International Journal of Creative and Open Research in Engineering and Management
ISSN: 3108-1754 (Online)
Volume 02 Issue 11 November-2026 | Impact Factor: 3.5



Mushroom Cultivation Using IOT

Mrs.Vidyashree K.V[1], Aishwarya Premanand Naik[2], Kishan B L[3], Vinod Kumar S[4]

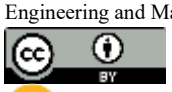
[1] KS School of Engineering and Management, Bengaluru, India
[2] KS School of Engineering and Management, Bengaluru, India
[3] KS School of Engineering and Management, Bengaluru, India
[4] KS School of Engineering and Management, Bengaluru, India
[5] KS School of Engineering and Management, Bengaluru, India

Corresponding Author Email: aishwaryanaik0527@gmail.com

Abstract

How to Cite this Article:
K.V, V., S, A. N., L, K. B. & S, V. K. (2026). Mushroom Cultivation Using IOT. International Journal of Creative and Open Research in Engineering and Management, 02(11), 1-9. https://doi.org/10.55041/ijcope.v2i11.153

License:
This article is published under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author(s) and the source are credited.
© The Author(s). Published by International Journal of Creative and Open Research in Engineering and Management.







https://doi.org/10.55041/ijcope.v2i11.153

Mushroom cultivation is subject to environmental conditions and monitoring to facilitate proper growth and production of the mushrooms. Monitoring temperature, humidity, and health status of the mushrooms manually is time-consuming and could lead to late discovery of unhealthy mushrooms. This research paper proposes an IoT based automated mushroom cultivation monitoring system coupled with deep learning-based image classification system for pink oyster mushrooms (Pleurotus djamor). The system design includes an ESP8266 NodeMCU as the controller and DHT11 sensor to monitor temperature and humidity in the environment of the cultivation chamber. Relay module is employed to control the operations of the ventilator fan, humidifier/misting unit and warm light. Real-time sensor data and status of the relays are sent to the Blynk IoT cloud platform making it possible to monitor the chamber using the mobile dashboard, control actuators to be either auto or manual and get alerts when any environmental parameter crosses its threshold level.

Furthermore, other than the environmental monitoring, the system also makes use of the image-based classification of mushroom health status through the ESP32-CAM. The device takes pictures of pink oyster mushrooms and sends the image data via Wi-Fi connection to the computer application written in Python for analysis. The image data is then analyzed using the EfficientNetB2 CNN that has been created through the transfer learning technique. The classification of the images into different categories (healthy, normal and diseased) is done based on the analysis results obtained. The classification results, confidence scores and the prediction probabilities are displayed on the application interface. The system makes it easy to identify unhealthy mushrooms. The system incorporates all of its functions (automated environmental control, cloud monitoring and deep learning-based image classification) into a unified mushroom cultivation system. It lowers the amount of required manual labor, provides continuous observation of the cultivation conditions and helps to identify unhealthy mushroom growth The proposed system is composed of two main subsystems, namely the environmental monitoring and automatic control, and the image-based mushroom health classification system. For the environmental monitoring and control subsystem, a NodeMCU microcontroller and a DHT11 sensor are employed for constant monitoring of the temperature and humidity levels in the cultivation environment. Relay module controls the operation of the fan, humidifier/misting device, and warm light depending on pre-defined threshold values. Live readings of the sensors and relay statuses are sent to the Blynk IoT cloud platform via Wi-Fi connection. Using the mobile application, one can observe the environment-related parameters, control the equipment, and receive alerts when conditions exceed the threshold values.

Keywords—Mushroom cultivation, Internet of Things (IoT), ESP8266 NodeMCU, ESP32-CAM, EfficientNetB2, Convolutional Neural Network (CNN), Mushroom classification, Mushroom health monitoring, Environmental monitoring, Smart agriculture.



I. INTRODUCTION

Agriculture is one of the most important sectors that are essential for human survival and economic development. It plays an important role in sustaining humans and economic development. Smart agriculture using Internet of Things (IoT) enables automation and control of the environment and reduces human intervention in farming and increases output. Oyster mushrooms are widely used across the globe. There is an increasing demand for oyster mushrooms which results in their large-scale production and it is a great way to recycle agricultural waste and produce nutritious food.

There is a need to maintain some specific environmental parameters during oyster mushrooms' growing period including temperature, humidity, air circulation, and lighting for better yields. There is no ideal temperature for oyster mushrooms since it changes based on different types of strains. Winter strains: 8°C to 12°C, universal strains: 14°C to 17°C, Summer strains: 16°C - 30°C. Therefore, pink oyster mushroom (Pleurotus djamor) is a common edible fungus that is known for its appealing pink color, smooth texture, and high nutrient composition.

The fungus is composed of proteins, fiber, vitamins, minerals, and antioxidants, and therefore qualifies to be a nutritious food item. Owing to its fast growth cycle and rising consumer demand, the fungus has gained prominence in the commercial farming of mushrooms. The other benefit associated with the fungus is that it is eco-friendly due to its use of agro-waste substrates. The proper growing of pink oyster mushrooms requires proper environmental conditions to be provided. The proper temperature should range between 20°C to 30°C, whereas proper humidity needs to range from 80% to 90%. Indirect lighting should also be provided for the growth of pink oyster mushrooms. Agricultural wastes like paddy straw, wheat straw, sugarcane bagasse, and corn cobs are the substrates used because they contain nutrients necessary for the growth of mushrooms. It becomes challenging for farmers if the monitoring task is manual. On the contrary, automation of this task is possible with the help of an appropriate monitoring system in place.

II. LITERATURE REVIEW

A. Mushroom:

The mushrooms are edible fungi which are grown for their nutritional value and commercial purposes. The types of mushrooms that are grown for food production include the oyster mushrooms which belong to the genus Pleurotus. They are extensively grown since they are capable of growing on different types of agricultural substrates and can be grown on a small scale as well as commercially [12], [14]. For the proper growth and production of mushrooms, certain environmental conditions should be met. Prior works have emphasized the need of measuring the environmental conditions for the proper growth and production [1], [5]. The image-processing technology and deep learning can help in the development of smart mushroom cultivation system. These technologies can help in analyzing images of mushrooms to determine the various conditions or categories of mushrooms. In recent years, there has been research work on the automated classification and disease detection of mushrooms using the deep learning models [2], [3]

B. Existing Humidity and Temperature Monitoring System:

Smart Farming involves application of IoT technology to make the process more efficient through environmental monitoring and automation. For mushroom cultivation, temperature and humidity play vital roles in providing the required environmental conditions. A system based on IoT technology involving fuzzy logic and predictive control mechanism for monitoring cultivation conditions was suggested [1]. A system based on ESP8266, environmental sensors, and a Telegram bot for automating the process has been suggested [4]. A monitoring system was designed using Raspberry Pi 4, Arduino UNO, DHT22, Node-RED, and MQTT for remote monitoring of cultivation conditions [5]. Temperature and humidity were controlled using ATmega328P, ESP32, and DHT22 sensors along with fans and water pumps [7]. In other systems, Arduino, XBee communication, and LabVIEW are used with sprinklers and exhaust fans for regulating the environment [8]. Monitoring based on Arduino using DHT22 sensor, solenoid valve, and mister has been used [9]. Remote monitoring based on Arduino and ESP8266 has also been done [10], and sensor-based monitoring using Arduino Uno, soil moisture sensor, and LM35 has been used [11]. There is also a domestic system that uses NodeMCU



and Raspberry Pi 4B for monitoring temperature, humidity, light intensity, and soil moisture [12]. In addition, mobile monitoring based on ESP32, ESP8266 with DHT11 and Blynk, and Raspberry Pi with ESP8266 wireless sensor node and cloud storage for mushroom cultivation has been designed.

