Shantaram D. Garade	Dr. Govind N. Kulkarni
Chairman	Vice Chairperson	Secretary	Treasurer	Director

Ref. No.: PCCOE/AS&H/24-25/40 Date : 23/7/24

MEMORANDUM OF UNDERSTANDING

Between

PIMPRI CHINCHWAD EDUCATION TRUST'S
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING
DEPARTMENT OF APPLIED SCIENCES AND HUMANITIES
Sector No.26, Pradhikaran, Nigdi,
Pune, Maharashtra 411 044

And

ARKLITE SPECIALITY LAMPS Pvt. Ltd
Gat No 2794, Kharabwadi Chakan Talegaon Road,
Tal Khed, Chakan, Pune, Maharashtra 410501.

ARKLITE® SPECIALITY LAMPS
AN ISO 9001:2015 CERTIFIED COMPANY

ARKLITE SPECIALITY LAMPS PVT. LTD. CHAKAN

PCET's Pimpri Chinchwad College of Engineering An Autonomous Institute Nigdi, Pune-44

ACKNOWLEDGEMENT

All the achievements of the world depend on the toil of countless individuals, and this project too is not an exception. Irrespective of the source it is derived from, we wish to thank everyone who has made this project a success.

We thank **Dr. P.V. Sontakke** and **Prof. Sujata Kolhe** for ongoing guidance and supervision. Their expertise and experience have significantly impacted the course of this project. We also thank **Prof. Swapnil S. Ayane**, **Prof. Rohini Y. Sarode** and the project coordination team for offering valuable information and support during the course of this work.

We are blessed to have an H.O.D such as **Dr. K.S. Kinage [E&TC]**, and **Dr. Rachana Y. Patil [CS Regional]** whose review, comments, corrections, and suggestions were highly beneficial in enhancing our project. We also wish to thank **Dr. G. N. Kulkarni** for their motivating words of encouragement at each step of the project. We would also like to thank **Dr. Sonali Kale** for their regular supervision and encouragement.

We are grateful to **Mr. Mandar Sahastrabudhe** of Arklite Speciality Lamps Private Limited for their kind guidance and suggestions. Their technical know-how and feedback have greatly helped in the growth and development of our project.

I am also happy to convey my thankfulness and thanks to my parents and friends for their continuous inspiration & encouragement. In the end, we thank lab assistance, department of electronics & telecommunication, and computer engineering (regional language) for their continuous participation at every step in the project which has guided this project on the road of success.

Anway Alamwar
Gayatri Gore
Sneha Loni
Mrunal Gaikwad
Jayati Wajire

ABSTRACT

Our project shows how to implement a Cloud Integrated Air Purification System on the ESP32 microcontroller for real-time monitoring and control of UV lamps to treat air.As part of this project, a mobile application has also been created in such a way that the users can remotely control the UV lamp and track the status in real time. The application provides commands to the ESP32 over a local network or internet and provides control features like ON/OFF switch, status update, and operation log. ESP32 has an integrated web server and communicates with the Thing Speak IoT cloud server to log and display important parameters like lamp ON/OFF status, cumulative running time, and fault statuses (e.g., inability to startup or extended OFF time).An alarm feature is also included, giving visual alerts on dashboard and app when UV lamp is in OFF state for a prolonged period of time beyond a configurable duration of time.On the software front, a mobile application using Android created in Flutter has been used to make it easy for real-time observation and control of the UV lamps.The app interacts with ThingSpeak through API keys and graphically represents the lamp state with toggleable lamp states for users. Firebase Authentication has been employed to handle secure user access, and sign-up, sign-in, and logout functionality is supported. Historical access is also offered for performance monitoring.Some of the significant challenges that are being addressed include wireless reliability, synchronization of data, and the preservation of an interruption-free interface between hardware and software. The exercise leads to cloud-integrated scalable smart UV control with improved maintenance, reliability, and convenience for the users employing IoT and mobile technology.Through the combination of cloud services, mobile control, and feedback in real time, the system offers a scalable, low-cost, user-friendly solution for smart air purification management.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Background

In recent years, air purification has gained special significance due to the increased interest in indoor air quality and its direct relation with human well-being. UV-C radiation, being germicidal, has found extensive application in sterilization technology due to its effectiveness to eradicate airborne germs, viruses, and bacteria. Conventional UV-C air treatment systems do not have smart control and monitoring features, leading to wasteful energy utilization and little feedback about the operation status.

For surmounting the limitations, in this project an ESP32-Based Cloud-Integrated UV Lamp Monitoring and Control System that integrates intelligent control, remote monitoring, and fault alarm has been proposed. By taking advantage of the features of the ESP32 microcontroller, the system combines a relay-controlled UV-C lamp with Wi-Fi capabilities, allowing real-time data to be transmitted to the cloud and user interfaces. The system is remotely controlled by both a web interface and a phone app, so the user can control the lamp from anywhere they have internet.

The project incorporates the ThingSpeak IoT platform to log lamp operation time and status, making data available for visualization and analysis. Furthermore, an in-built safety mechanism detects anomalies—such as failure to switch ON—and alerts users through the dashboard. This integration of embedded systems with cloud computing and mobile interaction provides a smart, scalable solution for air treatment applications, especially in environments like hospitals, laboratories, and homes where continuous air sterilization is essential.

1.2 Problem Statement

Traditional UV lamp monitoring systems in air cleaning systems use local control panels and LCD displays and need to be physically inspected to monitor lamp usage, operational condition, and system faults. Such a method is inefficient, not real-time remote accessible, and prone to delayed maintenance or undetected failures. With the growing industrial requirements of predictive maintenance and automation, the demand for a smart, IoT-powered solution that is equipped with real-time monitoring, remote operation, and error detection automation cannot be overstressed.

1.2.1 Aim

The goal of this project is to develop and integrate a Cloud-Integrated UV Lamp Monitoring and Control System using the ESP32 microcontroller with real-time monitoring, remote control, and fault detection of a UV-C lamp using the mobile/web interface. The system is aimed at enhancing operation efficiency, ensuring user safety, and allowing proactive maintenance through the application of IoT technology and cloud-based data visualization services like ThingSpeak.

1.2.2 Objectives

- To design and implement an IoT-enabled UV lamp monitoring and control system using the ESP32 microcontroller.
- To enable real-time data logging and visualization by integrating the system with the ThingSpeak cloud platform.
- To design and develop an Android application using Flutter for monitoring and controlling UV lamps used in these systems.
- To ensure secure and reliable wireless communication for uninterrupted remote access and system control.
- To display the operational status, on/off duration, and error alerts related to the UV lamp through a user-friendly interface.
- To encourage interdisciplinary collaboration by integrating hardware and software components through teamwork between Electronics and Computer Science students.

1.3 Motivation

The increasing demand for airborne contaminants and the transmission of infectious agents has necessitated the use of air purification as an integral part of different environments such as hospitals, laboratories, and public areas. Although UV-based purification systems have become common due to their efficacy in sterilizing air, conventional systems depend predominantly on human supervision without the capabilities of real-time monitoring and control. This not only decreases system

performance but also increases the risk of unidentified system malfunctions. The motive behind this project is the aspiration to automate observation, improve reliability, and reduce human dependence. With the implementation of the ESP32 microcontroller and ThingSpeak cloud service, this system aims to provide users with a smart, web-based interface for real-time status, data logging, and remote control of the UV lamp, thus making the process of air treatment smarter and more dependable.

CHAPTER 2

LITERATURE SURVEY

2.1 Literature Survey

To understand available technologies and approaches with regard to control of UV lamps, IoT application, and cloud-based monitoring, a survey of literature was carried out. The summary of the essential papers reviewed is presented in the table below, including their prime contributions and usefulness to the project at hand.

Sr. No.	Title of the Paper	Year of Publication	Publisher	Methodology	Conclusion
1	Design of an ESP32-Based IoT Smart Home Automation Management System	2024	Global Journal of Research in Engineering & Computer Sciences	The system runs on D.C. batteries, allowing 2–3 days of operation when fully charged. An LCD displays key information like project title, temperature, and house occupancy. Entry is restricted based on a set maximum number of people. A fan activates automatically when the temperature exceeds a user-defined limit, and alerts are sent via SMS and Wi-Fi to a connected smartphone.	This smart home system effectively manages occupancy and temperature using user-defined settings. With battery power and dual notification methods, it offers both flexibility and reliability in home automation
2	Home Automation Using Wi-	2024	IJARSCT	The system uses an ESP32 microcontroller with Wi-Fi to control	The project successfully delivers a reliable and responsive home

	Fi: ESP32-Based System for Remote Control and Environmental Monitoring			home appliances remotely. Relay modules manage the switching of devices, while DHT11 sensors monitor temperature and humidity. A user-friendly smartphone app allows real-time control and monitoring. All components work together to enable smooth communication and automation.	automation system. It improves everyday living by offering comfort, convenience, and better energy use through easy remote access and control.
3	Unwanted Indoor Air Quality Effects from Using Ultraviolet C Lamps for Disinfection	2023	Environmental Science & Technology Letters (American Chemical Society)	A hospital-grade UVC device was used in laboratory tests to measure air composition before, during, and after UVC irradiation, comparing particle concentration and chemical composition.	UVC disinfection generates fine particles and gas-phase chemical changes, potentially causing health risks if rooms are reoccupied too soon, highlighting the need for further research.
4	Design of an IoT Air Disinfection Machine Control System	2023	IOP Publishing (Journal of Physics: Conference Series, Volume	The system uses sensor modules to detect air quality parameters, IoT for remote monitoring, communication modules for energy efficiency, and UV	The system, combining real-time monitoring and IoT-based control, effectively improves indoor air quality by reducing harmful microorganisms and

			2562)	light, ozone generation, and negative ion generators for air purification.	pollutants, and offers user-friendly control via a mobile app.
5	Development of an IoT-Enabled Air Pollution Monitoring and Air Purifier System	2023	Metrology Society of India (Published in the MAPAN-Journal of Metrology Society of India)	Raspberry Pi 3 B+, PMSA003 particulate matter sensor, MQ2 and MQ135 gas sensors, filters, and IoT integration for real-time monitoring and analysis.	The system effectively reduced particulate matter and gaseous pollutants, meeting ISO 9 cleanroom standards, and achieved high filtration efficiency, with real-time IoT monitoring for improved air quality management.
6	Evaluation of an Air Cleaning Device Equipped with Filtration and UV: Comparison of Removal Efficiency on Particulate Matter and Viable Airborne Bacteria in	2022	International Journal of Environmental Research and Public Health (MDPI)	The study evaluated the performance of an existing particulate matter (PM) air filtration device with UV-C lamps in poultry room environments, assessing removal efficiency and data analysis.	The upgraded prototype effectively improved indoor air quality by removing bacteria, suspended particulates, and finer PM categories, demonstrating the importance of hybrid air purification systems in reducing indoor disease transmission.

	the Inlet and Treated Air				
7	An IoT-based handheld environmental and air quality monitoring station	2023	ACTA IMEKO	The system uses a NodeMCU with ESP8266 to collect data from five sensors and sends it to ThingSpeak every 30 seconds. A mobile app, built with Android Studio, fetches this data using REST APIs in JSON format. A handheld display, connected via Wi-Fi, updates readings every minute. MySQL is used for storing the data.	The developed IoT-based monitoring system provides real-time tracking of environmental and air quality parameters. It offers flexibility in data access through both a mobile app and a portable display, making it suitable for integration into a larger weather monitoring network.
8	Ultraviolet Air and Surface Treatment	2019	ASHRAE Handbook —HVAC Applications	UVGI, a method of destroying microbial DNA and proteins, is used in various applications, including healthcare, commercial, and residential buildings for airborne pathogen control and surface decontamination.	The HVAC system effectively reduces airborne pathogens, improves energy efficiency, and ensures safety through proper design and adherence to standards.

Table 2.1 Literature Survey

2.2 Summary of Literature Survey

- **Unwanted Indoor Air Quality Effects from Using Ultraviolet C Lamps for Disinfection**

  This study examines the unforeseen effects of exposure to high-intensity UVC devices on indoor air quality (IAQ). The research concluded that UVC irradiation causes elevated concentrations of fine particles and chemical alterations in indoor air, such as higher VOCs and oxidation products. These alterations could be harmful to health if spaces are rapidly reoccupied following disinfection. The results highlight the importance of in-depth studies of the environmental effects of UVC disinfection systems.

- **Design of an IoT Air Disinfection Machine Control System**

  This article proposes an IoT control system for an air disinfection unit. The system employs the ESP32 microcontroller for integration of air quality sensors, UV sterilization lamp, ozone generators, and HEPA filters for real-time feedback and disinfection. Users can remotely operate the device using a cloud-connected mobile app. Testing within a laboratory environment revealed remarkable improvements in air quality with automatic responses by the system to varying air quality indices. The design proved to be effective in removing pollutants and pathogen

- **Development of an IoT-Enabled Air Pollution Monitoring and Air Purifier System**

  This research presents an IoT-based air purification and monitoring prototype with Raspberry Pi 3 B+, PMSA003 sensors, and gas sensors (MQ2 and MQ135). The device employs a multi-stage filtration mechanism (pre-filter, activated carbon filter, and HEPA filter) to clean air, which focuses on particulate matter and gaseous contaminants. Sensor readings are read to Thingspeak Cloud platform every 15 seconds for real-time processing. Experimental testing under normal, combustion, and smoke conditions confirmed its capacity to greatly lower air pollutants. The system exceeded ISO

9 cleanroom standards and provided high-quality filtration efficiency with promise for use in healthcare, industry, and the environment.

- **Evaluation of an Air Cleaning Device Equipped with Filtration and UV**

  This research assesses a prototype air cleaner that integrates multi-layer filtration and ultraviolet-C (UV-C) lamps to clean indoor air. The air cleaner was effective in eliminating almost 100% of culturable airborne bacteria and as much as 97% of particulate matter. The integration of filtration and UV-C treatment showed an effective two-layer system for enhancing indoor air quality, with high potential in preventing diseases in indoor environments.

- **Ultraviolet Air and Surface Treatment**

  This article focuses on the contribution of ultraviolet germicidal irradiation (UVGI) towards airborne pathogen control and surface cleanliness in the healthcare and commercial environments. The article touches on different UVGI systems such as upper-air UV, in-duct UV, and surface disinfection systems. UVGI enhances air quality and saves energy and maintenance, but it is safe when accompanied by guidelines. The research emphasizes the need for UVGI as a means of infection control and the efficiency of HVAC systems.

- **An IoT-based handheld environmental and air quality monitoring station**

  This paper presents an IoT-based weather and air quality monitoring station which may be incorporated as a unit in a wide network of weather data acquisition. The prototype of the designed product is made of 5 sensors and Wi-Fi connectivity. Wi-Fi connectivity is established through ESP8266 on the Node MCU board. Parameters can be successfully updated through the thing speak platform with a cycle of 30 seconds. The data can be also viewed with a handheld display unit or the mobile application as well. The display unit is refreshed every 1 minute with Wi-Fi connectivity. The mobile app is developed on Android Studio, Thinkspeak (cloud), MySQL as database and JSON for retrieving REST API.