C.Existing Image-Based Mushroom Classification and Disease Detection Systems:

Image processing and deep learning methods are employed in agriculture for automation of identification and disease detection. Manual inspection of mushrooms is a time-taking process; therefore, the automated image-based identification method is helpful for detecting diseases in mushrooms. A disease detection system using deep learning approach was proposed employing YOLOv5 to classify Oyster mushrooms as healthy and diseased with 90% accuracy with 94% precision and 93% recall rate using 1,100 images [2]. In another study, YOLOv3-tiny and YOLOv4-tiny were applied on a Raspberry Pi platform for real-time mushroom detection and maturity classification as mature and immature. The YOLOv4-tiny achieved 83.9% accuracy [3]. Another study that has used the combination of IoT for mushroom farming involving environmental monitoring, machine learning, and image processing to classify mushrooms and detect diseases [6]. Furthermore, the use of image capturing alongside environmental monitoring was also used in home-based mushroom monitoring system [12]. These researches indicate the use of deep learning and image processing for automatic identification of mushrooms, conditions, and detection of any diseases.

D. NODEMCU ESP8266:

The NodeMCU ESP8266 module is a Wi-Fi enabled development board commonly used in IoT for environmental monitoring and automation in mushroom growing processes. The board allows connecting sensors, processing of information, and performing wireless communication, which makes it ideal for the development of smart mushroom cultivation systems. An IoT-based environmental control system for mushroom cultivation was developed using ESP8266, environment sensors, and Telegram bot [4]. Another mushroom cultivation environmental control system included Raspberry Pi 4, Arduino UNO, DHT22 sensors, Node-RED, and MQTT communication technology for remote monitoring [5]. A temperature and humidity control system were developed based on ATmega328P, ESP32, DHT22 sensors, fans, and water pumps [7]. Another study used Arduino microcontroller, XBee wireless communication technology, LabVIEW, sprinklers, and exhaust fans for environmental control [8]. Also, an Arduino-based environmental control system that uses DHT22 sensors, solenoid valves, and misting system was developed [9].

E. DHT 11 Temperature and Humidity Sensor:

The DHT11 is a digital sensor for measuring the temperature and relative humidity in the ambient environment. The sensor can interface with microcontrollers like NodeMCU ESP8266 and Arduino to monitor environmental conditions. In mushroom cultivation, temperature and humidity control play an important role in achieving optimum growth. There have been various research works in which DHT22 sensors were used to monitor the environmental conditions and control temperature and humidity [5], [7]. The Oyster mushroom mini-farm monitoring system based on IoT used ESP8266, DHT11, and Blynk app [14]. The DHT11 sensor in the proposed system will be connected to the NodeMCU ESP8266 for continuous temperature and humidity sensing within the mushroom cultivation room. These sensed readings will then be processed by the NodeMCU ESP8266 and used to operate the ventilation fan, humidifier, and warm light using a relay module. Environmental readings will also be remotely monitored using the Blynk software.

F. Relay:

Relay is an electromechanical device that switches on/off high-power electrical devices using low-power signals received from the microcontroller. Using relays, devices like fans, humidifiers, lights, etc., can be turned ON/OFF depending on their operational requirements. The relay module has been widely used in IoT-based automation systems to turn ON and OFF various electrical devices. In the case of mushroom farming, the relay module automates environmental control through activation of various devices based on temperature and humidity. In our proposed system, the relay module is interfaced with NodeMCU ESP8266 which controls the three devices including the ventilation fan, humidifier, and warm light. The NodeMCU evaluates the temperature and humidity data coming from the DHT11 sensor and controls the relay module depending on the threshold value. In the case that the temperature exceeds the threshold value, the ventilation fan turns ON. If the humidity falls below the threshold value, then the humidifier will be turned ON. Also, when the temperature falls



below the threshold value, the warm light will be turned ON. Similar environmental control mechanism has been tested in various mushroom farming projects through various devices like fans, water pump, sprinklers, and misting device [7], [8], [9]. Mini farm monitoring system through IoT has also been made with ESP8266, DHT11, and Blynk [14].

G. ESP32-CAM Module:

ESP32-CAM is a small-sized camera development board that consists of an ESP32 microcontroller, camera module, and Wi-Fi connectivity. The device is utilized in IoT and computer vision projects for taking pictures and wireless image transmission. The onboard camera helps capture the images, and Wi-Fi connectivity helps transmit these images to the outside processing device. In mushroom growing, image acquisition could help identify and classify mushrooms according to their appearance. The ESP32-CAM module in the proposed scheme acquires the pictures of Oyster mushrooms in the cultivation chamber. The images are then sent to the computer via Wi-Fi with HTTP communication protocol. The computer analyzes the obtained images by utilizing Python programming language and the EfficientNetB2 deep-learning model, in order to categorize the mushrooms as healthy or unhealthy. This work combines image acquisition and classification using deep learning in order to facilitate the automated process of mushroom health assessment. Prior works include image-based detection of mushroom diseases [2], mushroom detection and maturity classification with YOLO-Tiny [3], and mushroom classification and disease detection using image processing techniques [6].

H. Deep Learning-Based Classification:

Deep Learning is a form of artificial intelligence that is used to analyze images and detect patterns. When it comes to mushroom cultivation, mushroom detection through classification can assist in the identification of whether a mushroom is healthy or not. Previous research has utilized deep learning methods in developing automated Oyster mushroom disease detection systems [2]. Moreover, real-time mushroom detection and classification of their maturity level with YOLO-Tiny model have been investigated [3]. Besides, image processing techniques have been implemented in mushroom detection and disease identification [6]. In the designed system, the EfficientNetB2 neural network model is employed for analyzing the Oyster mushroom pictures obtained from the ESP32-CAM. The images are then sent to the computer and preprocessed using the Python programming language prior to being classified. The trained model will identify the visual features contained in the images and classify the mushrooms as either Healthy or Unhealthy. The output is displayed on a desktop application.

I. Blynk Application for Remote Monitoring and Control:

Blynk is a platform based on the Internet of Things (IoT), where a user can monitor and manage his connected devices via a mobile app. The platform provides users the functionality of viewing sensor data and managing electrical appliances remotely using the internet. Remote monitoring of mushroom cultivation can help users monitor environmental parameters like temperature and humidity without the need to physically visit the chamber regularly. An IoT-based Oyster mushroom mini-farm monitoring system already exists and uses ESP8266, DHT11 sensors, and Blynk app for environmental monitoring [14]. In the proposed system, Blynk will be used alongside the NodeMCU ESP8266 to show real-time data concerning temperature and humidity collected via the DHT11 sensor. The application will also allow controlling the functions of the ventilation fan, humidifier, and warm light by using the relay module. Users will have the ability to monitor the environment and control the devices remotely using a smartphone. The combination of NodeMCU, DHT11, relay-controlled devices, and Blynk offers an easy way for remote monitoring and automation in Oyster mushroom farming.

III. RESEARCH GAPS

Most of the current mushroom growing systems are based on monitoring of the environment parameters, such as temperature and humidity, and controlling devices based on some predetermined conditions. The approach with the system's functioning through unmanned condition checking is not widely used. Inspections using human powers can take a considerable time and require continuous attention from the owner. An additional disadvantage is that some automatic systems do not allow flexible control over the device by the user. Thus, the hybrid approach, when both manual and automatic modes are available, gives the cultivator an opportunity to use the system depending on his/her cultivation needs. As a result, the development of an integrated system that combines environment monitoring, automation and machine vision for the checkup of the mushroom conformity needs to be introduced. In the presented work, it is explained how an integrated mechanism that uses ESP8266, DHT11, ESP32-



CAM, EfficientNetB2, and Blynk automation has been developed to eliminate those drawbacks.