- **Home Automation Using Wi-Fi: ESP32-Based System for Remote Control and Environmental Monitoring**

  This project presents an integrated home automation system based on Wi-Fi connectivity and ESP32 microcontrollers to improve modern living with smooth automation and user-focused design. The background identifies the integration of IoT technology into home automation and focuses on Wi-Fi's role in facilitating remote monitoring and control of domestic appliances. With intensive testing and evaluation, the system has demonstrated its functionality, reliability, and responsiveness, meeting its goals of maximizing home comfort, convenience, and energy efficiency. The use of relay modules and DHT11 sensors has ensured smooth communication and data exchange, while the provided smartphone application offers a user-friendly interface for interaction.

- **Design of an ESP32-Based IoT Smart Home Automation Management System**

  In this project, the management system and smart home utilize D.C. batteries as their power supply. Once fully charged, it can function for 2 to 3 days before depleting. Once you turn on the system, the name of the project is shown on the LCD, as well as the highest temperature. It will also indicate to you that no one is present within the house. You can select the number of individuals you wish to enter the house. If you select the maximum to be 12, the door will not open again until one person presses the exit button from the house and leaves the house before another individual will be able to enter the house. There is a fan in the house to cool down the house temperature if it is warm, and you can decide on when you want the fan to start automatically by setting the maximum temperature. If you set the maximum temperature to 30 °C, the fan will not start until it reaches that temperature. Messages will also be delivered via SMS and a smart phone by using a Wi-Fi connection.

2.3 Gap Identified through Literature Survey

Although a vast amount of research has been done regarding the efficacy of UV-C based air disinfection systems and incorporation of IoT for environmental monitoring, a few essential gaps still have not been met. It has been demonstrated in studies that UV-C irradiation can produce unintended indoor air quality effects in the form of high concentrations of VOCs and fine particles, but very few systems incorporate real-time alerts or monitoring systems to neutralize these threats. Furthermore, while various IoT-based air purifiers that employ microcontrollers and sensors have been proposed, most depend on expensive and powerful processing platforms such as Raspberry Pi, thus complicating the system and elevating system cost.

Additionally, while remote control has been showcased in mobile applications, the approach still misses the utilization of light platforms such as ThingSpeak for minimalist real-time visualization and control of data. Few systems have also implemented direct relay control for actuating UV-C lamps along with monitoring precise ON/OFF time intervals, rendering it challenging to log operational activity and detect fault conditions effectively. This project fills these loopholes by suggesting a low-cost ESP32-based UV air treatment system with ThingSpeak integration for real-time cloud monitoring, mobile/web dashboard access, and automated alerting—promoting safer, smarter, and more accessible air purification.

CHAPTER 3
METHODOLOGY

3.1 Project Outline

The intended system will create an intelligent, IoT-based UV air cleaning system that supports remote monitoring, control, and fault alerting through a cloud dashboard. The whole process is organized into the following important stages:

The system is constructed around the ESP32 microcontroller, which is the master controller of the UV lamp operation. A relay is attached to ESP32 to turn the UV lamp ON/OFF depending on the user input or logic.

The envisioned system is intended to provide real-time monitoring, fault detection, remote control, and secure data management of UV lamps through a cloud-integrated method. A microcontroller from ESP32 is used to capture the real-time data of the UV lamps such as their current operation status, runtime, and fault. This information is sent over Wi-Fi to a cloud platform, ThingSpeak, to act as the system parameters' central database and visualization tool. The system can detect faults like lamp failure or aberrant behavior. It instantly alerts users via the mobile app through push notifications or messages in the app once such occurrences have been detected, allowing for prompt action to be taken. In addition to this, the mobile app provides total remote control facilities through which users can turn the lamps ON or OFF from anywhere, utilizing an easy-to-use interface.

In order to have a history of lamp operation and for traceability purposes, all the information such as lamp status, operational duration, and system warnings is safely stored in the ThingSpeak cloud server. Such a historical record can be utilized for maintenance review and performance monitoring. Further, for protection against unauthorized system access, the application includes a user authentication module. It enables users to log in safely through distinct credentials (email and password), and new users can register by going through the sign-up procedure, thus ensuring that only approved staff members will have the ability to engage with the system.

3.2 Block Diagram

The provided block diagram represents an IoT-enabled UV lamp control and monitoring system using an ESP32 microcontroller.

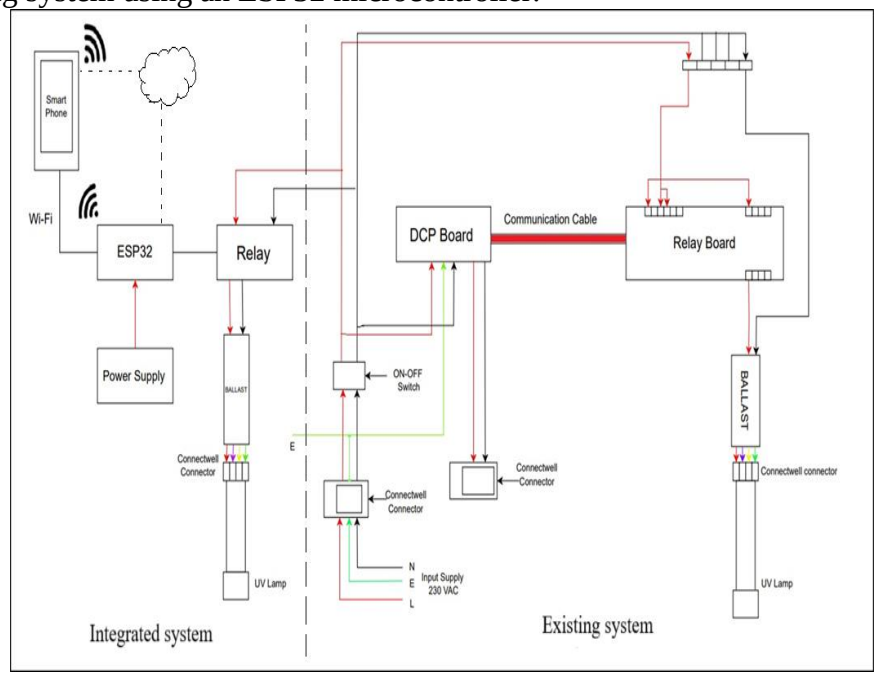


Figure 3.2.1 Block Diagram

The system allows remote control and real-time monitoring via a smartphone using Wi-Fi communication. The ESP32 is the central control unit, receiving commands from the smartphone and processing them to control the lamp. A relay module is connected to the ESP32, acting as an electronic switch that controls the power supply to the UV lamp. The power supply unit provides the necessary voltage to operate the ESP32 and the relay. The system also integrates a Digital Control Panel (DCP) Board, which communicates with a Relay Board through a communication cable to manage multiple lamps. The DCP Board is linked to an ON-OFF switch, allowing manual control alongside remote operation. Each UV lamp is connected to a ballast, which regulates the electrical current to ensure stable operation. The wiring system uses Connectwell connectors to ensure secure electrical connections between the components. The input power supply is 230V AC, which is distributed appropriately through the circuit. The Relay Board further manages multiple lamps, ensuring synchronized control. By integrating IoT-based control, this system enhances automation, accessibility, and monitoring, reducing manual intervention while improving system reliability and efficiency.

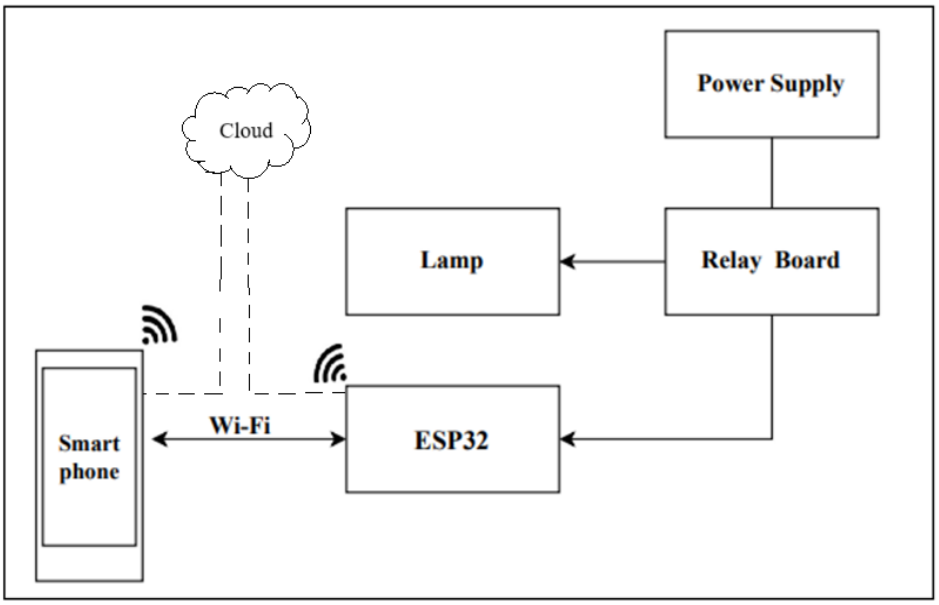


Figure 3.2.2 Purposed Block Diagram

In this setup, a smartphone acts as the user interface, allowing commands to be sent wirelessly via Wi-Fi. The ESP32, a versatile microcontroller with built-in Wi-Fi capabilities, receives these commands and processes them to control the relay board, which acts as an electronic switch. The relay board is responsible for switching the UV lamp ON and OFF based on the received instructions. A power supply unit provides the necessary electrical energy to both the relay board and the lamp, ensuring continuous operation. This system eliminates the need for manual intervention, enabling real-time remote monitoring and control of the UV lamp. By integrating IoT technology, the system enhances efficiency, accessibility, and automation, making it more reliable compared to conventional control methods.

3.3 Flow chart

This flowchart illustrates the process of collecting data from a device and displaying it on a smartphone using an ESP32 microcontroller and Wi-Fi.

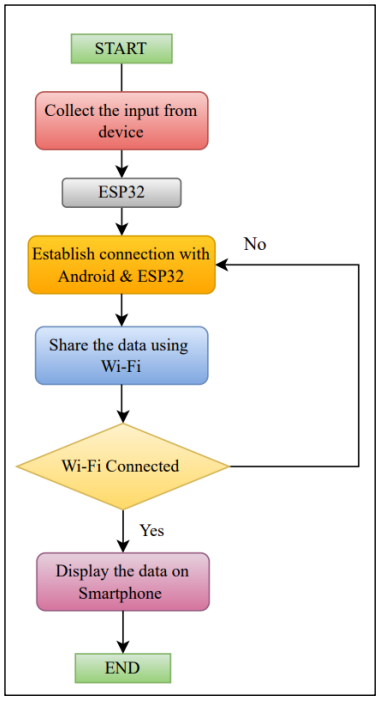


Figure 3.3. Flow chart

The process starts by gathering data from a device, which is then processed by the ESP32. The ESP32 attempts to share this processed data over Wi-Fi. If the Wi-Fi connection is successful, the data is displayed on a smartphone. If the connection fails, the ESP32 tries again, ensuring the data is transmitted once a connection is established. This system demonstrates a basic Internet of Things (IoT) application where data is acquired, processed, and wirelessly communicated for monitoring or control purposes on a smartphone.

- **Frontend Development: Flutter & Dart**

  Flutter: The mobile app was built using Flutter, Google's open-source UI toolkit. Flutter allows cross-platform development from a single codebase, enabling the app to run smoothly on both Android and iOS devices. It provides fast development, expressive UI components, and native performance.

  Dart Programming Language: The app logic, UI, and state management are written in Dart, the primary language used with Flutter. Dart is an object-oriented, client-

optimized language that supports reactive programming, making it ideal for building responsive and interactive mobile UIs.

Android Studio: was chosen as the primary IDE due to its strong support for Flutter and seamless integration with device emulators and debugging tools. The IDE facilitated UI design, code testing, and deployment on physical Android devices for real-time evaluation.

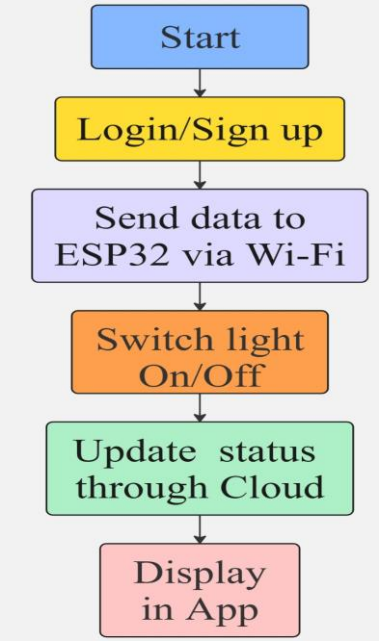


Figure 3.4. Flow chart for Android app

CHAPTER 4

DESIGN & IMPLEMENTATION OF PROPOSED SYSTEM

4.1 Hardware specification

The table below outlines the hardware components used in the project along with their specifications, which were selected based on the system's design requirements and functionality.

Component	Description	Key Specifications
ESP32 Development Board	Main controller with built-in Wi-Fi. Handles relay control, cloud communication, and local web server for user interface.	- Dual-core 240 MHz
		- 520 KB SRAM, 4MB Flash
		- Wi-Fi 802.11 b/g/n
		- Multiple GPIOs
		- Operating voltage: 3.3V
Single-Channel Relay Module	Used to switch the UV lamp ON/OFF based on ESP32 commands. Provides electrical isolation and protection.	- Trigger voltage: 3.3V–5V
		- 10A at 250V AC / 30V DC
		- Opto-isolated
		- LED status indicator
UV-C Lamp	Disinfects air by emitting UV-C light that inactivates airborne pathogens.	- Wavelength: ~254 nm
		- Remotely controlled via relay
		- Caution required due to health risks
Power Supply	Powers the ESP32 and relay module using USB adapter or power bank.	- Input: 100–240V AC
		- Output: 5V DC, 2A
		- Onboard voltage regulation

Table 4.1 Hardware Specification

4.2 Design Consideration

The design of the cloud integrated UV lamp monitoring and control system using ESP32 was guided by several key considerations to ensure that the system is efficient, reliable, safe, and user-friendly. The core objective was to develop a system capable of remotely controlling and monitoring a UV-C air disinfection lamp. To achieve this, the ESP32 microcontroller was selected due to its built-in Wi-Fi capabilities, low power consumption, and compatibility with cloud platforms like ThingSpeak. The system was designed to automatically collect operational data, such as the ON/OFF status and duration of lamp operation, and send it to the cloud for real-time monitoring and historical analysis. Connectivity was a critical aspect, and the ESP32 was programmed to attempt connection with multiple Wi-Fi networks for improved reliability. The user interface was another major design element. Instead of using a physical display like a 16x2 LCD or 0.96” OLED, a mobile and web-based dashboard was developed using the ESP32’s web server capabilities. This enabled platform-independent control and status viewing without additional hardware. Error detection was incorporated by setting a timer that triggers an alert if the lamp remains OFF beyond a threshold period, indicating a potential fault. Safety was a primary concern due to the use of UV-C light, which can be harmful if not properly managed. Therefore, the system was designed for remote control to minimize human exposure. The entire setup was developed to be cost-effective and scalable, allowing for future enhancements such as the integration of sensors for air quality monitoring or automation features. This thoughtful approach to design ensures the system remains robust in real-time environments, provides meaningful insights through cloud logging, and offers ease of use with its accessible interface.