IV.PROPOSED SYSTEM

A. SYSTEM ARCHITECTURE

The system proposed here consists of two components: an environmental monitoring and control unit and an AI mushroom classification unit. The environmental unit utilizes an ESP8266 NodeMCU coupled with a DHT11 sensor to keep track of the temperature and humidity readings. Depending on the readings, a relay module controls the fan and humidifier in addition to warm light. Additionally, the system also allows automatic and manual control through the Blynk application.

The second component is the AI unit which makes use of an ESP32-CAM for the purpose of taking images of mushrooms. The images are sent over to the computer via Wi-Fi where they are then processed on a deep learning model based on EfficientNetB2. This model classifies the state of the mushroom into three categories: healthy, normal and diseased. Moreover, the result is presented to the user together with the corresponding degree of confidence as an output from the desktop application.

Hardware Components

a. **ESP8266 NodeMCU** – Main controller for environmental monitoring and device control.

b. **DHT11 Sensor** – Measures temperature and humidity.

c. **Relay Module** – Controls the connected electrical devices.

d. **Fan** – Used for ventilation.

e. **Humidifier** – Maintains the required humidity level.

f. **Warm Light** – Provides the required lighting/heat.

g. **ESP32-CAM** – Captures images of the mushrooms.

h. **Power Supply** – Provides power to the system components.

Software Components

a. **Arduino IDE** – Used for programming the ESP8266 and ESP32-CAM.

b. **Blynk Application** – Provides Auto/Manual control of the connected devices.

c. **Python** – Used for the AI-based image classification application.

d. **EfficientNetB2** – Deep learning model used to classify mushrooms as **Healthy, Normal, or Diseased**.

e. **Desktop Application** – Displays the captured image, classification result, and confidence value.

B. BLOCK DIAGRAM

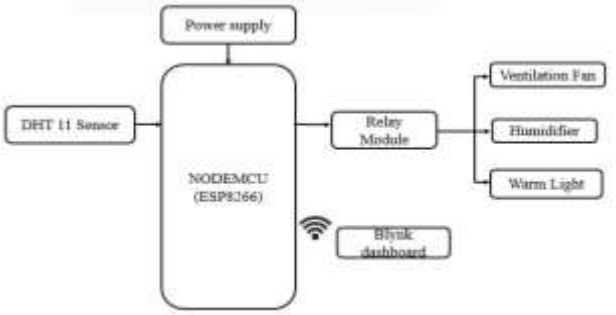


Fig.1: Block Diagram Of The System Monitoring Environmental Conditions

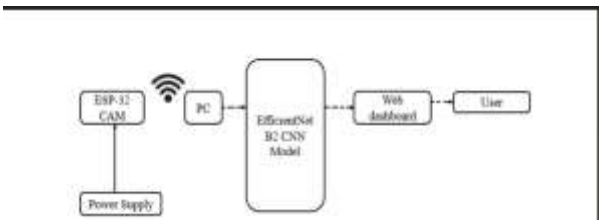


Fig 2: Block Diagram of Mushroom Classification

Fig. 1 illustrates the block diagram of the designed environmental monitoring and control system for mushroom cultivation. The system consists of a DHT11 sensor, ESP8266 NodeMCU, relay module, ventilation fan, humidifier, warm light, and Blynk application. The DHT11 sensor monitors the temperature and humidity of the cultivation environment, while the ESP8266 NodeMCU processes the sensor readings and controls the connected devices through the relay module. The Blynk application provides monitoring and control of the system, including automatic and manual operation of the connected devices. The functions of each component are discussed below.

a. **Power Supply:** The power supply module offers electrical energy to the components of the mushroom production system. The ESP8266 NodeMCU, DHT11 sensor, relay module, fan, moisture device, and warm light unit are powered by the power supply module. If the power stays constant, the monitoring and environmental control process will operate without interruption.

b. **DHT11 Sensor:** The purpose of the DHT11 sensor is to monitor temperature and humidity in the mushroom growing chamber. The readings of the sensor keep changing and sent to the ESP8266 NodeMCU for interpretation. Depending on the



conditions sensed, the NodeMCU is able to control the devices connected through the relay module.

c. **NodeMCU ESP8266:** The NodeMCU ESP8266 serves as the brain of the system and performs the basic function of receiving data concerning temperature and humidity from the DHT11 and processing the obtained information subsequently. It ensures that the relay modules are driven and that the fans, humidifiers, and heating light are operated in accordance with the control conditions embedded in the software should be introduced.

d. **Relay Module:** The relay module serves as the communication link between the ESP8266 NodeMCU and the electrical appliances in the growing room. It controls the ON/OFF switching of the ventilator fan, humidifier, warm light, etc. as per the command given to it from NodeMCU. Thus, the low power controller can control electrical loads safely.

e. **Ventilation Fan:** The ventilation fan is utilized to allow movement of air inside the chamber used for growing mushrooms. It is regulated by the relay module as per the needs of the environment and regulated system control. Through the use of the fans, conditions suitable for mushroom growth can be maintained.

f. **Humidifier:** The purpose of the humidifier is to help maintain the humidity levels inside the mushroom growing chamber. Every time the humidity level falls below the set value, the ESP8266 NodeMCU turns on humidifier with the help from relay module.

g. **Warm Light:** Warm light is necessary for maintaining temperature regulation in the mushroom growth chamber. If the temperature drops below the desired level, the warm light will be activated by the ESP8266 NodeMCU through the relay module, which will make it possible to maintain the necessary conditions in the environment.

h. **Blynk Application:** Blynk Application offers a user interface that can be used for monitoring and controlling the environmental control system. The application allows the user to monitor the sensor values and to control the devices connected to the system. The Blynk system supports both automatic and manual modes of operation, thus providing versatility for the user in controlling the cultivation environment.

Figure 2 illustrates the block diagram of mushroom classification system. The system uses the ESP32-CAM to take images of the mushrooms and sends them by Wi-Fi to a PC. Captured images are buffered into a neural network template based on EfficientNet-B2 to classify the mushrooms into two categories: healthy and unhealthy. The healthy category refers to mushrooms exhibiting full health, while unhealthy indicates mushrooms that are showing signs of an unhealthy condition. The classification result and the confidence value will then be represented on the desktop application, allowing the user to monitor the mushroom condition without manually inspecting each mushroom continually. Here, the operation of each element is described below:

a. **Power Supply:** The power supply unit avails the necessary power supply for the operation of the ESP32-CAM along with other components in the mushroom classification system. With a stable power supply, the ESP32-CAM can capture mushroom pictures and relay them through Wi-Fi for image classification.

b. **ESP32-CAM:** The ESP32-CAM captures images of mushrooms inside the cultivation chamber. The device contains an inbuilt camera and microcontroller having Wi-Fi connectivity which is important for image capture in the system being proposed. After capturing the images, they are sent to the computer for further processing and classification.

c. **Wi-Fi:** The ESP32-CAM uses Wi-Fi to transmit the mushroom pictures to the computer. The transmission of the images occurs through an HTTP connection, and those images are received and processed in the desktop application for classification.

d. **PC and Desktop Application:** The computer processes these images taken with the ESP32-CAM. The desktop program accepts those images. It then does the necessary image preprocessing of these images and feeds them to the EfficientNetB2 CNN model. After that, the application shows the classification result along with the value of confidence.

e. **EfficientNetB2 CNN Model:** The EfficientNetB2 model is utilized to classify mushrooms based on images. Initially, the image is resized and preprocessed prior to its use as input to the EfficientNetB2 model. The model takes the input image and extracts important features from the image through its convolutional layers. Based on the probability of the predictions made by the model, mushrooms are classified into either Healthy or Unhealthy category. The outputs and the confidence values are displayed on the desktop application.

f. **EfficientNetB2 Classification Operation**: The mushroom image taken is sent over Wi-Fi from the ESP32-CAM to the personal computer. Desktop



application pre-processes the image, converting and resizing it to the desired format. Processed image is then fed into the EfficientNetB2 model for classifying it. The model observes the cognitive features of the mushroom in the image and delivers the probability for the different classes. The class with the highest probability is then chosen as the outcome.

g. **User:** The user monitors the mushroom classification results via the desktop application. The application will show the predicted state of the mushroom as Healthy or Unhealthy alongside the confidence level. Thus, the user knows the condition of the mushroom with reference to the prediction made by the AI model.

C.FLOWCHART

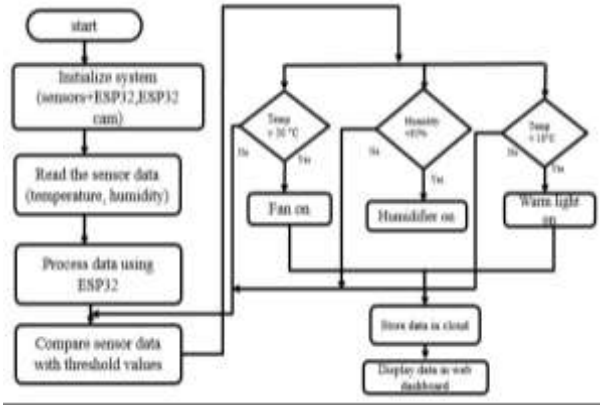


Fig.3: Flowchart of System Monitoring Environmental Control

- **Start:** The initiation of the environmental control takes place when the mushroom-growing system is switched ON. All the equipment necessary for control and monitoring becomes ready of the environment inside the growth chamber, hence ensuring that everything is set for optimum growing conditions.
- **System Initialization:** The ESP8266 NodeMCU initializes the DHT11 sensor and establishes the required connections for environmental monitoring and control.
- Sensor data reading: The DHT11 sensor takes temperature and humidity readings inside the mushroom cultivation chamber on a continuous basis and sends these to the ESP8266 NodeMCU.
- **Data processing:** The ESP8266 NodeMCU reads the temperature and humidity from the DHT11 sensor and formats them for comparison with the threshold values set.
- **Comparison with threshold values:** The measured temperature and humidity values are compared with the predefined threshold conditions to check whether the connected devices need to be turned ON or OFF.
- **Temperature Control:** When temperature exceeds 30 degree centigrade, the fan will switch ON and cool down the extra heat. when the temperature goes back into the normal range, then the fan is turned OFF.
- **Humidity Control:** If the humidity level decreases from 85%, the humidification mechanism will turn ON so that there is adequate moisture for the mushrooms to grow. On attaining the desired humidity levels, the humidification mechanism turns OFF automatically.
- **Warm Light Control:** Warm light should turn ON where temperature does not rise to 18 degrees to create a proper environment. When the required temperature is attained, the warm light will turn OFF.
- **Continuous Process:** The system will continuously cycle through taking the readings and comparing them to the set threshold values. This will then control the fan, humidifier and warm light to keep the environment at a suitable level.

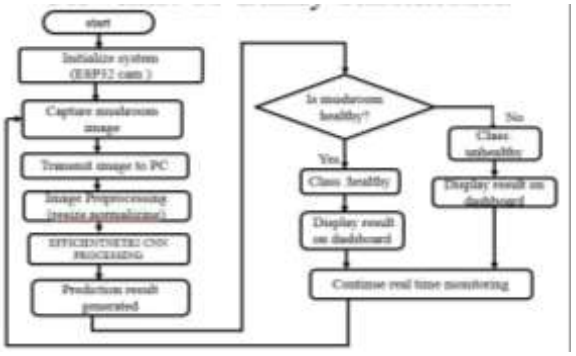


Fig. 4: Flowchart Of Binary Classification

- **System Initialization:** Binary classification starts with the initialization of the ESP32-CAM. This ensures that the image capture module communicates effectively with the image-processing system.
- **Image Capturing:** The camera takes a snapshot of the mushroom crops
- **Image Transmission:** This image is transmitted from the ESP32-CAM to the PC through the Wi-Fi network with the help of the HTTP connection.
- **Image Preprocessing:** The image is then preprocessed to be compatible with the requirements of the EfficientNetB2.
- **EfficientNetB2 Classification:** This processed image is then passed as an input into the EfficientNetB2 CNN model. The model goes on to analyze the visual features of the mushroom image and determines its status.
- **Healthy Classification:** In the case that the model determines the mushroom image to be Healthy, the



output with the confidence level will be displayed in the desktop application.

- **Unhealthy Classification:** In the case that the model determines the mushroom image to be Diseased, the output with the confidence level will be displayed in the desktop application.
- **Continuous Observation:** The process is continuous in capturing the mushroom image and observing its status through the desktop application.

V.IMPLEMENTATION

Hardware Implementation

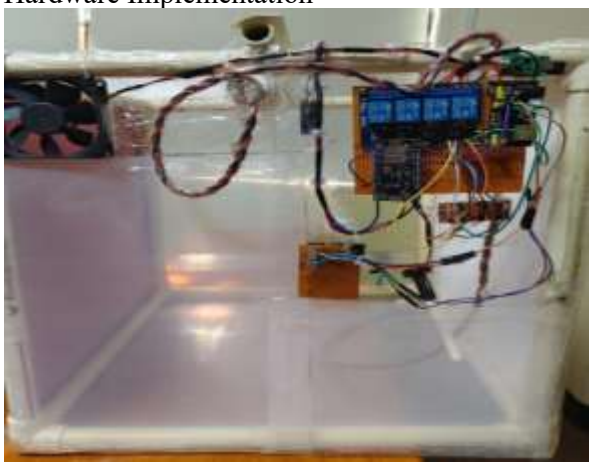


Fig. 5: Photograph of the complete mushroom cultivation setup

Figure 5 shows the fully developed prototype for mushroom cultivation system. In the figure above, all the necessary environmental monitoring and control equipment needed for mushroom cultivation have been incorporated. The ESP8266 NodeMCU is connected to the DHT11 sensor for temperature and humidity monitoring, whereas the relay module is used for controlling the fan, humidifier, and the warm light. The ESP32-CAM is used for taking pictures of mushrooms for classification by the AI-based classifier.

Fig. 6: ESP8266 NodeMCU with DHT11 and relay module

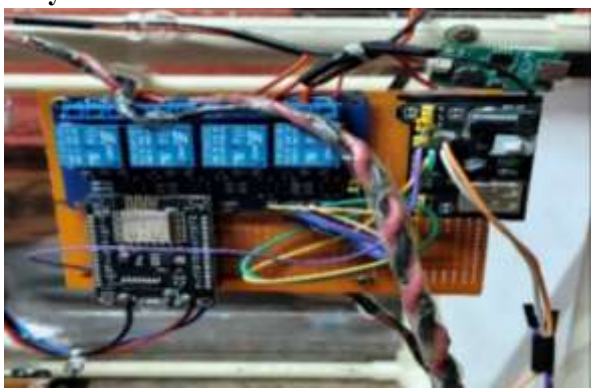


Fig. 4 illustrates the ESP8266 NodeMCU connected to the DHT11 sensor and relay module. The DHT11 sensor measures the temperature and humidity of the mushroom cultivation environment and sends the readings to the ESP8266 NodeMCU. The NodeMCU processes the sensor data and controls the connected actuators through the relay module. The relay module provides switching control for the ventilation fan, humidifier, and warm light based on the programmed environmental conditions.



Fig. 7: Fan, humidifier, and warm light connected to the relay

Figure 7 shows the connection between the ventilation fan, humidifier, and warm light to the relay module. The relay module acts as a controller for the three actuators depending on the control signals sent by the ESP8266 NodeMCU. The ventilation fan is used for air circulation, humidifier controls the humidity levels, while the warm light aids in the temperature regulation within the mushroom growing chamber.