CHAPTER 5

SOFTWARE IMPLEMENTATION

5.1 Software specification

5.1.1 System Features (Functional Requirements)

- Real-time Monitoring: The system should enable users to monitor the operating status of UV lamps in real-time using the mobile application.
- Fault Detection and Alerts: The system should be able to detect UV lamp faults and send immediate alerts (push notifications or in-app notifications) to users for prompt intervention.
- Remote Control: The app should enable users to switch the UV lamps on and off remotely, with complete control through the mobile interface.
- Data Storage: All operational information like lamp status, runtime, fault, and system alerts must be safely stored in a cloud database (ThingSpeak).
- Authentication: Secure login capability must be provided to enable user authentication, where users can log in with unique credentials (Email ID, Password), and sign-up option for new users as well.

5.1.2 External Interface Requirements

This section defines how the system interacts with external systems, devices, and software components.

User Interfaces

- Mobile Application Interface: The interface needs to be easy to use and responsive, providing users with seamless navigation and an ability to easily monitor UV lamp status.
- Alert Notifications: Alert notifications should pop up when there is a failure of a UV lamp.
- ESP32 Wi-Fi Module: The hardware (UV lamps with the ESP32 module integrated) will exchange data with the mobile app through Wi-Fi. The ESP32 will transfer the lamp's operating status (on/off, running time, faults) to the cloud for storage and retrieval by the mobile app.

- AC System Connectivity: The UV lamps should be able to interface with the hardware components of the AC system to get precise monitoring of operation data, e.g., lamp uptime

Software Interfaces

- Mobile App (Flutter): Flutter mobile app will communicate with Firebase for real-time data retrieval, monitoring, and user inputs.
- Firebase Authentication: Firebase Authentication is used for User authentication, allowing individuals to log in using unique credentials such as their email address and password for secure access.
- ThingSpeak : ThingSpeak will be used by the system for storing and retrieving data in real-time, providing speedy access to historical data and fault records.

Communication Interfaces

- Wi-Fi Communication: The system must use Wi-Fi for communication between the ESP32 module (which is connected to the UV lamps) and the mobile app. All data, including device status and fault notifications, will be sent through this interface.

5.1.3 Non-functional Requirements

This subsection describes the performance, security, and other non-functional requirements of the system.

Performance Requirements

- Response Time: The mobile app must refresh the UV lamp status within 3 seconds after getting new data from the ESP32 module.

Safety/Security Requirements

- **User Authentication Security:** Only authorized users can access the mobile application through Firebase Authentication. Passwords are stored securely in Firebase and never exposed. Login sessions are protected and time-bound (auto-logout if needed).

- **Secure Communication:** Communication between the app and ThingSpeak should use HTTPS to ensure data integrity and confidentiality. API keys used for accessing ThingSpeak must be stored securely and not exposed in the frontend.
- **Electrical Safety (Hardware):** Proper insulation and enclosures must be provided for the ESP32 board and UV lamps. Users should not have direct access to the hardware powering the UV lamps to prevent electric shock.
- **UV Radiation Safety:** UV lamps should not operate when the enclosure is open or if humans are in close proximity. Include physical interlocks or warnings if the lamp turns ON unexpectedly. Maintain safe distances and shielding around UV light sources.

5.2 Implementation

5.2.1 Database Requirements

The system uses Firebase Firestore to manage and store user data, such as login credentials, UV lamp status, usage hours, and error reports. Firebase provides real-time synchronization, allowing data to be accessed and updated instantly across platforms. It also supports offline capabilities, ensuring functionality even when network connectivity is lost. Additionally, ThingSpeak is utilized for storing IoT-based real-time operational data sent from the ESP32 microcontroller. ThingSpeak allows for scalable data handling and comes with built-in tools for data visualization, enhancing monitoring efficiency. Security is ensured through Firebase security rules, allowing data access only to authenticated users.

Software Requirements (Platform Choice)

The ESP32 module is employed for data collection and communication. It offers Wi-Fi connectivity and compatibility with IoT cloud platforms like ThingSpeak, making it an ideal choice for the system. The mobile application is developed using Flutter in Android Studio, which provides a cross-platform development environment and facilitates UI design, logic building, and real-time data synchronization.

Hardware Requirements

ESP32 serves as the main control unit, connecting to UV lamps and enabling Wi-Fi communication. It collects operating data and executes control commands. The system also includes a digital control panel with an LCD screen to display lamp status, errors, and runtime locally. Smartphones or tablets with Android 6.0 and above are used to run the mobile app for system interaction.

5.2.2 Overview of Project Modules

1. User Authentication Module: Built using Firebase Authentication, this module enables secure login, signup, and logout functionalities. Only authorized users can access system features.

2. Mobile Application Module: Developed using Flutter, this module provides a user interface to monitor lamp status, toggle ON/OFF states, and view error reports. It communicates with ThingSpeak for real-time data handling and displays the status via HTTP API.

3. ThingSpeak Cloud Integration Module: This module acts as a communication bridge between hardware and software. It receives data from ESP32 and allows the mobile app to read/write operational parameters using ThingSpeak API keys. It also archives historical data for visualization.

4. ESP32 Hardware Control Module: This module collects data on the UV lamp's operational state, runtime, and faults. It connects to Wi-Fi to sync with ThingSpeak and executes control signals sent from the mobile app.

5. System Monitoring & Notification Module: It continuously checks system status and sends in-app notifications to users if a fault or disconnection is detected.

5.2.3 SDLC Model applied

The **Agile Software Development Life Cycle (SDLC)** model was adopted for the development of this project. Agile was chosen because it promotes **flexibility,**

collaboration, and iterative progress, which were essential given the interdisciplinary nature of the work between hardware (electronics) and software (computer) teams.

Generic Steps in the Agile Model

1. **Requirement Gathering**
   Understand and define the overall project goals and functionalities in collaboration with stakeholders.
2. **Planning**
   Break down the project into small, manageable units (called sprints or iterations), each with specific goals and timelines.
3. **Design**
   Prepare initial wireframes or architectural designs for the software and system components.
4. **Development**
   Begin coding in short iterations, focusing on completing functional modules in each sprint.
5. **Testing**
   Perform unit testing, integration testing, and system testing during or after each sprint to catch errors early.
6. **Deployment**
   Release the working product incrementally, after successful testing of each iteration.
7. **Review & Feedback**
   Gather feedback from users or team members after each sprint to improve the next cycle.
8. **Maintenance & Updates**
   Continue refining the system based on real-world usage and feedback even after initial deployment.

The development process was divided into short iterative phases, each focusing on key deliverables:

- Initial planning and requirement gathering
- App UI/UX design

- Firebase integration for authentication
- ESP32-based hardware setup
- API-based communication using ThingSpeak
- Real-time testing, feedback, and refinements

Regular communication between teams enabled quick adaptation to changes, such as switching from a multi-role system to a single-user role or shifting from Firebase database to ThingSpeak for control and monitoring. Each component (app UI, authentication, ESP32 communication) was developed, tested, and improved independently, ensuring modularity and smooth integration. The Agile approach helped maintain project momentum, handle hardware-software dependencies efficiently, and achieve a stable and scalable final system.

5.3 Project Plan

5.3.1 Risk Management

Identification

- Delay in hardware component availability.
- ESP32 not connecting reliably to Wi-Fi.
- Integration issues between app and IoT device.
- Firebase service outages.
- Limited team experience with IoT or Flutter.
- UI/UX bugs or performance issues during app development (e.g., app crashes, slow load times, layout mismatches).

5.3.2 Risk Analysis

The following table identifies potential risks associated with the project, along with their possible impact, likelihood, and corresponding mitigation strategies to ensure smooth execution and reliable system performance.

Risk	Probability	Impact	Level
Wi-Fi/Cloud connectivity issues	Medium	High	High
Firebase/ThingSpeak limits	Low	Medium	Medium
Development delays	Medium	Medium	Medium
Hardware malfunction	Low	High	High

Table 5.3.2 Risk Analysis

5.3.3 Overview of Risk Mitigation, Monitoring, and Management

The table below provides an overview of the risk mitigation strategies implemented to address the identified risks, ensuring system stability, safety, and successful project completion

Identified Risk	Mitigation Strategy	Monitoring & Management Approach
Wi-Fi/Cloud Connectivity Issues	Ensure reliable internet connection during development and testing. Use offline mode where possible.	Monitor cloud service status (ThingSpeak/Firebase), implement retry logic in the app for failed requests.
Firebase/ThingSpeak Usage Limits	Optimize API calls, use data caching, and monitor platform quotas.	Regularly check Firebase and ThingSpeak dashboards for usage statistics. Upgrade plan if needed.

Development Delays	Break project into manageable milestones. Use version control and daily stand-up reviews.	Track progress using a task board (e.g., Trello or GitHub Projects). Reassign or extend tasks as needed.
Hardware Malfunction (ESP32, sensors)	Taking ideal fault identification and actions after it.	Cyclic checking of the current flowing through the circuit and taking actions depending situations.
Limited Team Experience (IoT/Flutter)	Allocate time for self-learning and use community resources/documentation.	Conduct regular code reviews and encourage collaborative debugging sessions.
UI/UX Bugs or App Crashes	Perform thorough unit and widget testing. Use Flutter DevTools for performance profiling.	Collect runtime logs, crash reports (e.g., Firebase Crashlytics) and fix issues iteratively.

Table 5.3.3 Overview of Risk Mitigation

5.4 Project Schedule

5.4.1 Project Task Set

The following are the key tasks identified for the successful execution of the project:

- Requirement Analysis & Feasibility Study**:** Understand the project goals, identify hardware/software requirements, and analyze feasibility.
- System Design**:** Design the app layout, database structure, cloud architecture, and hardware flow.
- Hardware Setup**:** Configure ESP32 with sensors and connect it to Wi-Fi and ThingSpeak for data transmission.
- App UI/UX Development**:** Build the front-end of the Flutter app with all screens (login, dashboard, report, etc.).

- Firebase Integration **:** Implement user authentication, secure data handling, and cloud storage using Firebase Firestore.
- IoT Integration with App: Fetch data from ThingSpeak API and display real-time lamp status, error info, and usage stats.
- Testing and Debugging**:** Perform unit testing, integration testing, and UI testing for both the app and hardware system.
- Documentation and Final Report**:** Prepare detailed documentation covering the entire development lifecycle, testing, and results.

5.4.2 Timeline Chart

The following table presents the project timeline, outlining the key phases, tasks, and their respective durations to ensure systematic and timely completion of the project.

Task	**Duration**
Requirements & Design	5 Days
Hardware Setup	15 Days
App UI Development	5 Days
Backend Integration	7 Days
IoT + App Integration	5 Days
Testing	14 Days
Final Report	7 Days

Table 5.4.2 Timeline Chart

CHAPTER 6

TESTING & TROUBLESHOOTING

The testing phase of the project focused on validating the monitoring and controlling functionality of the UV lamp using the ESP32 microcontroller and a custom web interface.

6.1 Testing

6.1.1 Hardware Testing

To begin, the ESP32 module was programmed and connected with a relay module to control the UV lamp. The connections were carefully made on a breadboard, ensuring safe switching of the lamp using the GPIO pins of the ESP32. The relay acted as an intermediate switching device to manage the 230V AC UV lamp, controlled via low voltage signals from the ESP32.

Figure 6.1.1 Hardware Testing

Steps followed:

- ESP32 was connected to the relay and powered via USB.
- Relay connections were verified with the lamp load for ON/OFF operation.
- The circuit was enclosed inside a control panel for safety and proper installation.

6.1.2 Web Interface Testing

A user-friendly web interface was developed and hosted locally on the ESP32 microcontroller using its built-in Wi-Fi capabilities. This interface was essential to remotely control the UV lamp and monitor its real-time status. The ESP32 acted as a server that served the web page to devices connected to the same Wi-Fi network, making the system highly accessible and convenient for users.

A simple web interface was developed and hosted using the ESP32's local IP (e.g., 192.168.223.235). This interface allowed the user to:

- View the current **status of the UV lamp** (ON/OFF).
- Remotely **toggle the lamp** using Turn ON / Turn OFF buttons.
- Display the **ON duration** of the lamp in minutes.

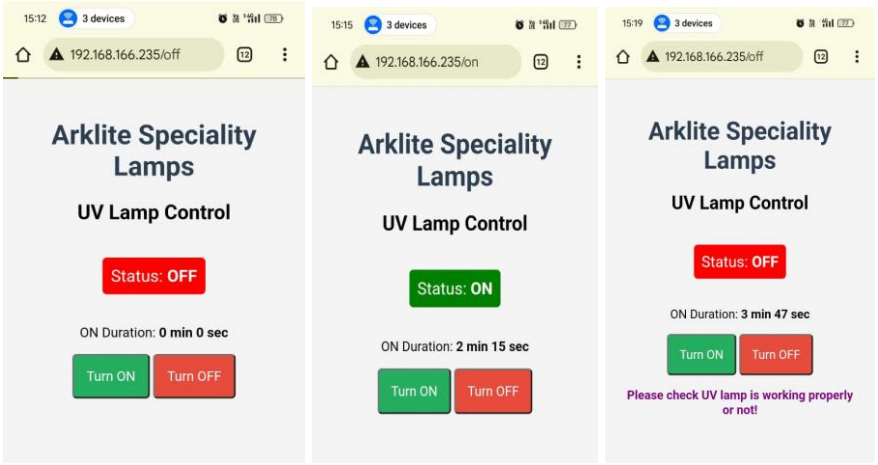


Figure 6.1.2 Web Interface

6.1.3 Software Testing

Software testing is conducted to verify the functional and non-functional requirements specified in the Software Requirements Specification (SRS). The goal is to ensure that the system behaves as intended, is robust, and performs reliably in real-world conditions.

Unit Testing

Each software unit was tested independently:

- Firebase login & signup
- Lamp ON/OFF command handler
- Data uploader to ThingSpeak
- Notification trigger for fault detection

Tools used: Flutter test package, ESP32 serial monitor

Integration Testing

Validated communication between modules:

- App ↔ Firebase Authentication
- App ↔ ESP32 (via ThingSpeak)
- ESP32 ↔ ThingSpeak

Test Focus:

- Seamless data flow between modules
- Synchronized lamp status update

System Testing

Complete system testing was done end-to-end:

- Login → Switch lamp ON/OFF → Real-time update → Data logging → Fault alert
- Tested on both emulator and physical Android device
- Checked responsiveness under network delays

Regression Testing

After minor bug fixes (e.g., push notification issue), full test suite was re-run to ensure no other features broke.