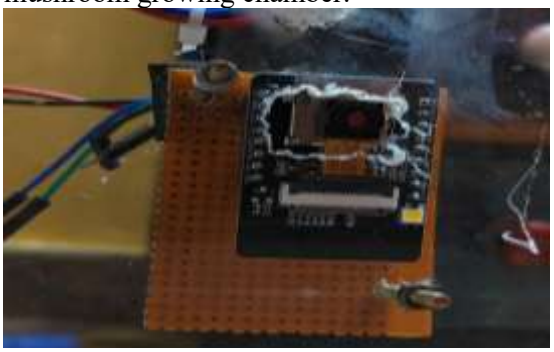


Fig. 8: ESP32-CAM installed inside/near the cultivation chamber

Figure 8 shows the ESP32-CAM camera that takes the image of the mushrooms. The camera is set in such a way that it captures images of the mushrooms in the cultivation box. These images are then sent via Wi-Fi to the computer, where they are classified using the classification algorithm that uses the EfficientNetB2 model.

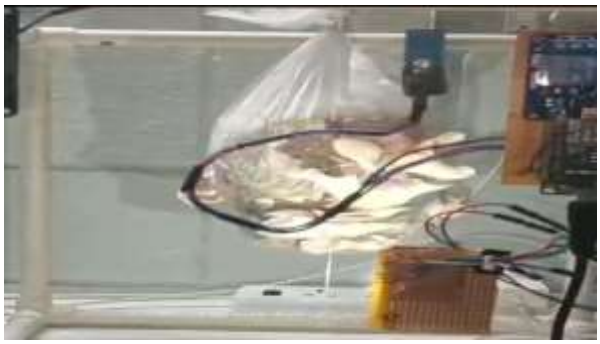


Fig. 9: ESP32-CAM capturing mushroom images

Figure 9 shows the ESP32-CAM device taking images of the mushrooms in the growing chamber. This camera captures images of the mushrooms and sends them via Wi-Fi to the PC. These captured images are fed into the classification model that uses EfficientNetB2 architecture for the prediction of whether the mushrooms are healthy or unhealthy.

Blynk Cloud Dashboard Design

The mobile dashboard is configured in the Blynk IoT platform under the template "Mushroomcultivation". The datastreams and widgets are mapped as in Table 4.3.

Table 1: Blynk virtual-pin design of the dashboard

Virtual pin	Widget	Direction	Content
V0	Label	controller → app	Temperature string, e.g. "26.5 C"
V1	Label	controller → app	Humidity string, e.g. "78 %"
V2	Switch	app → controller	Relay 1 manual command (0/1); actual state written back
V3	Switch	app → controller	Relay 2 manual command; actual state written back
V4	Switch	app → controller	Relay 3 manual command; actual state written back
V5	Switch	app → controller	Relay 1 mode: 0 = AUTO, 1 = MANUAL
V6	Switch	app → controller	Relay 2 mode: 0 = AUTO, 1 = MANUAL
V7	Switch	app → controller	Relay 3 mode: 0 = AUTO, 1 = MANUAL
Event "temp"	Notification	controller → app	Zone-change and high-humidity alerts

Two design details are worth noting. First, the app switches are command/request channels only: the controller holds the authoritative relay state and writes it back to V2–V4 after every control cycle, so the switches always display what the loads are actually doing, not what the user last asked. Second, on every successful cloud connection the firmware calls Blynk.syncVirtual() on V2–V7, which resynchronises modes and commands after a reboot or an internet outage — the chamber resumes exactly the operating mode the grower left it in.

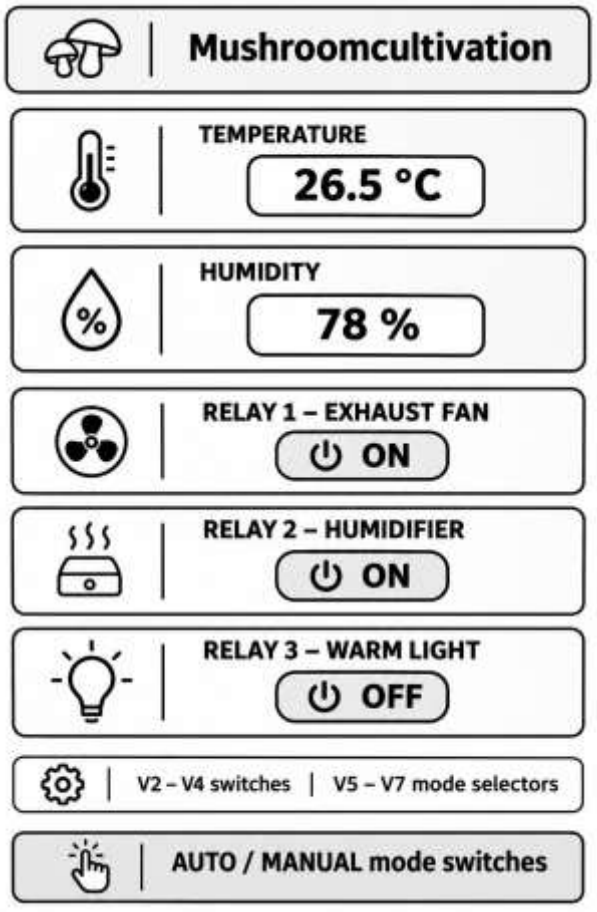


Fig 10: Blynk mobile dashboard design of the project



Deep-Learning Model Design

1. Class Definition

The classifier distinguishes three quality classes. bad corresponds to mushrooms that are diseased or spoiled (visualised as DISEASED in the application, red); good corresponds to healthy mushrooms (HEALTHY, green); normal corresponds to average-quality mushrooms that are neither exceptional nor defective (NORMAL, amber). The three-level scale was chosen over a binary healthy/diseased decision because a chamber always contains mushrooms at intermediate stages, and forcing them into extreme classes would produce noisy alerts.

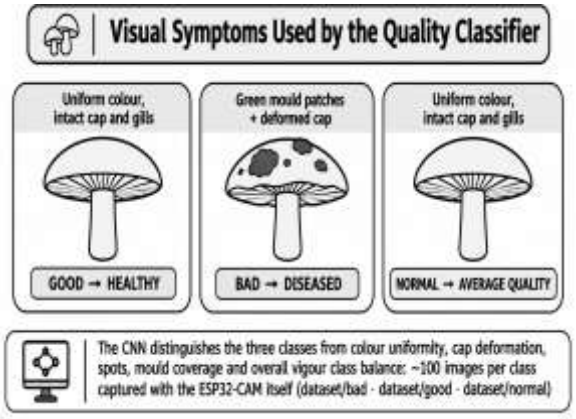


Fig. 11: Visual symptoms that separate the three quality classes

2. Architecture: EfficientNetB2 with a Custom Head

The backbone is EfficientNetB2, pretrained on ImageNet, with its classification top removed. EfficientNetB2 was selected for three reasons: its compound-scaling design gives it a better accuracy-to-computation ratio than ResNet-family baselines at small-to-medium scale; its 260 × 260 native input resolution matches the practical snapshot size captured by the ESP32-CAM; and its ImageNet weights transfer well to small custom datasets, which is exactly the regime of this project (≈300 images).

On top of the frozen backbone, a compact classification head was designed:

- GlobalAveragePooling2D — collapses the1408-channel final feature map of the backbone into a single 1408-dimensional feature vector, drastically reducing the parameter count that follows;
- Dropout (rate 0.35) — regularization against overfitting on the small dataset; Dense 128 with ReLU — learnable nonlinear combination of the transferred features;
- Dropout (rate 0.25) — second regularisation stage before the output;
- Dense 3 with softmax — the three class probabilities.