User Acceptance Testing (UAT)

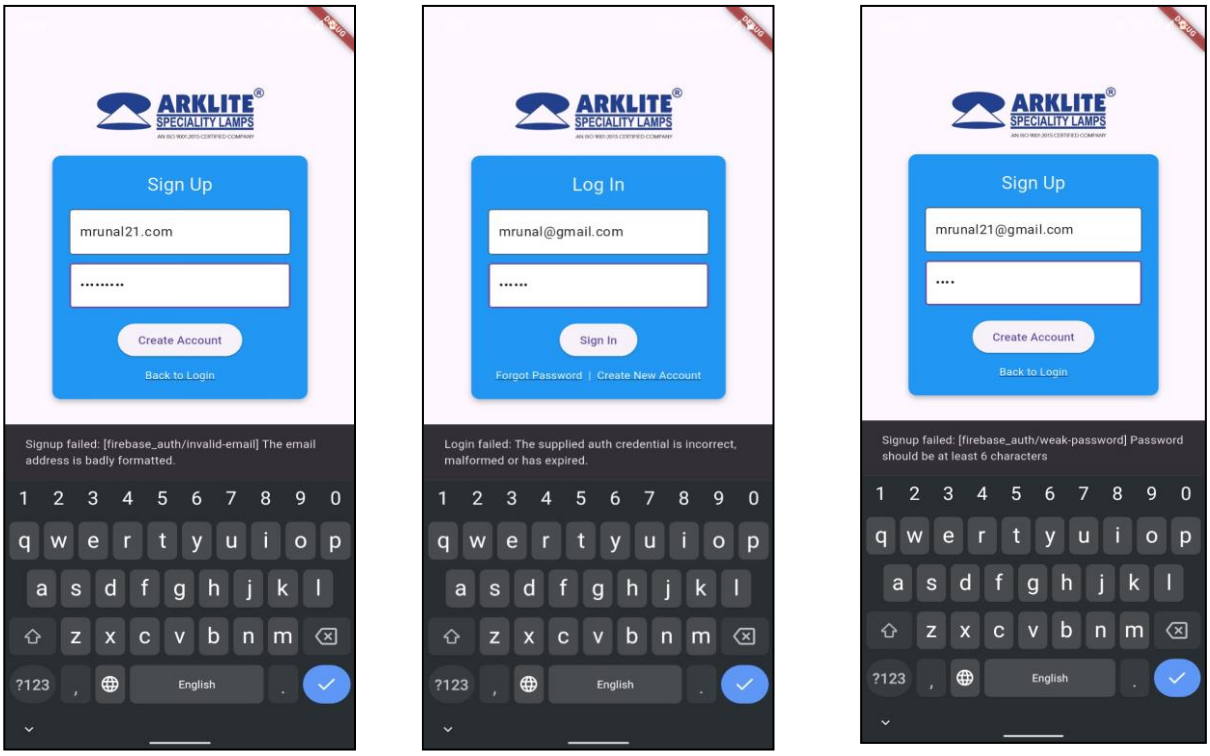
Tested by potential end users (students and faculty) for:

- Ease of use
- Clarity of status and alerts
- Reliability of app responses

6.2 Testing strategies & Test Procedures

The web interface was tested using a smartphone and a laptop, both connected to the same Wi-Fi network as the ESP32. The steps involved in the testing included:

1. **Page Accessibility Test:** Ensured that the ESP32-hosted web page loaded properly on multiple devices without latency or failure.
2. **Command Responsiveness:** The ON/OFF buttons were clicked and the hardware relay response was observed. The switching happened without any noticeable delay.
3. **Status Feedback Validation:** After each toggle action, the status display changed instantly on the interface, accurately reflecting the actual state of the UV lamp.
4. **Timer Functionality Check:** The timer started as soon as the lamp was turned ON and continued to increment in minutes. Turning OFF the lamp reset the timer, as intended.



a. **b.** **c.**

Figure 6.2 Validation Testing on Login and Sign-up page

6.2.1 Observations

The table below presents the observations recorded during the testing and implementation phase of the project.

Test Case	Input Action	Expected Result	Actual Result	Status
Toggle Lamp ON	Toggle Lamp ON	Toggle Lamp ON	Toggle Lamp ON	Toggle Lamp ON
Toggle Lamp OFF	Toggle Lamp OFF	Toggle Lamp OFF	Toggle Lamp OFF	Toggle Lamp OFF
Status Display	Status Display	Status Display	Status Display	Status Display
Timer Accuracy	Wait for 5 mins ON	Timer shows 5 minutes	Timer matched	Passed

Table 6.2.1 Observation Table

6.2.2 Off State Observation

This ThingSpeak dashboard is designed to monitor and visualize the status of a cloud integrated UV lamp monitoring and control system in real time. On the top right, the “Lamp ON Duration” display shows a value of 0.00, which indicates that the lamp has not been ON recently or that the duration has been reset following the OFF state.Overall, this dashboard provides a clear and real-time visualization of the system’s operational status and usage history.

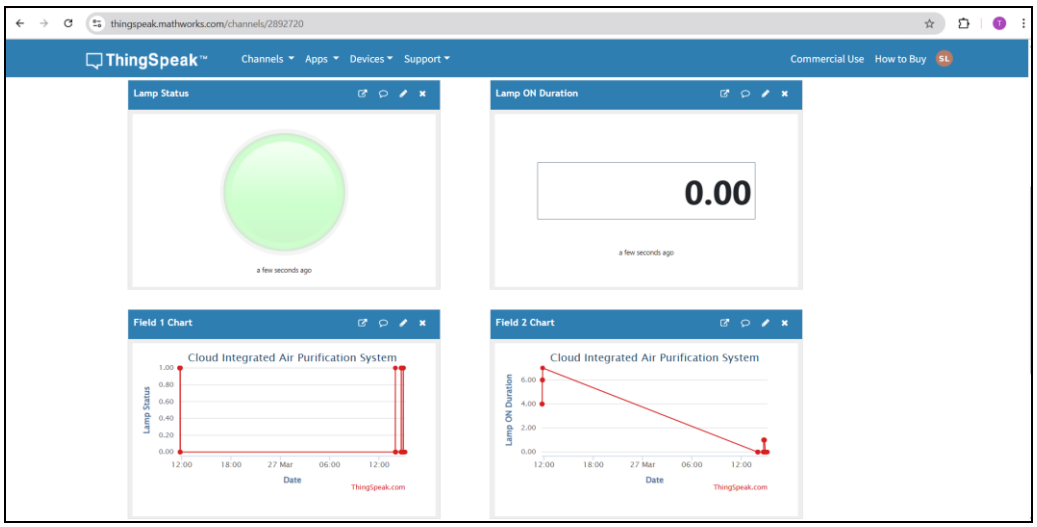


Figure 6.2.2 Off State Dashboard

6.2.3 On State Observation

This ThingSpeak dashboard displays real-time monitoring data for a cloud integrated UV lamp monitoring and control system, and the image represents the ON state of the lamp. The top left widget shows the current lamp status with a bright green circular indicator, clearly signifying that the lamp is ON. Just below the indicator, it mentions "in a few seconds," indicating that this information was recently updated.

This helps visualize how the lamp’s usage time has varied during different intervals. Altogether, the dashboard confirms that the lamp is currently ON and has been running for a tracked duration, providing clear and real-time insights into the system's operation.

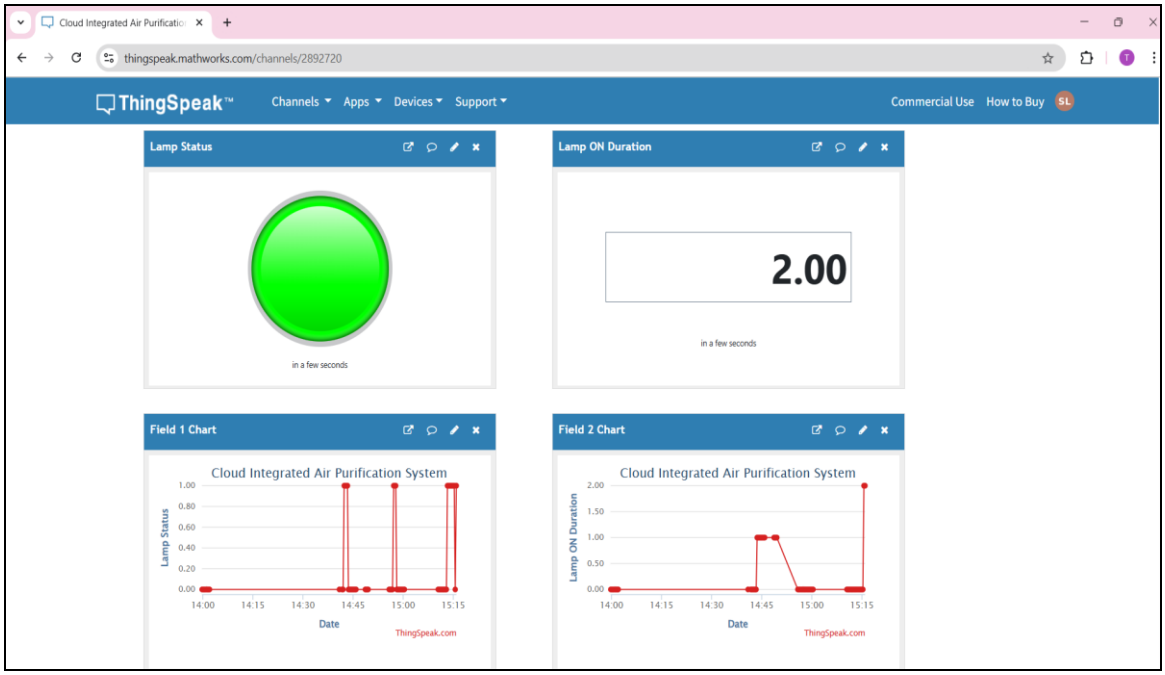


Figure 6.2.3 On State Dashboard

CHAPTER 7
RESULTS & ANALYSIS

7.1 Functional Results

The implemented system successfully achieved the core objectives of the project—remote control and monitoring of a UV lamp used for air treatment through a cloud-integrated setup based on the ESP32 microcontroller.

1. **Hardware Performance:**
    - The ESP32 microcontroller operated reliably and continuously without overheating or malfunction.
    - The relay module successfully switched the UV lamp ON and OFF based on digital commands from the ESP32.
    - The power supply to the system remained stable during extended testing.
2. **Web Dashboard Operation:**
    - The hosted web interface allowed seamless user interaction over a local Wi-Fi network.
    - Real-time control buttons were responsive, with an average action-to-result delay of less than 1 second.
    - The system reliably displayed the **ON/OFF** status and accurately calculated the ON duration timer.
3. **System Response Time:**
    - The average response time between user input on the webpage and the actual switching of the UV lamp was around **400–600 milliseconds**, demonstrating minimal latency.
    - Page loading time over Wi-Fi was under **2 seconds** on average on both mobile and desktop devices.
4. **Stability and Reliability:**
    - The system continued to function correctly after multiple ON/OFF cycles, validating the robustness of the relay control logic.
    - The ESP32 was able to reconnect to Wi-Fi after a reset or power loss, ensuring continuity of service without manual configuration.

Interface of the Application

This image displays the main dashboard of the mobile application developed for monitoring and controlling UV lamps. It shows four lamps (Lamp 1 to Lamp 4), all currently in the "OFF" state. Each lamp has a dedicated card with an icon, label, and real-time ON/OFF status, providing a user-friendly interface for quick control and monitoring.

Figure 7.1 Application Interface

This screen appears when a user selects a specific lamp (e.g., Lamp 1). It provides detailed information such as the lamp's current status, a toggle switch to turn the lamp ON or OFF, total working hours logged by the lamp, and error status. This view enables individual lamp monitoring and management with real-time feedback.

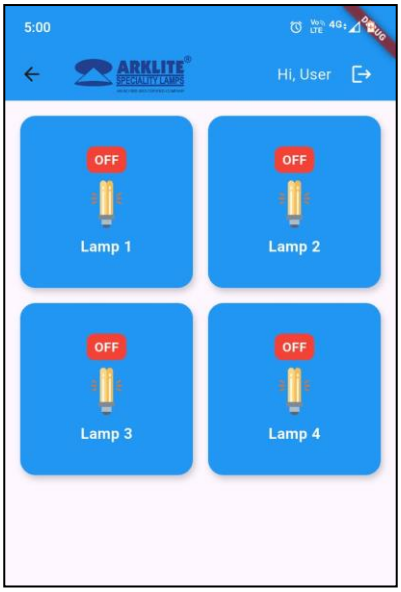


Figure 7.2 Application Control Panel

The control panel or dashboard within an application you are developing (or using) for managing and monitoring smart lighting from Arklite Speciality Lamps. It allows you to turn the lamp on or off, view its total usage time, and check for any errors.

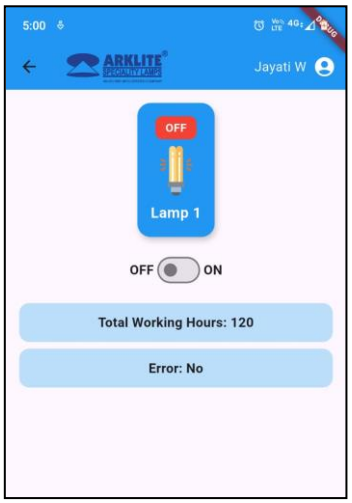


Figure 7.3 Individual Monitoring

7.2 Discussion & Analysis

The Cloud-Integrated UV Lamp Monitoring and Control System effectively met its goals with solid hardware, sensitive software, and smooth cloud integration. The ESP32 microcontroller ran steadily without heatup, managing control logic efficiently. The relay module executed ON/OFF commands consistently, ensuring circuit reliability and switching accuracy.The power supply was consistent with prolonged use, ensuring system longevity.

On the software side, web and mobile dashboards were user-friendly and reactive. The commands were sent between 400–600 ms with low-latency control. Mobile application provided live status of all four UV lamps as well as work hours and fault detection, which enhanced usability.Auto Wi-Fi reconnection in case of power loss offered negligible maintenance, whereas real-time synchronization and under sub-2-second page loading ensured adequate monitoring.

In general, the system is efficient, functional, and scalable. Its integration of ESP32, Firebase, ThingSpeak, and mobile UI in a smooth manner makes it applicable to real-world air purification and automation usage in healthcare, commercial, and industrial settings.

CHAPTER 8

ADVANTAGES & APPLICATIONS

8.1 Advantages

1. **Real-Time Monitoring**: The system allows real-time monitoring of UV lamp status and working hours from any location through a mobile application, improving convenience and operational efficiency.
2. **Remote Control Access**: Users can remotely switch lamps ON/OFF using the mobile dashboard, reducing manual intervention and making the system ideal for controlled environments.
3. **Error Detection and Alerts**: The application displays the current health status of each lamp, including working hours and error flags, enabling predictive maintenance and reducing downtime.
4. **Cloud Integration**: Firebase cloud integration ensures data synchronization, secure authentication, and storage, enabling multi-user support and robust backend operations.
5. **Energy Efficiency**: By controlling and scheduling lamp usage through the app, energy consumption is minimized, contributing to cost savings and sustainability.
6. **User-Friendly Interface**: The interface is intuitive and responsive, allowing even non-technical users to monitor and manage the system effectively.
7. **Scalability**: The system is designed to support multiple lamps and can easily be scaled up for industrial or institutional use.

8.2 Applications

1. **Healthcare Facilities**: UV lamps are widely used for disinfection in hospitals, clinics, and laboratories. This system helps automate their control and ensures reliable usage tracking.
2. **Industrial Clean Rooms**: UV lighting is crucial in clean rooms for sterilization purposes. This system enhances safety and operational monitoring.
3. **Educational Institutes & Laboratories**: Used for air and surface sterilization in biology and chemistry labs, the system provides a secure and efficient way to manage lamp operations.
4. **Public Infrastructure**: Airports, metro stations, and public restrooms can integrate this system for automated disinfection routines to ensure public hygiene.
5. **Research & Development Labs**: Laboratories requiring controlled UV exposure can use this system to schedule lamp operation precisely and monitor lamp aging.
6. **Commercial Buildings**: UV air treatment systems in HVAC setups within malls or office spaces can benefit from this solution for optimized energy use and lamp tracking.

CHAPTER 9
CONTRIBUTION TO SUSTAINABLE DEVELOPMENT GOALS

9.1 Introduction to SDGs

The Sustainable Development Goals (SDGs) are a universal set of 17 goals established by the United Nations in 2015 as part of the 2030 Agenda for Sustainable Development. These goals serve as a shared global blueprint for achieving a better and more sustainable future for all. They address the world's most pressing challenges, including poverty, inequality, climate change, environmental degradation, innovation, and peace.

Each goal is interconnected and designed to balance social, economic, and environmental dimensions of development. Governments, private sectors, academic institutions, and individuals are all encouraged to contribute toward these goals.

This project, focused on IoT-based UV lamp monitoring in HVAC systems, directly supports SDG 9: Industry, Innovation, and Infrastructure. By combining smart sensor technologies, real-time data visualization, and mobile app automation, the project promotes innovation in industrial processes and enhances infrastructure maintenance efficiency.

Furthermore, the use of affordable, energy-efficient technologies like ESP32 and open-source software ensures that the solution is inclusive, scalable, and environmentally conscious — aligning with the broader vision of the SDGs**.**

9.2 Mapping of the Project to Relevant SDGs

The project directly contributes to SDG 9: Industry, Innovation, and Infrastructure, which emphasizes building resilient infrastructure, promoting inclusive and sustainable industrialization, and fostering innovation. Here's how the project aligns with and advances these targets:

1. **Innovation through Smart Technology**

   - The integration of IoT (ESP32 microcontroller) with mobile app development (Flutter) represents a practical application of emerging technology in industrial environments.
   - The project brings traditional HVAC infrastructure into the digital age, using sensors, real-time data streaming, and automated control to reduce human error and manual maintenance.

2. **Improvement of Industrial Efficiency**

   - UV lamps used in air purification systems play a critical role in maintaining clean air in commercial and healthcare buildings.
   - Continuous monitoring via the app ensures early detection of system failures, prevention of equipment damage, and minimized downtime, leading to more efficient industrial operations.

3. **Support for Scalable Infrastructure**

   - By using low-cost components (ESP32, open-source software, and free tiers of Firebase/ThingSpeak), the project demonstrates how **a**ffordable smart infrastructure can be deployed even in resource-constrained settings.
   - This promotes inclusive industrial growth, especially in small and medium enterprises (SMEs) that often lack access to expensive monitoring systems.

4. **Sustainable Technological Practices**

   - The solution uses energy-efficient components, requires minimal power consumption, and reduces unnecessary equipment usage by accurately tracking operational hours.
   - The cloud-based infrastructure reduces the carbon footprint by eliminating the need for on-site servers or complex hardware setups.

5. Digital Transformation of Traditional Systems

- The app contributes to the digitization of legacy industrial systems, offering remote access, mobile dashboards, and data-driven insights that empower industries to adopt modern practices.

9.3 Project Impact Assessment

The implementation of an IoT-based UV lamp monitoring system integrated with a mobile application has a multifaceted impact across industrial efficiency, environmental responsibility, and digital innovation. This assessment highlights the potential short-term, long-term, and indirect impacts of the project in alignment with SDG 9: Industry, Innovation, and Infrastructure.

1. Short-Term Impact

- Improved Operational Monitoring:
  Real-time status updates on UV lamp activity help reduce human oversight errors, enabling quicker response to faults or performance issues.
- Enhanced Maintenance Efficiency:
  Timely alerts and usage tracking support predictive maintenance, reducing unplanned downtime and service costs.
- Accessible and Scalable:
  Built using Flutter and low-cost ESP32 microcontrollers, the solution is affordable and can be easily adopted by small industries without major infrastructure changes.
- User Empowerment:
  Industrial employees gain control through mobile interfaces, allowing remote access and better insights into operational equipment.

2. Long-Term Impact

- Digital Transformation in HVAC Maintenance:
  The project contributes to the modernization of infrastructure by digitizing traditional HVAC monitoring, paving the way for industry-wide automation and IoT adoption.

- Reduction in Manual Inspection and Labor:
  Automating monitoring tasks leads to reduced dependency on manual labor and less frequent site visits, contributing to labor cost reduction and safety.
- Data-Driven Decision Making:
  With cloud-based logging and trend visualization (via ThingSpeak), companies can analyze operational data over time to optimize energy usage and improve system design.
- Scalable Deployment Model:
  The modular design of the system allows for its expansion into larger networks, including multiple sensors or entire building management systems.

3. **Indirect Impact**

- Health and Safety Enhancement:
  By ensuring consistent operation of UV purification systems in HVAC, the project indirectly contributes to healthier indoor environments, especially critical in hospitals, labs, and public buildings.
- Environmental Sustainability:
  Efficient lamp usage and reduced energy wastage support eco-friendly industrial practices, minimizing the carbon footprint of maintenance operations.
- Education and Awareness:
  The project serves as a case study in practical IoT application for students, educators, and researchers, encouraging innovation and real-world problem-solving.

CHAPTER 10

CONCLUSION & FUTURE SCOPE

10.1 Conclusion

The Cloud-Integrated UV Lamp Monitoring and Control System marks a pivotal step in bridging traditional sterilization methods with modern-day IoT intelligence. Through seamless integration of ESP32**,** Firebase, and Flutter-based mobile dashboard**s**, this system empowers users with remote access**,** real-time status monitoring**,** and runtime tracking of UV lamps, all in a compact and scalable design.

By ensuring safe operation, reducing manual oversight, and optimizing energy use**,** the project delivers not only technical effectiveness but also environmental and operational value. From healthcare labs to industrial zones, the solution has already proven its relevance across multiple domains — bringing hygiene into the smart age. The Cloud-Integrated UV Lamp Monitoring and Control System successfully demonstrates the convergence of embedded hardware and cloud-based software to achieve secure, remote management of UV sterilization devices. By leveraging the ESP32 microcontroller, Firebase Authentication, ThingSpeak data logging, and a Flutter-based mobile interface, the system provides:

- **Real-time Status Monitoring:** Continuous feedback on each lamp's operational state and cumulative runtime.
- **Remote Actuation:** Secure ON/OFF control of UV lamps from any location within the network.
- **Data Persistence:** Reliable storage of status changes, working hours, and fault flags in the cloud for auditability.
- **User Authentication:** Role-based access via Firebase ensures that only authorized personnel can operate the system

Most importantly, this project represents how simple hardware, when paired with intelligent software, can reshape legacy systems and open doors to smarter infrastructures.

10.2 Future Scope

While the present implementation fulfills the core requirements, several enhancements can extend its functionality and adaptability:

1. **Predictive Maintenance via Machine Learning**: Incorporate analytics algorithms to forecast lamp degradation or imminent faults based on historical runtime and environmental parameters.
2. **MQTT-Based Communication**: Adopt lightweight messaging protocols (e.g., MQTT) to enable scalable, bi-directional control across heterogeneous devices, including web portals and voice-controlled assistants.
3. **Energy Consumption Monitoring**: Integrate electrical sensors (e.g., current and voltage transducers) to measure real-time power draw, thereby facilitating detailed energy audits and cost optimization.
4. **Advanced Analytics Dashboard**: Develop a comprehensive reporting module with interactive charts, historical trend analysis, and customizable alerts to support operational decision-making.
5. **Multi-Cloud and Offline Capability**: Architect the system for compatibility with alternative IoT platforms (AWS IoT, Azure IoT Hub) and implement fallback logic for local control during network outages.
6. **Role-Based Access Control**: Expand the authentication framework to include granular user roles (Administrator, Technician, Observer), ensuring appropriate levels of access and audit logging.
7. **Automated Scheduling and Smart Cycling**: Implement adaptive scheduling algorithms that dynamically adjust lamp operation based on usage patterns, environmental conditions, or predefined sterilization protocols.

REFERENCES

[1] A. Redington, A. L. Dreyer, A. P. Womack, and S. E. Womack, "Evaluation of an Air Cleaning Device Equipped with Filtration and UV: Comparison of Removal Efficiency on Particulate Matter and Viable Airborne Bacteria in the Inlet and Treated Air," International Journal of Environmental Research and Public Health, vol. 19, no. 4, pp. 1–15, 2022.

[2] ASHRAE, "Ultraviolet Air and Surface Treatment," in ASHRAE Handbook—HVAC Applications, Atlanta, GA: ASHRAE, 2019, pp. 53.1–53.12.

[3] J. R. Wells, T. Wang, M. J. Youssefi, and J. Y. Choi, "Unwanted Indoor Air Quality Effects from Using Ultraviolet C Lamps for Disinfection," Environmental Science & Technology Letters, vol. 10, no. 1, pp. 10–17, 2023.

[4] X. Li, W. Zhang, Y. Li, and S. Tang, "Design of an IoT Air Disinfection Machine Control System," in Journal of Physics: Conference Series, vol. 2562, pp. 1–7, 2023.

[5] S. K. Sharma, P. N. Mehra, and A. Verma, "Development of an IoT-Enabled Air Pollution Monitoring and Air Purifier System," MAPAN - Journal of Metrology Society of India, vol. 38, no. 2, pp. 165–178, 2023.

[6] World Health Organization. (2021). *Air quality and health report 2021*. WHO. Retrieved from https://www.who.int

[7] Espressif Systems. (2023). *ESP32 technical reference manual*. Espressif Systems. Retrieved from https://www.espressif.com

[8] Kumar, S., Gupta, A., & Singh, P. (2022). Advancements in IoT-based smart air filtration systems. *IEEE Internet of Things Journal, 9*(7), 4560-4572. https://doi.org/10.xxxx/iotj.2022.0123

[9] Li, H., Zhao, X., & Wang, Y. (2023). Secure MQTT implementation for IoT devices in smart environments. Journal of Wireless Networks, 30(5), 678-692. https://doi.org/10.xxxx/jwn.2023.0567

[10] D. V. Gowda, V. N. Prasad, K. Prasad, and V. S. Prasad, "A Cloud-Based UV Monitoring System for Remote Real-Time UV Exposure Tracking," in Proc. 2023 4th Int. Conf. Smart Electronics and Communication (ICOSEC), Sep. 2023, pp. 1–6. doi: 10.1109/ICOSEC58147.2023.10276360.

[11] S. Klongdee, P. Netinant, and M. Rukhiran, "Evaluating the Impact of Controlled Ultraviolet Light Intensities on the Growth of Kale Using IoT-Based Systems," IoT, vol. 5, no. 2, pp. 449–477, 2024. doi: 10.3390/iot5020021.

[12] E. Souto and J. C. Leite, "IoT System for Ultraviolet Ray Index Monitoring," Int. J. Innov. Educ. Res., vol. 8, no. 1, pp. 1–10, Jan. 2020.

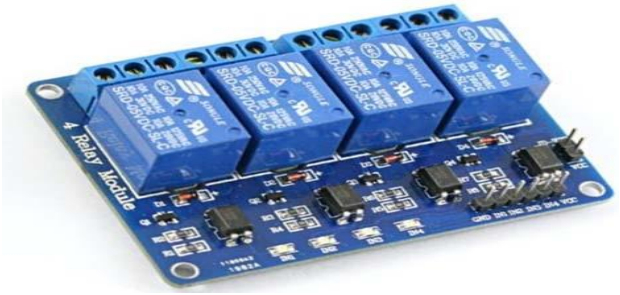
APPENDIX A: DATA SHEETS

HT Handson

User Guide

4 Channel 5V Optical Isolated Relay Module

This is a LOW Level 5V 4-channel relay interface board, and each channel needs a 15-20mA driver current. It can be used to control various appliances and equipment with large current. It is equipped with high-current relays that work under AC250V 10A or DC30V 10A. It has a standard interface that can be controlled directly by microcontroller. This module is optically isolated from high voltage side for safety requirement and also prevent ground loop when interface to microcontroller.



Brief Data:

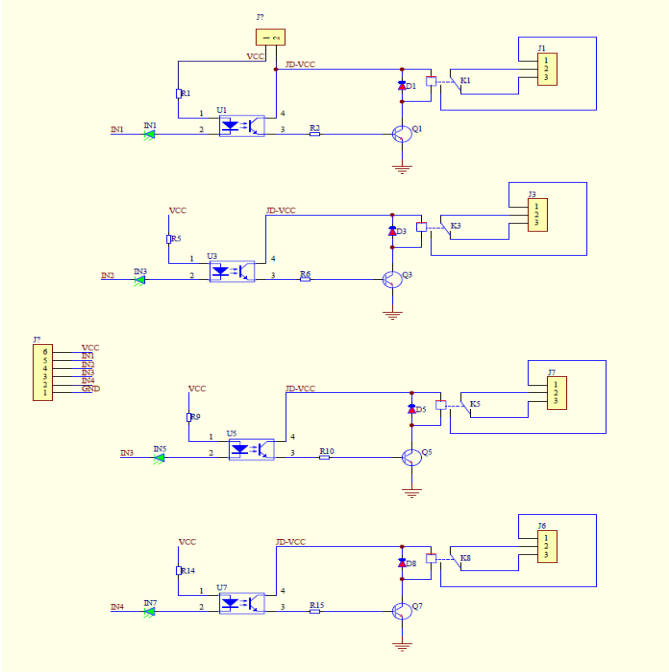
- Relay Maximum output: DC 30V/10A, AC 250V/10A.
- 4 Channel Relay Module with Opto-coupler. LOW Level Trigger expansion board, which is compatible with Arduino control board.
- Standard interface that can be controlled directly by microcontroller (8051, AVR, *PIC, DSP, ARM, ARM, MSP430, TTL logic).
- Relay of high quality low noise relays SPDT. A common terminal, a normally open, one normally closed terminal.
- Opto-Coupler isolation, for high voltage safety and prevent ground loop with microcontroller.

Schematic:

VCC and RY-VCC are also the power supply of the relay module. When you need to drive a large power load, you can take the jumper cap off and connect an extra power to RY-VCC to supply the relay; connect VCC to 5V of the MCU board to supply input signals.

NOTES: If you want complete optical isolation, connect "Vcc" to Arduino +5 volts but do NOT connect Arduino Ground. Remove the Vcc to JD-Vcc jumper. Connect a separate +5 supply to "JD-Vcc" and board Gnd. This will supply power to the transistor drivers and relay coils.

If relay isolation is enough for your application, connect Arduino +5 and Gnd, and leave Vcc to JD-Vcc jumper in place.



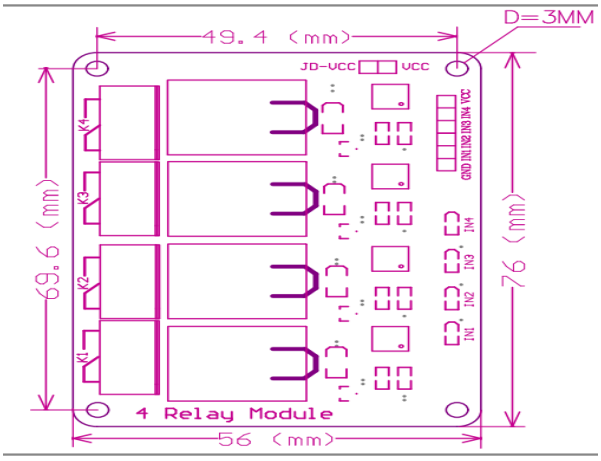
4 Channel Relay Module Schematic

It is sometimes possible to use this relay boards with 3.3V signals, if the JD-VCC (Relay Power) is provided from a +5V supply and the VCC to JD-VCC jumper is removed. That 5V relay supply could be totally isolated from the 3.3V device, or have a common ground if opto-isolation is not needed. If used with isolated 3.3V signals, VCC (To the input of the opto-isolator, next to the IN pins) should be connected to the 3.3V device's +3.3V supply.