The full architecture and training strategy are illustrated in Fig. 4.8. The input resolution is 260 × 260 × 3, and EfficientNet's own preprocessing is applied before the backbone.

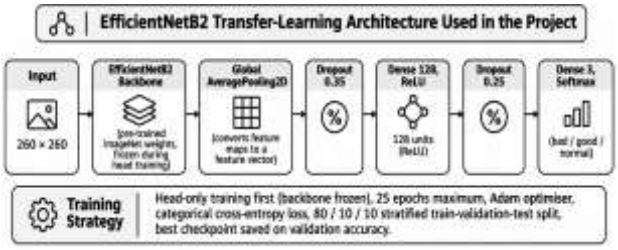


Fig. 12: EfficientNetB2 transfer-learning architecture and training strategy of the project

3. Data Design

The dataset is captured with the project's own camera (capture.py) into three label folders — dataset/bad, dataset/good, dataset/normal — and uploaded to Google-Drive :/MyDrive/musroom_dataset/{bad,good,normal}. Approximately 300 images were collected with near-equal class counts. The split is 80 % training, 10 % validation and 10 % testing, applied with stratification so that every partition preserves the class balance; stratification is essential on a small dataset to prevent a skewed test set. The composition is summarised in Fig. 4.9.

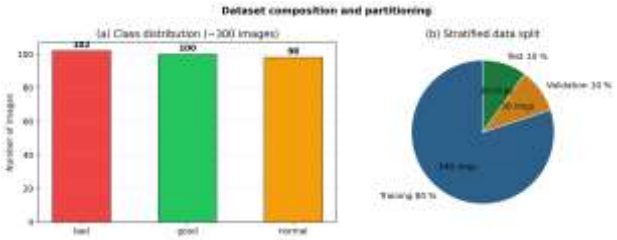


Fig. 13: Dataset composition and stratified 80/10/10 partition

4. Training Design

Training is performed in Google Colab on a T4 GPU with the Adam optimizer and categorical cross-entropy loss. Because the backbone is frozen, only the head learns initially, which both speeds up convergence and protects the transferred features. Three callbacks shape the training run: Model Checkpoint saves the best weights by validation accuracy, Early Stopping halts when validation accuracy ceases to improve, and



ReduceLROnPlateau lowers the learning rate when progress flattens. Training is allowed up to 25 epochs; in practice convergence occurred well before the limit. This checkpoint-based design guarantees that the deployed. keras file is the best-performing snapshot of the run, not merely the last one.

5. Desktop Application Design

5.1 Interface Design

The inspection application is a Tkinter window of 1250 × 780 pixels with a dark modern theme. The layout is divided as follows (Fig. 4.10): a header with the application title; a large left panel for the live camera feed; a right column containing the CURRENT STATUS card (class name in red/amber/green plus confidence), the CLASS PROBABILITIES panel with three animated bars, and a SYSTEM panel showing the camera connection state and live FPS; and two action buttons, STOP CAMERA and EXIT APPLICATION. The colour code follows dashboard conventions — red for disease, amber for normal, green for healthy — so a grower can interpret the screen from across the room.



Fig. 14: Design of the Mushroom AI Inspector desktop dashboard

5.2 Concurrency Design

The application separates acquisition from presentation. A daemon thread runs the camera loop: HTTP GET to the camera's /capture URL (5-second timeout), JPEG decode with OpenCV, and inference with the model. The Tkinter main thread owns all widgets; results are pushed to it from the worker thread, which prevents the UI from freezing during network waits or CPU-bound inference. The camera IP address lives in a single shared configuration module (camera_config.py) used by both the inspection app and the capture tool, so re-deploying the camera is a one-line change.

5.3 Error-Handling Design

The loop distinguishes three failure classes and reacts differently to each: a non-200 HTTP status marks the camera as ERROR and retries after 0.5 s; a connection exception marks it OFFLINE and retries after 1 s; a frame that fails to decode is skipped silently. In every case the application keeps running — an unstable Wi-Fi link degrades the frame rate but never crashes the inspection session. The complete runtime flow is shown in Fig. 11.

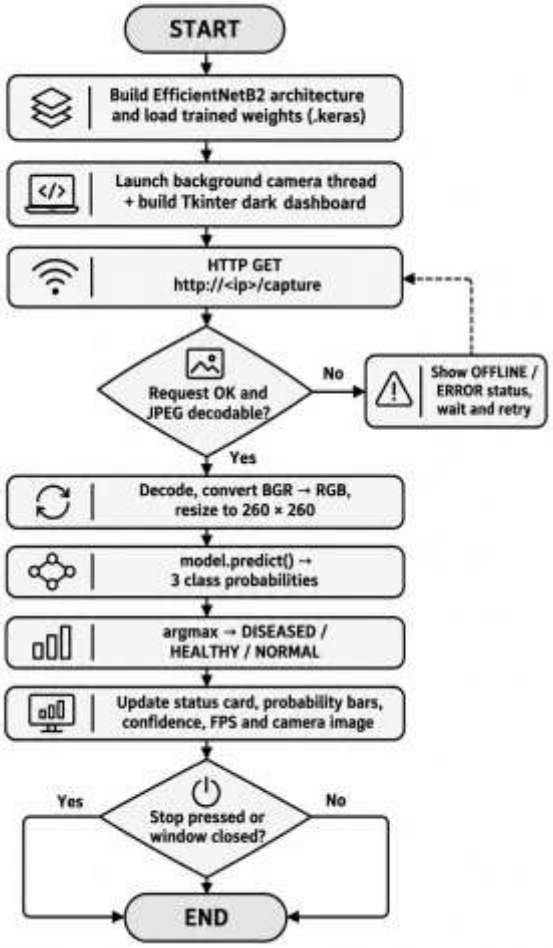


Fig. 15: Inference loop of the desktop inspection application

5.4 Operational Scenario Walkthrough

To make the design concrete, a representative 24-hour operating cycle of a summer day illustrates how the designed elements cooperate:

1. Early morning (05:00, 22 °C, 65 % RH). The reading lies below LOW_TEMP, so the automatic policy energises Relay 3 (warm light). The zone change (Zone 0 → Zone −1) fires one low-temperature notification to the grower's phone. Because the relay states are published after every control cycle, the dashboard shows the warm light ON.

2. Mid-morning (10:00, 26 °C, 58 % RH). The reading is inside the comfort band; all automatic outputs are OFF. A re-entry notification confirms that conditions returned to 24–30 °C. The grower opens the Blynk app, sees the chamber normal, and briefly sets Relay 2 (humidifier) to MANUAL-ON



to top up moisture, then restores AUTO.
3. Afternoon peak (14:30, 31 °C, 72 % RH). Both automatic predicates fire (T > 30 °C and the combined warm-humid rule); Relays 1 and 2 energise, the fan vents hot air and the misting unit assists by evaporative cooling. A high-temperature alert reaches the grower who is away from the chamber.
4. Sensor fault simulation (15:00). The DHT11 wire is accidentally loose; reads return NaN. The firmware marks the sensor invalid, forces all automatic outputs OFF, and the serial monitor prints the failure. Manual switching remains available, so the grower can still run the fan by hand until the sensor is reseated.
5. Evening inspection (18:00). On the PC, the Mushroom AI Inspector shows the camera feed with the current batch classified HEALTHY at high confidence. One bag previously flagged DISEASED was removed by the grower after the morning alert; the same frame now classifies the remaining mushrooms as NORMAL/HEALTHY.
6. Night (23:00, 25 °C, 78 % RH). The combined-stress rule energises the ventilation group again; the grower receives the high-humidity edge-triggered alert exactly The scenario demonstrates the intended division of labour:

6. Design Traceability

Each major design decision traces back to a requirement or a survey finding:

Table 2: Traceability from requirements and survey findings to design decisions

Design decision	Source
ESP8266 as main controller	Cost gap (Ch. 2); Table 3.3 comparison; NFR1
Per-relay AUTO/MANUAL switches	Control-flexibility gap (Ch. 2); FR3
Active-LOW fail-safe boot	Reliability requirement NFR3; relay module design (Ch. 3)
Edge-triggered Blynk events	Notification usability (NFR2); FR5
EfficientNetB2 + frozen backbone	Small-dataset regime (Ch. 2); FR8; NFR5
260 × 260 input and snapshot interface	Camera node design; FR7; NFR5
Tkinter dashboard with colour codes	FR9, FR10; NFR2
Single shared camera configuration module	NFR4, NFR6

7. Binary classification

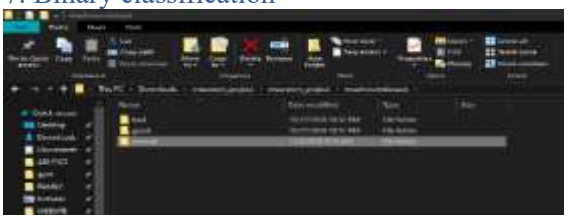


Fig. 16: Dataset/training setup

Figure 16 depicts the results of the mushroom classification model training using the EfficientNetB2 model. It shows how the model performs while training and validation and is helpful in analyzing the learning behavior of the model. The results of training the model help analyze how effectively the model can classify mushroom images as Healthy and Unhealthy.

8. Model Training Results

8.1 Training Convergence:

The EfficientNetB2 classifier was trained for 18 epochs (well within the 25-epoch budget) before the early-stopping criterion ended the run. Validation accuracy rose steeply during the first epochs and plateaued near 100 %, while the losses fell monotonically; the ReduceLROnPlateau callback stepped the learning rate down as the curves flattened. The checkpoint mechanism ensured that the deployed weights correspond to the best validation epoch, not the last.

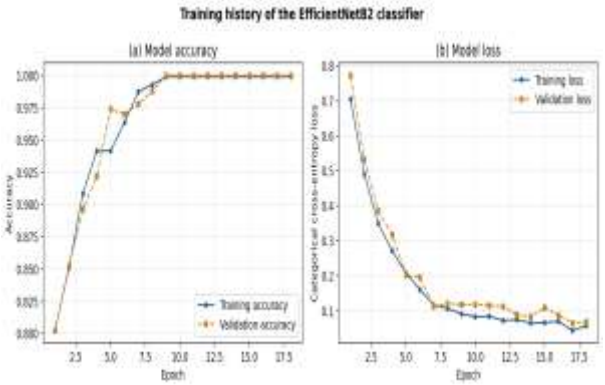


Fig.17: Training and validation history of the EfficientNetB2 classifier: (a) accuracy, (b) loss

8.2 Test-Set Evaluation:

The best checkpoint was evaluated on the held-out stratified test set of 30 images (10 per class), which the network had never seen in any form. The model classified all 30 images correctly, achieving 100 % test accuracy, with every test image of each class predicted as its own class.

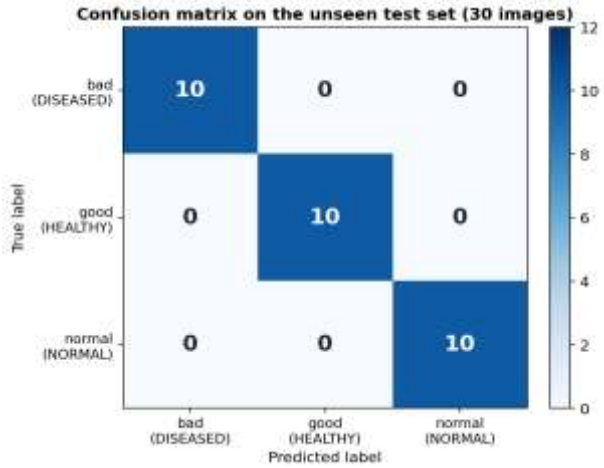


Fig. 18: Confusion matrix of the EfficientNetB2 model on the unseen test set (30 images)

Table 3: Classification report on the test set

Class	Precision	Recall	F1-score	Support
bad (DISEASED)	1.00	1.00	1.00	10
good (HEALTHY)	1.00	1.00	1.00	10
normal (NORMAL)	1.00	1.00	1.00	10
Accuracy			1.00	30
Macro avg	1.00	1.00	1.00	30
Weighted avg	1.00	1.00	1.00	30

A single-image demonstration also confirmed the expected workflow end-to-end: the test image was preprocessed exactly as a live frame would be, and the model returned the correct class with its confidence score.

8.3 Interpretation of the Perfect Test Score

A perfect score on a 30-image test set must be read with statistical care: with ten images per class, even one misclassification would reduce accuracy to 96.7 %, so the observed result indicates that the decision boundary learned from the training images separated the captured classes cleanly. The small dataset and the single-camera capture domain both help the model (the deployment and training distributions coincide), but the same factors caution against expecting the score to transfer unchanged to different lighting, backgrounds or strains. This motivates the periodic retraining workflow already integrated into the project (capture â†' upload â†' Colab â†' redeploy).

8.4 Integration Testing of the Inspection Subsystem

The desktop application was exercised against the live ESP32-CAM. Startup behaviour, steady-state behaviour and failure behaviour were all verified:

Table 4 : Inspection application integration tests

Test	Scenario	Expected result	Observed
I1	Start app with camera online	Window opens, feed live, model loads ("Model loaded successfully")	Pass
I2	Present a healthy mushroom	Status HEALTHY (green) with dominant probability bar	Pass
I3	Present a diseased mushroom	Status DISEASED (red) with dominant probability bar	Pass
I4	Present an average mushroom	Status NORMAL (amber)	Pass
I5	Power off the camera	Status changes to OFFLINE, app keeps running	Pass
I6	Camera returns non-200 response	Status changes to ERROR, retries	Pass
I7	Stop camera button	Worker loop stops, button disables	Pass
I8	Exit application / close window	Clean exit, no orphan processes	Pass

8.5 Live Performance

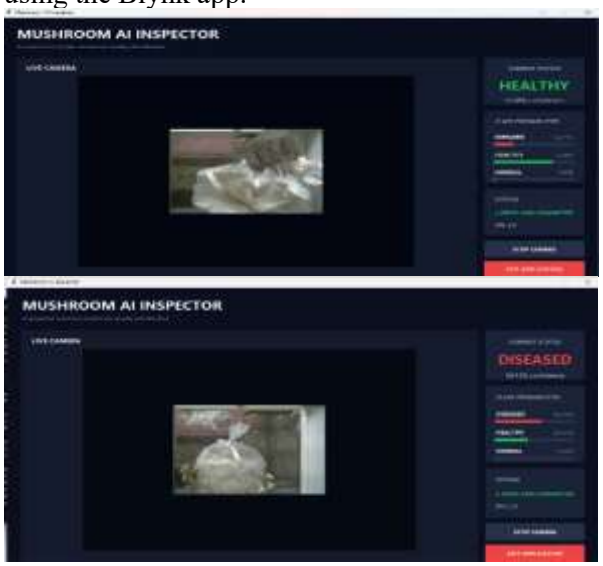
On the test PC (Python 3.10, TensorFlow CPU) the pipeline sustained approximately 1â€“2 frames per second end-to-end, dominated by CPU inference of the EfficientNetB2 backbone on 260 Ã— 260 inputs plus the HTTP snapshot round-trip. This rate is appropriate for the application: mushroom disease develops over hours and days, and the grower watches the dashboard interactively, so multi-Hz video is unnecessary. The FPS indicator allowed the operator to quantify the link quality at any moment.