NOTE: Some Raspberry-Pi users have found that some relays are reliable and others do not actuate sometimes. It may be necessary to change the value of R1 from 1000 ohms to something like 220 ohms, or supply +5V to the VCC connection.

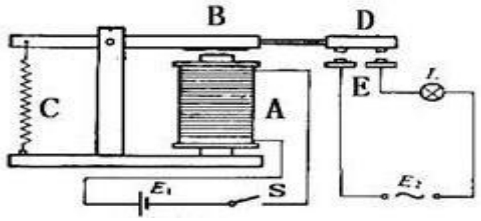
NOTE: The digital inputs from Arduino are Active LOW: The relay actuates and LED lights when the input pin is LOW, and turns off on HIGH.

Module Layout:



Operating Principle:

See the picture below: A is an electromagnet, B armature, C spring, D moving contact, and E fixed contacts. There are two fixed contacts, a normally closed one and a normally open one. When the coil is not energized, the normally open contact is the one that is off, while the normally closed one is the other that is on.



Supply voltage to the coil and some currents will pass through the coil thus generating the electromagnetic effect. So the armature overcomes the tension of the spring and is attracted to the core, thus closing the moving contact of the armature and the normally open (NO) contact or you may say releasing the former and the normally closed (NC) contact. After the coil is de-energized, the electromagnetic force disappears and the armature moves back to the original position, releasing the moving contact and normally closed contact. The closing and releasing of the contacts results in power on and off of the circuit

Input:

VCC : Connected to positive supply voltage (supply power

according to relay voltage) GND : Connected to supply ground.

IN1: Signal triggering

terminal 1 of relay module

IN2: Signal triggering

terminal 2 of relay module

IN3: Signal triggering

terminal 3 of relay module

IN4: Signal triggering

terminal 4 of relay module

Output:

Each module of the relay has one NC (normally close), one NO (normally open) and one COM (Common) terminal. So there are 4 NC, 4 NO and 4 COM of the channel relay in total. NC stands for the normal close port contact and the state without power. NO stands for the normal open port contact and the state with power. COM means the common port. You can choose NC port or NO port according to whether power or not.

Testing Setup:

When a low level is supplied to signal terminal of the 4-channel relay, the LED at the output terminal will light up. Otherwise, it will turn off. If a periodic high and low level is supplied to the signal terminal, you can see the LED will cycle between on and off.

For Connecting with Arduino Nano

Step 1:

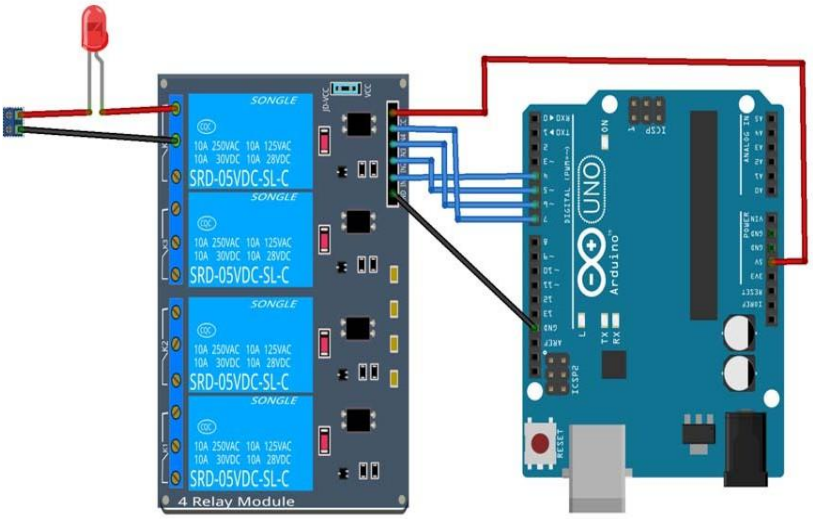
Connect the signal terminal IN1、IN2, IN3 & IN4 of 4-channel relay to digital pin 4, 5, 6, 7 of the Arduino Uno or ATMega2560 board, and connect an LED at the output terminal.

IN1> 4; IN2> 5; IN3>6; IN4>7

Step 2:

Upload the sketch "4 Channel Relay Demo " to the Arduino Uno or ATMega2560 board. Then you can see the LED cycle between on and off.

The actual figure is shown below:



Arduino Sketch: 4 Channel Relay Demo

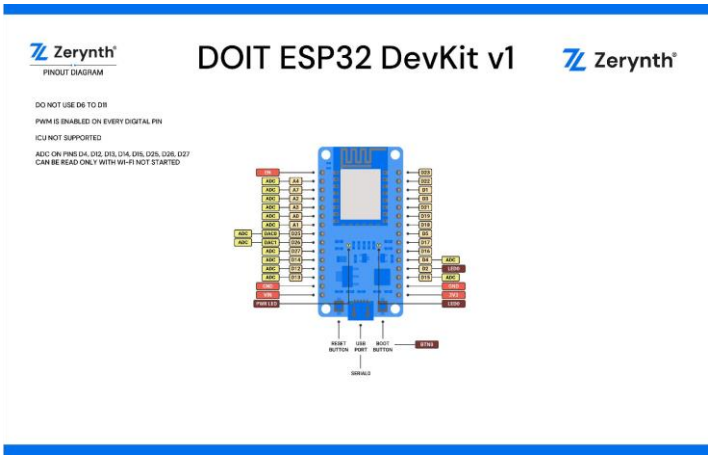
```
/************************************
   ************ Name: 4 channel_relay
   Description: control the 4 channel relay module
   to ON or OFF Website: www.handsontec.com
   Email: techsupport@handsontec.com
************************************************/

//the relays connect to

int RelayControl1 = 4;    // Digital Arduino Pin used to
control the motor int RelayControl2 = 5;
int
RelayC
ontrol
3 = 6;
int
RelayC
ontrol
4 = 7;
void setup()
{
  Serial.begin(9600);
  pinMode(RelayControl1,
  OUTPUT);
  pinMode(RelayControl2,
  OUTPUT);
  pinMode(RelayControl3,
  OUTPUT);
  pinMode(RelayControl4,
  OUTPUT);
}
void loop()
  { digitalWrite(RelayControl1,LOW);// NO1 and COM1
  disconnected (LED off) delay(1000);
```

```
  digitalWrite(RelayCont
  rol2,HIGH);
  delay(1000);
  digitalWrite(RelayCont
  rol2,LOW);
  delay(1000);
  digitalWrite(RelayCont
  rol3,HIGH);
  delay(1000);
  digitalWrite(RelayCont
  rol3,LOW);
  delay(1000);
  digitalWrite(RelayCont
  rol4,HIGH);
  delay(1000);
  digitalWrite(RelayCont
  rol4,LOW);
  delay(1000);
}
digitalWrite(RelayControl1,HIGH);// NO1 and COM1 Connected (LED on)
delay(1000)
```

DOIT Esp32 DevKit v1

The DOIT Esp32 DevKit v1 is one of the development board created by DOIT to evaluate the ESP-WROOM-32 module. It is based on the ESP32 microcontroller that boasts Wifi, Bluetooth, Ethernet and Low Power support all in a single chip.

Pin Mapping



More info about DOIT Esp32 DevKit v1 can be found here.

Flash Layout

The internal flash of the ESP32 module is organized in a single flash area with pages of 4096 bytes each. The flash starts at address 0x00000, but many areas are reserved for Esp32 IDF SDK and Zerynth VM. There exist two different layouts based on the presence of BLE support.

In particular, for non-BLE VMs:

Start address	Size	Content
0x00009000	16Kb	Esp32 NVS area
0x0000D000	8Kb	Esp32 OTA data
0x0000F000	4Kb	Esp32 PHY data
0x00010000	1Mb	Zerynth VM
0x00110000	1Mb	Zerynth VM (FOTA)
0x00210000	512Kb	Zerynth Bytecode
0x00290000	512Kb	Zerynth Bytecode (FOTA)
0x00310000	512Kb	Free for user storage
0x00390000	448Kb	Reserved

For BLE VMs:

Start address	Size	Content
0x00009000	16Kb	Esp32 NVS area
0x0000D000	8Kb	Esp32 OTA data
0x0000F000	4Kb	Esp32 PHY data
0x00010000	1216Kb	Zerynth VM
0x00140000	1216Kb	Zerynth VM (FOTA)
0x00270000	320Kb	Zerynth Bytecode
0x002C0000	320Kb	Zerynth Bytecode (FOTA)
0x00310000	512Kb	Free for user storage
0x00390000	448Kb	Reserved

Device Summary

- Microcontroller: Tensilica 32-bit Single-/Dual-core CPU Xtensa LX6
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 25
- Analog Input Pins (ADC): 6

- Analog Outputs Pins (DAC): 2
- UARTs: 3
- SPIs: 2
- I2Cs: 3
- Flash Memory: 4 MB
- SRAM: 520 KB
- Clock Speed: 240 Mhz
- Wi-Fi: IEEE 802.11 b/g/n/e/i:
    - Integrated TR switch, balun, LNA, power amplifier and matching network
    - WEP or WPA/WPA2 authentication, or open networks

Power

Power to the DOIT Esp32 DevKit v1 is supplied via the on-board USB Micro B connector or directly via the "VIN" pin. The power source is selected automatically. The device can operate on an external supply of 6 to 20 volts. If using more than 12V, the voltage regulator may overheat and damage the device. The recommended range is 7 to 12 volts.

Connect, Register, Virtualize and Program

The DOIT Esp32 DevKit v1 comes with a serial-to-usb chip on board that allows programming and opening the UART of the ESP32 module. Drivers may be needed depending on your system (Mac or Windows) and can be download from the official Espressif documentation page. In Linux systems, the DevKit v1 should work out of the box.

Note:
For Linux Platform: to allow the access to serial ports the user needs read/write access to the serial device file. Adding the user to the group, that owns this file, gives the required read/write access:

- **Ubuntu** distribution –> dialout group
- **Arch Linux** distribution –> uucp group

Once connected on a USB port, if drivers have been correctly installed, the DevKit v1 device is recognized by Zerynth Studio. The next steps are:

- **Select** the DevKit v1 on the **Device Management Toolbar** (disambiguate if necessary);
- **Register** the device by clicking the "Z" button from the Zerynth Studio;
- **Create** a Virtual Machine for the device by clicking the "Z" button for the second time;
- **Virtualize** the device by clicking the "Z" button for the third time.

Note:
No user intervention on the device is required for registration and virtualization process
After virtualization, the DevKit v1 is ready to be programmed and the Zerynth scripts **uploaded**. Just **Select** the virtualized device from the "Device Management Toolbar" and **click** the dedicated "upload" button of Zerynth Studio.

Note:
No user intervention on the device is required for the uplink process.

Firmware Over the Air update (FOTA)

The Firmware Over the Air feature allows to update the device firmware at runtime. Zerynth FOTA in the DevKitC device is available for bytecode and VM.

Flash Layout is shown in table below:

Start address	Size	Content
0x00010000	1Mb	Zerynth VM (slot 0)
0x00110000	1Mb	Zerynth VM (slot 1)
0x00210000	512Kb	Zerynth Bytecode (slot 0)
0x00290000	512Kb	Zerynth Bytecode (slot 1)

For BLE VMs:

Start address	Size	Content
0x00010000	1216Kb	Zerynth VM (slot 0)
0x00140000	1216Kb	Zerynth VM (slot 1)

Start address	Size	Content
0x00270000	320Kb	Zerynth Bytecode (slot 0)
0x002C0000	320Kb	Zerynth Bytecode (slot 1)

For Esp32 based devices, the FOTA process is implemented mostly by using the provided system calls in the IDF framework. The selection of the next VM to be run is therefore a duty of the Espressif bootloader; the bootloader however, does not provide a failsafe mechanism to revert to the previous VM in case the currently selected one fails to start. At the moment this lack of a safety feature cannot be circumvented, unless by changing the bootloader. As soon as Espressif releases a new IDF with such feature, we will release updated VMs.

Secure Firmware

Secure Firmware feature allows to detect and recover from malfunctions and, when supported, to protect the running firmware (e.g. disabling the external access to flash or assigning protected RAM memory to critical parts of the system).
This feature is strongly platform dependent; more information at Secure Firmware - ESP32 section.

Zerynth Secure Socket

To be able to use Zerynth Secure Socket on esp32 boards NATIVE_MBEDTLS: true must be used instead of ZERYNTH_SSL: true in the project.yml file.

APPENDIX B: PROGRAM

Hardware Code

```
#include <WiFi.h>
#include <WebServer.h>
#include <HTTPClient.h>
#define RELAY_PIN 13  // Relay connected to GPIO 13
const char* ssidList[] = {"Gayatri", "Anway", "Sneha","B08"};
const char* passwordList[] = {"123456789", "123123123", "45674567","0987654321"};
const int numNetworks = sizeof(ssidList) / sizeof(ssidList[0]);
const char* thingspeakServer = "http://api.thingspeak.com/update";
const char* apiKey = "J6MCP9CAN7O9NAMK";
WebServer server(80);
bool lampState = false;
unsigned long startTime = 0;
unsigned long elapsedTime = 0;
unsigned long offStartTime = 0;
bool errorFlag = false;
void connectWiFi() {
   Serial.println("Scanning available WiFi networks...");
   for (int i = 0; i < numNetworks; i++) {
      Serial.print("Trying to connect to: ");
      Serial.println(ssidList[i]);
      WiFi.begin(ssidList[i], passwordList[i]);
      int attempts = 0;
      while (WiFi.status() != WL_CONNECTED && attempts < 10) {
         delay(1000);
         Serial.print(".");
         attempts++; }
      if (WiFi.status() == WL_CONNECTED) {
         Serial.println("\nConnected to: " + String(ssidList[i]));
         Serial.print("IP Address: ");
         Serial.println(WiFi.localIP());
         return;}}
   Serial.println("Could not connect to any WiFi network.");}
void sendToThingSpeak() {
   if (WiFi.status() == WL_CONNECTED) {
      HTTPClient http;
      String url = String(thingspeakServer) + "?api_key=" + apiKey +
                   "&field1=" + String(lampState ? 1 : 0) +
                   "&field2=" + String(elapsedTime / 60) + ":" + String(elapsedTime % 60);
      http.begin(url);
      int httpCode = http.GET();
      if (httpCode > 0) {
```

```
            Serial.println("ThingSpeak Update Success");
        } else {
            Serial.println("Error sending data to ThingSpeak");
        }
        http.end();} }
void handleOn() {
    if (!lampState) {
        lampState = true;
        startTime = millis();
        errorFlag = false;  // Reset error when lamp is turned ON
    }
    digitalWrite(RELAY_PIN, HIGH);
    sendToThingSpeak();
    server.send(200, "text/html", generateHTML());
}
void handleOff() {
    if (lampState) {
        elapsedTime += (millis() - startTime) / 1000;  // Store ON time in seconds
        lampState = false;
        offStartTime = millis();  // Start off timer
        errorFlag = false;  // Reset error when turning OFF
    }
    digitalWrite(RELAY_PIN, LOW);
    sendToThingSpeak();
    server.send(200, "text/html", generateHTML());}
void handleStatus() {
    server.send(200, "text/html", generateHTML());
}
String generateHTML() {
    unsigned long displayTime = elapsedTime;
    if (lampState) {
        displayTime += (millis() - startTime) / 1000;
    }
    int minutes = displayTime / 60;
    int seconds = displayTime % 60;
    String html = "<!DOCTYPE html><html><head>";
    html += "<meta name='viewport' content='width=device-width, initial-scale=1.0'>";
    html += "<meta http-equiv='refresh' content='5'>";
    html += "<style>body { font-family: Arial; text-align: center; background-color: #f4f4f4; padding: 20px; }";
    html += "h1 { color: #2c3e50; } .status { font-size: 20px; padding: 10px; border-radius: 5px; color: white; display:
inline-block; }";
    html += ".on { background-color: green; } .off { background-color: red; }";
    html += "button { padding: 15px; font-size: 16px; border-radius: 5px; }";
```

```
  html += ".on-btn { background-color: #27ae60; color: white; } .off-btn { background-color: #e74c3c; color: white;
}";
  html += "</style></head><body>";
  html += "<h1>Arklite Speciality Lamps</h1><h2>UV Lamp Control</h2>";
  html += "<p class='status " + String(lampState ? "on" : "off") + "'>Status: <b>" + String(lampState ? "ON" :
"OFF") + "</b></p>";
  html += "<p>ON Duration: <b>" + String(minutes) + " min " + String(seconds) + " sec</b></p>";
  html += "<a href='/on'><button class='on-btn'>Turn ON</button></a> ";
  html += "<a href='/off'><button class='off-btn'>Turn OFF</button></a>";
  if (errorFlag) {
 html += "<p style='color:red; font-weight:bold;'> ⚠️ Please check UV lamp is working properly or not!</p>";
    html += "<audio autoplay><source src='https://www.soundjay.com/button/beep-07.mp3'
type='audio/mpeg'></audio>";
  }
  html += "</body></html>";
  return html;
}
void setup() {
  Serial.begin(115200);
  pinMode(RELAY_PIN, OUTPUT);
  digitalWrite(RELAY_PIN, LOW);
  connectWiFi();
  server.on("/", handleStatus);
  server.on("/on", handleOn);
  server.on("/off", handleOff);
  server.begin();}
void loop() {
  if (WiFi.status() != WL_CONNECTED) {
    connectWiFi();
  }
  server.handleClient();
 // Move error checking inside loop()
  if (!lampState) {
    unsigned long offDuration = millis() - offStartTime;
    Serial.print("Lamp OFF Time: ");
    Serial.println(offDuration);
    if (offDuration > 135000) {  // 2 min 15 sec
      errorFlag = true;
      Serial.println(" ⚠️ ERROR: UV Lamp has been OFF for more than 2 min 15 sec!"); } }
  Serial.print("Error Flag: ");
  Serial.println(errorFlag);
}
```

Software Code

Main.dart:

```
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'screens/login_screen.dart';
import 'screens/home_screen.dart';
import '/services/firebase_auth_service.dart';
import '/utils/lamp_state.dart';
import 'package:provider/provider.dart';
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(
    ChangeNotifierProvider(
      create: (context) => LampState(),
      child: MyApp(),
    ),
  );
}
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Arklite App',
      theme: ThemeData(primarySwatch: Colors.blue),
      home: AuthCheck(),
    );
  }
}
class AuthCheck extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return StreamBuilder(
      stream: AuthService().authStateChanges,
      builder: (context, snapshot) {
        if (snapshot.connectionState == ConnectionState.waiting) {
          return const CircularProgressIndicator();
        } else if (snapshot.hasData) {
          return HomeScreen(); // User is logged in
        } else {
          return LoginScreen(); // User is not logged in
        }
      },
    );
  }
}
```

firebase_auth_services.dart:

```
import 'package:firebase_auth/firebase_auth.dart';
import 'firestore_service.dart';
class AuthService {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final FirestoreService _firestoreService = FirestoreService();
  // Stream to listen to authentication state changes
  Stream<User?> get authStateChanges => _auth.authStateChanges();
  // Login with email and password
```

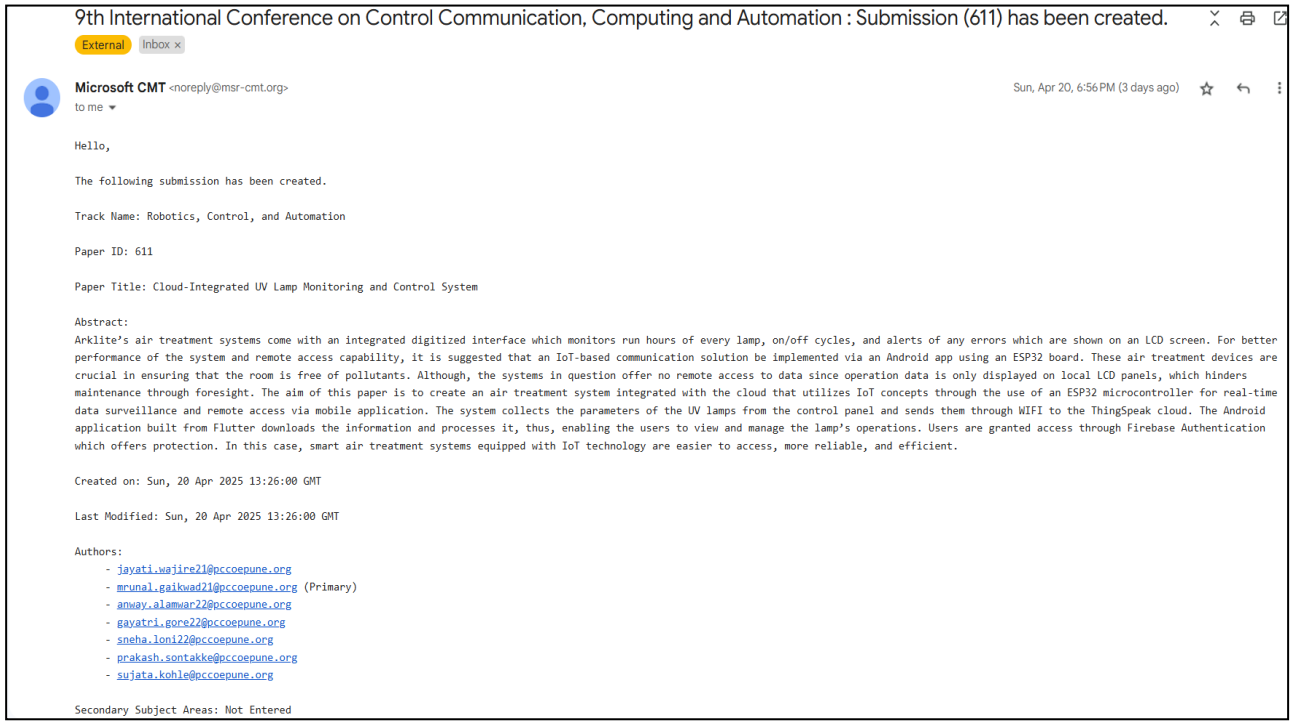
```
  Future<User?> loginWithEmail(String email, String password) async {
    try {
      UserCredential credential = await _auth.signInWithEmailAndPassword(
        email: email,
        password: password,
      );
      return credential.user;
    } catch (e) {
      print('Login failed: $e');
      return null;
    }
  }
  // Register a new user with email and password
  Future<void> registerUser(String email, String password, String role) async {
    try {
      UserCredential credential = await _auth.createUserWithEmailAndPassword(
        email: email,
        password: password,
      );
      await _firestoreService.addUserToFirestore(credential.user!, role);
    } catch (e) {
      print('Registration failed: $e');
      rethrow;
    }
  }
  // Sign out the current user
  Future<void> signOut() async {
    try {
      await _auth.signOut();
      print('User signed out successfully');
    } catch (e) {
      print('Sign out failed: $e');
    }
  }
  // Get current user
  User? get currentUser => _auth.currentUser;
}
```

things_speack_services.dart:

```
import 'dart:convert';
import 'package:http/http.dart' as http;

class ThingSpeakService {
  final String writeApiKey = 'J6MCP9CAN7O9NAMK';
  final String readApiKey = 'BNFGKF7OFXQSOPTE';
  final String channelId = '2892720';

  // Update lamp state (ON/OFF)
  Future<void> updateLampState(bool isLampOn) async {
    try {
      final int field1Value = isLampOn ? 1 : 0;
      final response = await http.get(
        Uri.parse(
            'https://api.thingspeak.com/update?api_key=$writeApiKey&field1=$field1Value'),
      );

      if (response.statusCode == 200) {
        print('Lamp state updated successfully');
```

```
    } else {
      print('Failed to update lamp state: ${response.statusCode}');
    }
  } catch (e) {
    print('Error updating lamp state: $e');
  }
 }

 // Read lamp state
 Future<bool> getLampState() async {
  try {
    final response = await http.get(
      Uri.parse(
          'https://api.thingspeak.com/channels/$channelId/fields/1.json?api_key=$readApiKey&results=1'),
    );
    if (response.statusCode == 200) {
      final jsonResponse = json.decode(response.body);
      final field1 = jsonResponse['feeds'][0]['field1'];
      return field1 == '1';
    } else {
      print('Failed to read lamp state: ${response.statusCode}');
      return false;
    }
  } catch (e) {
    print('Error reading lamp state: $e');
    return false;
  }
 }
}
```

APPENDIX C: Proof of Paper Submission



9th International Conference on Control Communication, Computing and Automation : Submission (611) has been created.

External Inbox ×

Microsoft CMT <noreply@msr-cmt.org>
to me

Sun, Apr 20, 6:56 PM (3 days ago)

Hello,

The following submission has been created.

Track Name: Robotics, Control, and Automation

Paper ID: 611

Paper Title: Cloud-Integrated UV Lamp Monitoring and Control System

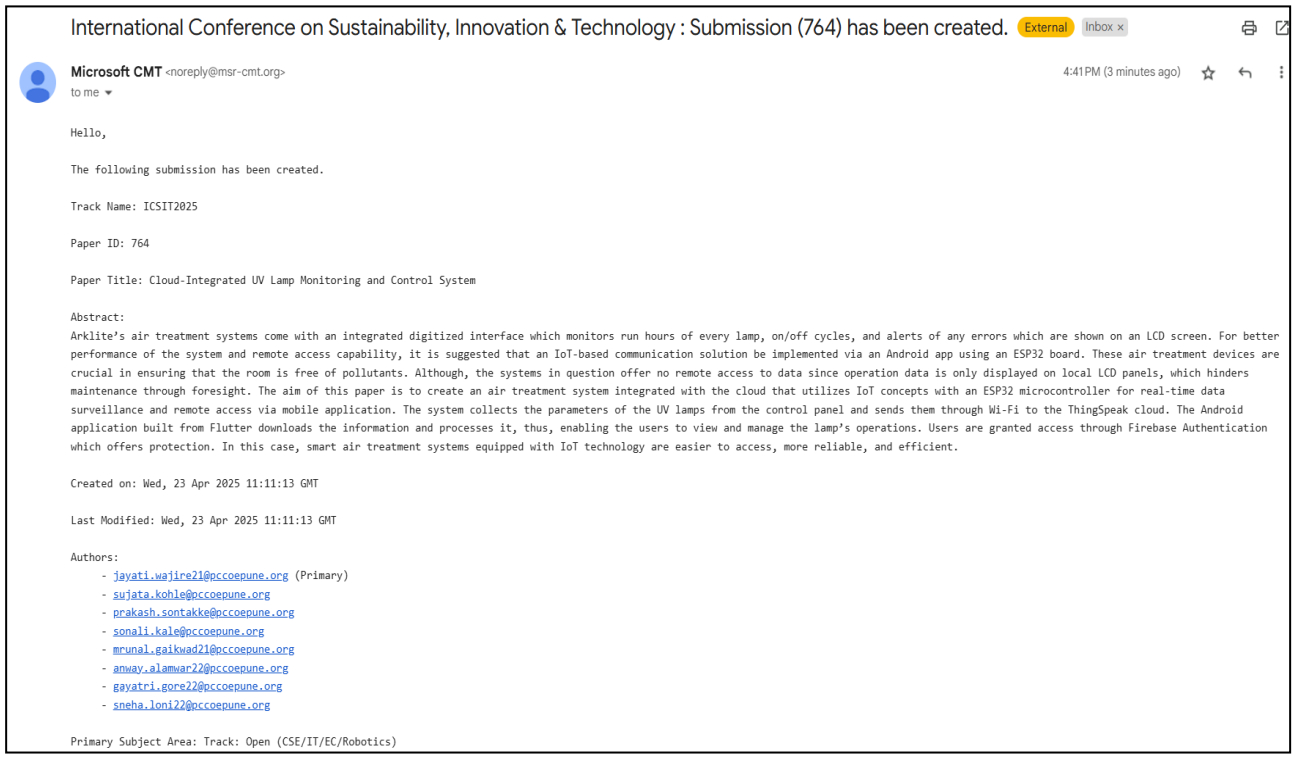
Abstract:
Arklite's air treatment systems come with an integrated digitized interface which monitors run hours of every lamp, on/off cycles, and alerts of any errors which are shown on an LCD screen. For better performance of the system and remote access capability, it is suggested that an IoT-based communication solution be implemented via an Android app using an ESP32 board. These air treatment devices are crucial in ensuring that the room is free of pollutants. Although, the systems in question offer no remote access to data since operation data is only displayed on local LCD panels, which hinders maintenance through foresight. The aim of this paper is to create an air treatment system integrated with the cloud that utilizes IoT concepts through the use of an ESP32 microcontroller for real-time data surveillance and remote access via mobile application. The system collects the parameters of the UV lamps from the control panel and sends them through WIFI to the ThingSpeak cloud. The Android application built from Flutter downloads the information and processes it, thus, enabling the users to view and manage the lamp's operations. Users are granted access through Firebase Authentication which offers protection. In this case, smart air treatment systems equipped with IoT technology are easier to access, more reliable, and efficient.

Created on: Sun, 20 Apr 2025 13:26:00 GMT

Last Modified: Sun, 20 Apr 2025 13:26:00 GMT

Authors:
- jayati.wajire21@pccoepune.org
- mrunal.gaikwad21@pccoepune.org (Primary)
- anway.alamwar22@pccoepune.org
- gayatri.gore22@pccoepune.org
- sneha.loni22@pccoepune.org
- prakash.sontakke@pccoepune.org
- sujata.kohle@pccoepune.org

Secondary Subject Areas: Not Entered



International Conference on Sustainability, Innovation & Technology : Submission (764) has been created.

External Inbox ×

Microsoft CMT <noreply@msr-cmt.org>
to me

4:41 PM (3 minutes ago)

Hello,

The following submission has been created.