VII. RESULTS AND DISCUSSION



A. Environmental Monitoring and Control Results

The system that was designed was tested for the purposes of monitoring and controlling the environment within the mushroom growth chamber. Temperature and humidity sensors provided accurate readings which were then sent to the ESP8266 NodeMCU. In accordance with pre-defined threshold values, the NodeMCU was used for controlling the operations of the fan, humidifier, and the warm light through the relay module. Monitoring and controlling of the environmental parameters were also carried out using the Blynk app.



B. Mushroom Classification Results

This classification framework for the mushrooms was validated using the images acquired by the ESP32-CAM. These acquired images were then sent to the computer and analyzed using the trained EfficientNetB2 model. This model categorized the images of the mushrooms into two classes, which are Healthy and Unhealthy, and showed their respective classes with the level of confidence using the developed desktop application.

VIII. CONCLUSION

The Proposed system is aimed at the integration of smart mushroom cultivation and includes environmental monitoring, automatic control and artificial intelligence mushroom classification. ESP8266 NodeMCU along with the DHT11 sensor constantly monitors the temperature and humidity values and controls the work of ventilation fan, humidifier and warm light on the basis of the predetermined conditions. Monitoring and manual control of the mentioned devices are performed using the Blynk application.

Additionally, the ESP32-CAM makes pictures of mushrooms, which are transmitted via Wi-Fi to the PC, where the pictures are processed. The mushroom condition is classified into two categories: Healthy and Unhealthy with the help of the EfficientNetB2-based CNN model. The classification result and confidence value are shown through the desktop application. Overall, proposed system allows avoiding constant manual monitoring and presents the integrated approach to the maintenance of the cultivation environment and mushroom condition.

IX. FUTURE SCOPE

- Deployment of Edge AI: The trained deep learning model can be further optimized for deployment on the edge, thus decreasing reliance on the PC and increasing speed of image classification locally.
- Other Environmental Sensors: Other environmental sensors like CO_2 sensor and light intensity sensor can be used for providing more thorough environmental sensing of the mushroom growing process.
- Mushroom Growth Estimation and Monitoring: Future developments may involve estimation and monitoring of the mushroom growth stage based on images.

ACKNOWLEDGMENT

The authors would like to thank Mrs. Vidyashree KV from KS School of Engineering and Management for her valuable advice and support during the course of this research. Her valuable inputs and technical knowledge have contributed immensely towards the completion of this research project. They are thankful to her for her motivation and encouragement throughout the course of this research.

REFERENCES

[1] K. A. Kamaruzzaman, R. M. Ramli, and A. A. Ahmad, “A Sustainable IoT-Based Smart System for Mushroom Cultivation Using Fuzzy Logic and Predictive Control,” in *2025* IEEE 8th International Conference on Electrical, Control and Computer Engineering (InECCE), Kuantan, Malaysia:IEEE, Aug.2025.DOI:10.1109/InECCE64959.2025.11151098

© 2026 The Author(s). Published by IJCOPE Journal. Website: https://ijcope.org/



[2] Shilpashree . P. S, C. V, S. Lagali, R. G. N, and C. D, "Automated Disease Detection in Oyster Mushroom Cultivation Using Deep Learning Technique," in 2025 3rd International Conference on Smart Systems for applications in Electrical Sciences (ICSSES), Tumakuru, India: IEEE, Mar. 2025,DOI:10.1109/ICSSES64899.2025.11009558

[3] S. Duman, A. Elewi, and A. Souag, "Real-Time Mushroom Detection and Maturity Classification Using YOLO-Tiny on Raspberry Pi Platform," in 2025 9th International Symposium on Innovative Approaches in Smart Technologies (ISAS), Gaziantep, Turkiye: IEEE, Jun. 2025 DOI: 10.1109/ISAS66241.2025.11101938.

[4] E. Triandini, M. K. Afrianto, B. A. Irawan, A. Maricar, and P. N. Crisnapati, "IoT-Based Automated Environmental Control System for Oyster Mushroom Cultivation Using ESP8266 and Telegram Bot," in 2024 Ninth International Conference on Informatics and Computing (ICIC), Medan, Indonesia: IEEE, Oct. 2024, pp. 1–6. DOI: 10.1109/ICIC64337.2024.10956269.

[5] N. R. Bujal, S. Jusoh, M. Hashim, S. M. Dali, N. S. Ali, and M. N. A. Bidin,"Development of IoT-Based Compact Mushroom Cultivation Monitoring System,"in 2024 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), Malaysia, 2024, pp. 1–6, DOI: 10.1109/ICSIPA62322.2024.10991207.

[6] B. Puviyarasi, G. Logaadhashini, P. D. Shruthi, and S. Suveka, "Smart Mushroom Cultivation with Classification of Mushrooms and Disease Detection using Image Processing," in 2024 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS), Chennai, India: IEEE, Oct. 2024, pp. 1–4. doi: 10.1109/ICPECTS62210.2024.10780137.

[7] J. A. Bautista, J. B. Rivera, I. P. A. Tipan, J. F. Villaverde, and R. V. Pellegrino, "Humidity and Temperature Control System for Oyster Mushroom," in 2023 IEEE 15th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management *(HNICEM)*, Coron, Palawan, Philippines: IEEE, Nov. 2023, pp. 1–6. DOI: 10.1109/HNICEM60674.2023.10589133.

[8] R. Ferdian, R. A. Fattah, and T. Erlina, "Implementation of Environmental Monitoring and Controlling for The Oyster Mushroom Based on The Internet of Things," in 2023 Eighth International Conference on Informatics and Computing (ICIC), Manado, Indonesia: IEEE, Dec. 2023, pp. 1–5. doi: 10.1109/ICIC60109.2023.10381954..

[9] J. J. Elenzano, "IOT - Based Smart Environmental Monitoring for Oyster Mushroom (Pleurotus ostreatus) Production," in 2023 IEEE International Conference on ICT in Business Industry & Government (ICTBIG), Indore, India: IEEE, Dec. 2023, pp. 1–4. doi: 10.1109/ICTBIG59752.2023.10456207.

[10] J. L. Eben, C. Kaur, M. T. Thelly, and Parimita, "IoT based Monitoring of Mushroom," in 2023 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS), Erode, India: IEEE, Mar. 2023, DOI:10.1109/ICSCDS56580.2023.10104815.

[11] I. Rana, A. Gupta, R. Mittal, H. Chaudhary, S. Tyagi, and P. Chawla, "Smart Mushroom Cultivation: Enhancing Growth Conditions through Sensor-Based Monitoring" on Advances in Computation, Communication and Information Technology (ICAICCIT), Faridabad, India: IEEE, Nov.2023,DOI:10.1109/ICAICCIT60255.2023.10466086.

[12] Chong, Jiu Li, Kit Wayne Chew, Angela Paul Peter, Huong Yong Ting, and Pau Loke Show. "Internet of Things (IoT)-Based Environmental Monitoring and Control System for Home-Based Mushroom Cultivation." Biosensors 13, no. 1 (2023): 98. https://doi.org/10.3390/bios13010098.

[13] Rajakumar. P, S. S. B, S. B, and R. S, "Automation and Monitoring System for Mushroom Cultivation using Mobile application and Esp-32," in 2022 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS), Chennai, India: IEEE, Dec. 2022, DOI: 10.1109/ICPECTS56089.2022.10046843

[14] Shafri Shaiful, Mohd Shafiq, Radzi Ambar, Mohd Helmy Abd Wahab, and Muhammad Mahadi Abd Jamil. "IoT-Based Data Monitoring and Environment Controlling System for Oyster Mushroom Mini-Farm." 2022 International Visualization, Informatics and Technology Conference (IVIT), November 1, 2022,DOI:10.1109/IVIT55443.2022.10033410.

[15] Singh, Sampada. "Smart Mushroom Cultivation Using IoT." International Journal of Engineering Research 8, no. 13 (2020).