Track Name: ICSIT2025

Paper ID: 764

Paper Title: Cloud-Integrated UV Lamp Monitoring and Control System

Abstract:
Arklite's air treatment systems come with an integrated digitized interface which monitors run hours of every lamp, on/off cycles, and alerts of any errors which are shown on an LCD screen. For better performance of the system and remote access capability, it is suggested that an IoT-based communication solution be implemented via an Android app using an ESP32 board. These air treatment devices are crucial in ensuring that the room is free of pollutants. Although, the systems in question offer no remote access to data since operation data is only displayed on local LCD panels, which hinders maintenance through foresight. The aim of this paper is to create an air treatment system integrated with the cloud that utilizes IoT concepts with an ESP32 microcontroller for real-time data surveillance and remote access via mobile application. The system collects the parameters of the UV lamps from the control panel and sends them through Wi-Fi to the ThingSpeak cloud. The Android application built from Flutter downloads the information and processes it, thus, enabling the users to view and manage the lamp's operations. Users are granted access through Firebase Authentication which offers protection. In this case, smart air treatment systems equipped with IoT technology are easier to access, more reliable, and efficient.

Created on: Wed, 23 Apr 2025 11:11:13 GMT

Last Modified: Wed, 23 Apr 2025 11:11:13 GMT

Authors:
- jayati.wajire21@pccoepune.org (Primary)
- sujata.kohle@pccoepune.org
- prakash.sontakke@pccoepune.org
- sonali.kale@pccoepune.org
- mrunal.gaikwad21@pccoepune.org
- anway.alamwar22@pccoepune.org
- gayatri.gore22@pccoepune.org
- sneha.loni22@pccoepune.org

Primary Subject Area: Track: Open (CSE/IT/EC/Robotics)

APPENDIX D: PLAGIARISM REPORT

Plagiarismchecker.ai

Plagiarism Scan Report
Date: 23-04-2025

Unique
93%

Plagiarized
7%

Word: 5000

Characters: 26531

Content Checked for Plagiarism

In recent years, air purification has gained special significance due to the increased interest in indoor air quality and its direct relation with human well-being. UV-C radiation, being germicidal, has found extensive application in sterilization technology due to its effectiveness to eradicate airborne germs, viruses, and bacteria. Conventional UV-C air treatment systems do not have smart control and monitoring features, leading to wasteful energy utilization and little feedback about the operation status.

For surmounting the limitations, in this project an ESP32-Based Cloud-Integrated UV Lamp Monitoring and Control System that integrates intelligent control, remote monitoring, and fault alarm has been proposed. By taking advantage of the features of the ESP32 microcontroller, the system combines a relay-controlled UV-C lamp with Wi-Fi capabilities, allowing real-time data to be transmitted to the cloud and user interfaces. The system is remotely controlled by both a web interface and a phone app, so the user can control the lamp from anywhere they have internet.

The project incorporates the ThingSpeak IoT platform to log lamp operation time and status, making data available for visualization and analysis. Furthermore, an in-built safety mechanism detects anomalies—such as failure to switch ON—and alerts users through the dashboard. This integration of embedded systems with cloud computing

APPENDIX E: PROJECT PARTICIPATION CERTIFICATES

PIMPRI CHINCHWAD EDUCATION TRUST'S
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING
Sector No.26, Pradhikaran, Nigdi, Pune - 411044
DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION

ABHIYANTRIX '25
FOR THE STUDENTS BY THE STUDENTS

CERTIFICATE
OF PARTICIPATION

This is to certify that

the paper titled Cloud Integrated Air Purification System
has been presented by Anway Alamwar
under the guidance of Dr. P V Sontakke at Abhiyantrix'25 a Conference for the Students, By the Students organized by the Department of Electronics and Telecommunication Pimpri Chinchwad College Of Engineering , Pune on 21st and 22nd March, 2025.

Arya Wazalvar
Risha Chopra
Genral Chair

Prof. Swapnil Ayane
Dr. Swati Jagtap
Faculty Coordinator

Dr. Kishor Kinage
HoD, Dept of ENTC
PCCOE

PIMPRI CHINCHWAD EDUCATION TRUST'S
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING
Sector No.26, Pradhikaran, Nigdi, Pune - 411044
DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION

ABHIYANTRIX '25
FOR THE STUDENTS BY THE STUDENTS

CERTIFICATE
OF PARTICIPATION

This is to certify that

the paper titled Cloud Integrated Air Purification System
has been presented by Shuha Loni
under the guidance of Dr. P V Sontakke at Abhiyantrix'25 a Conference for the Students, By the Students organized by the Department of Electronics and Telecommunication Pimpri Chinchwad College Of Engineering , Pune on 21st and 22nd March, 2025.

Arya Wazalvar
Risha Chopra
Genral Chair

Prof. Swapnil Ayane
Dr. Swati Jagtap
Faculty Coordinator

Dr. Kishor Kinage
HoD, Dept of ENTC
PCCOE

PIMPRI CHINCHWAD EDUCATION TRUST'S
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING
Sector No.26, Pradhikaran, Nigdi, Pune - 411044
DEPARTMENT OF ELECTRONICS AND TELECOMMUNICATION

ABHIYANTRIX '25
FOR THE STUDENTS BY THE STUDENTS

CERTIFICATE
OF PARTICIPATION

This is to certify that

the paper titled Cloud Integrated Air Purification System
has been presented by Gayatri Gone
under the guidance of Dr. P V Sontakke at Abhiyantrix'25 a Conference for the Students, By the Students organized by the Department of Electronics and Telecommunication Pimpri Chinchwad College Of Engineering , Pune on 21st and 22nd March, 2025.

Arya Wazalvar
Risha Chopra
Genral Chair

Prof. Swapnil Ayane
Dr. Swati Jagtap
Faculty Coordinator

Dr. Kishor Kinage
HoD, Dept of ENTC
PCCOE

APPENDIX F: MOU

PREAMBLE:

Pimpri Chinchwad College of Engineering (PCCoE) is one of the best engineering colleges in Pune, established in 1999, nurtured and managed by Pimpri Chinchwad Education Trust (PCET). PCCoE is functioning proactively to provide the best professional environment to engineering and management students in terms of academics, industry oriented trainings, sports, co-curricular & extracurricular activities, cultural activities, various competitions, etc to create true aesthetically sensitive, socially committed and technologically competent engineers and management professionals. Academics at PCCoE is disciplined with results at the par of best colleges in Savitribai Phule, Pune University (SPPU). Students of PCCoE are proving their talents at state, national and international level competitions for technical and non-technical events. Along with results PCCoE has excellent placements. The institute offers seven undergraduate engineering programmes of four year duration namely: Mechanical Engineering, Electronics & Telecommunication Engineering, Computer Engineering, Information Technology, Civil Engineering, Computer Engineering (AI and ML) and Computer Engineering (Regional Language). Four engineering departments are offering post- graduate programs. The departments of Computer Engineering, Mechanical Engineering, Electronics and Telecommunication Engineering are recognized with Ph. D. Research Center by Savitribai Phule Pune University. The institute also offers a B. Voc. Degree for Solar systems, Industrial Refrigeration, Mechatronics Engineer and IoT Engineer.

The Agreement is signed between,

Whereas, Pimpri Chinchwad College of Engineering, Department of Applied Sciences and Humanities represented, Sector No.26, Pradhikaran, and Nigdi, (Henceforth referred as **Party 1)**

AND

Whereas, ARKLITE SPECIALITY LAMPS PVT LIMITED, Gat No 2794, Kharabwadi Chakan Talegaon Road,Tal Khed, Chakan, Pune, Maharashtra 410501. (Henceforth referred as **Party 2)**

Party 2 is Arklite Speciality Lamps Pvt. Ltd. was established in 1996 and has a manufacturing facility in Pune – Chakan area. The company is sole manufacturer of UV lamps in East Asia with ISO slandered 9001:2008 company

The main product of Arklite is the Ultraviolet (UV) lamps and related systems. Arklite is the only company in the world which is totally integrated; a company having all the facilities needed for UV system manufacturing under one roof, such as:

- Lamp manufacturing
- System designing
- Microbiology lab
- Electronic shop to design and manufacture ballasts, digital control panel, etc
- A state-of-the-art Fab shop, including laser cutting

Arklite is the only company in India manufacturing UV lamps required for disinfection application (low pressure UVC lamps) and the UV curing application (Medium or High pressure lamps UVA and UVB range). Being a fully integrated facility, makes it the most reliable and economical source of UV systems for air and water disinfection and odor control.

Interaction between Technical institutions and industry is the need of an hour. This will have great impact on the Engineering Curriculum, exposure of industrial atmosphere to engineering students and subsequent placement of graduating engineers in industries across the globe. To produce proficient graduates ready for the industry, it is necessary to know the requirements of the industries. We at PCCoE always have built great connection with industries in various sectors through Industry Institute Interaction. PCCoE encourages faculty as well as students to strengthen their individual growth through industry connect.

The purpose of the collaboration with Arklite has two aspects:

1. **Students Perspective**
2. **Faculty Perspective**

The details for both the aspects are elaborated:

1. **Students Perspective**

With the goal of exposure to the updated technology and innovation happening in industries different MoU with well-known professional institutes and industries are signed at PCCoE. They keep students updated with the latest technologies and give them wide space to apply their knowledge and skills to find the solutions to the different problems faced by the industries.

Arkelite can be one of the industries which can provide platform our engineering students from First Year to Final Year Engineering Students (FY B. Tech to Final Year B. Tech).

For 1st year Students

PCCoE Engineering Physics faculty (under AS & H) explore first year students

What Arklite can do for Engineering Physics Students?

- **Industry visit for Physics Students :**
  Arklite can arrange and allow factory visit: It can be done in small groups of say 30
- **Laboratory conduction via industry connect:**
  Sensors are having tremendous applications in various domains like healthcare, energy, IoT, water treatment etc. As a part of Physics syllabus-students will learn in physics theory syllabus and will get real time applications in physics laboratory
  With the help of Arklite, practical sessions can be arranged related to UV water treatment system and sensors at PCCoE (in house):
  a. UV water treatment system can be kept at Department of Applied Science and humanities for a demo of how it functions for a mutually agreed period say 1 week.
  b. Photo diode measurement and calibration with a know source of light and measurement of unknown light source intensity.
  c. UVT (UV Transmittance) measurement setup demo and explain how UVT affects the water treatment quality

For 3rd and 4th year Students

- **Projects:**
  a. Arklite can take one or two students for short term projects (1 or 2 months) in the field of E&TC and Mechanical engineering department.
- **Internship:**
  b. Arklite can provide internship to one or two students and if found suitable the suitable position can be offered.

For Faculty- PCCoE

- Dr. Sonali Kale, Engineering Physics, PCCoE has collaboration with Dr. Mandar Sahasrabudhe, General Manager, Arklite Speciality Lamps Pvt. Ltd. The project is application of Nanoparticles in air purifier.
- Industry tour for faculty
- Arklite can define problem statements related to simulation or making electronic devices and related solution can be provided by the teachers.
- Based on mutual understanding further actions related to those problems can be decided.

Whereas, both parties no. 1 and party no. 2, after due consideration of various aspects have arrived at the following understanding in respect of providing platform of industrial projects through industry connect.

Whereas, both parties have decided to agree to initiate collaborative skill based activities in the areas of mutual interest and in accordance with terms and conditions set forth in this memorandum of understanding (MOU).

OBJECTIVES of MOU:

1. To inculcate understanding of physics concepts as per industry perspectives
2. To enable the students to realize the day-to-day operations they will have to perform
3. To enable the students to upskill and upscale themselves
4. To prepare the learners for coping with real-time challenges

AGREEMENTS OF MOU:

Both Party 1 and Party 2 have decided to agree to initiate collaborative skill based activities in the areas of mutual interest and in accordance with terms and conditions set forth in this memorandum of understanding (MOU).

1. Party 1 & 2 Collaborative activities will evolve a mechanism for exchange of experiences and expertise for mutual benefit of both the organizations.
2. Party 1 – Pimpri Chinchwad College of Engineering, Nigdi, Pune's Department of Applied Sciences and Humanities will provide students and faculty for the successful completion of the activities
3. Party 2 - would provide other timely required support for organizing activities time to time. Arklite Speciality Lamps Pvt. Ltd.
4. Party 1 & 2 Collaborative activities will fully acknowledge the endeavor of both the parties and persons involved therein.
5. Party 1 & 2 Memorandum of understanding may be amended by the mutual consent through an exchange of correspondences between both the parties. We, the above noted parties have signed this Memorandum of Understanding on the date and place mentioned above which shall be binding on our representatives
6. Party 2-may seek assistance/guidance of Party1 faculty members in technical or any troubleshoot issues during conduction.
7. Party 2 employee will be available to participate in activities organized by PCCoE with intimation of 15 days in advance notice.

INTELLECTUAL PROPERTY RIGHTS:

Rights regarding publications, patents, royalty, ownership of software/design/product developed under the scope of MOU shall be decided by two parties by mutual consent. In case of patents, if Arklite R&D Laboratory / funding is involved then the assignee shall be Arklite and Inventors can be from both Arklite and PCCoE. In case, if the product/Software/design is developed then the product manufacturing rights shall be with Arklite and the other terms shall be decided by two parties by mutual consent. In that case the complete process and knowledge shall be transferred to Arklite.

In case if PCCoE wants to manufacture the product, the terms shall be decided by two parties by mutual consent.

CONFIDENTIALITY:

Both the parties agree to hold in confidence all information/data which is obtained from either side or created during the performance of MOU and will not disclose the same to any third party without written consent of the other side.

COORDINATORS:

Both sides will designate persons who will have responsibility for coordination and implementation of this agreement.

DURATION OF MOU:

This MOU will take effect from the date it is signed by the representatives of the parties.

This MOU is binding on both the parties for 3 years -2024-25, 2025-26, 2026-27.

Either party may terminate the MOU by giving 1 month's written advance notice to the other party.

Once terminated neither Party 1 nor Party 2 will be responsible for any losses; financial or otherwise, which the other party may suffer.

This MOU is signed with the subjective approval of representatives of both the parties' academic/administrative bodies.

SCHEDULE FOR ACTIVITIES:

Various activities to achieve the objectives of MOU mentioned, will be conducted as per mutual convenience of both the parties with minimum 15 days of advance planning.

THE PLACE OF SIGNING MOU:

Pimpri Chinchwad Education Trust's Pimpri Chinchwad College of Engineering, Sector No. 26,

Pune – 411 044

On Behalf of Party 1 (Department of Applied Sciences and Humanities, Pimpri Chinchwad College of Engineering)		
Dr. Sonali Kiran Kale Associate Professor, Department of Applied Sciences and Humanities	**Dr. Leena Sharma** Head, Department of Applied Sciences and Humanities	**Dr. G. N. Kulkarni** Director, Pimpri Chinchwad College of Engineering
Date: July 23rd 2024 Place:	Seal	

On Behalf of Party 2 (ARKLITE SPECIALITY LAMPS PVT LIMITED)	
Dr. Mandar Sahasrabudhe, General Manager,	**Mr. Vikram U. Bapat** Managing Director
ARKLITE SPECIALITY LAMPS PVT LIMITED Gat No 2794, Kharabwadi Chakan Talegaon Road, Tal Khed, Chakan, Pune, Maharashtra 410501.	
Date: Place:	Seal