E-ISSN 2586-9027
Homepage : https://tci-thaijo.org/index.php/SciTechAsia Science & Technology Asia
Vol.31 No.1 January - March 2026 Page: [29-37]
Original research article

Development Coding to Support a Smart Greenhouse System for Household Vegetable Gardening with Convolutional Neural Network

Panyaporn Prangjarote[1], Chirasak Mongkolkeha[1,*], Sakdinan Bumkhunthod[2], Natthaphat Suebloei[2], Natthawat Suebloei[2], Sawitree Duangtagua[2], Surachart Buachum[3]

[1]*Department of Computational Science and Digital Technology, Faculty of Liberal Arts and Science, Kasetsart University, Nakhon Pathom 73140, Thailand*
[2]*Chumphaesuksa School, Khon Kaen 40130, Thailand*
[3]*Technology Digital Business Program, Chandrakasem Rajabhat University, Bangkok 10900, Thailand*

Received 2 October 2025; Received in revised form 14 February 2026
Accepted 23 February 2026; Available online 27 March 2026

ABSTRACT

This study presents the outcomes of the "Coding Coaching and Coding Learning" project, which aimed to enhance technological literacy and programming proficiency among secondary school students and teachers. A smart greenhouse system prototype was developed to facilitate household vegetable cultivation, integrating Internet of Things (IoT) technologies through a project-based learning (PBL) approach. Participants received training in microcontroller programming, sensor integration, and automation system design using the Arduino ESP32 platform and the Blynk application for remote monitoring and control. The prototype enabled real-time environmental management of temperature, humidity, light, and soil moisture. A 35-day cultivation trial involving lettuce, spring onions, and holy basil demonstrated significant improvements in IoT and coding knowledge (scores increased from 2.1 to 4.3, $p < 0.05$). The greenhouse maintained optimal conditions, produced an average of 750 grams per cycle, and saved approximately 30% water compared to traditional irrigation. User satisfaction scores averaged 4.8/5. Additionally, an AI-based CNN model detected leaf abnormalities achieved an accuracy of 87.5% on 500 images, highlighting the potential for small-scale smart farming systems in education and practical application.

Keywords: Digital agriculture; IoT; Home vegetable gardening; Smart greenhouse system

***Corresponding author:** chirasak.m@ku.th **doi:** xx.xxxxx/scitechasia.xxxx.xx

1. Introduction

In the present era, Thailand's agricultural sector faces numerous challenges, including climate change, labor shortages in rural areas, rising production costs, and limited access to advanced technology especially for small-scale farmers and households with limited space and resources [1]. These limitations make it difficult for traditional home gardening practices to cope effectively with environmental fluctuations. Within the context of national development under the Thailand 4.0 [2] policy framework, there is increasing emphasis on integrating digital technologies into agriculture, particularly the Internet of Things (IoT) and automation systems. These technologies can enhance the precision, efficiency, and sustainability of farming practices. However, the adoption of such technologies among youth and households remains limited, largely due to the lack of tangible,hands-on learning opportunities.

In response to these challenges, the Coding Coaching and Coding Learning project was developed to equip youth and teachers with coding skills and technological knowledge through the design and construction of a prototype smart greenhouse system for household vegetable gardening. This system can automatically control internal environmental conditions—such as temperature, humidity, light, and water—using sensors, control devices, and internet-connected alert systems. The initiative aims to promote a deeper understanding of real-world technology applications and to demonstrate that smart agriculture solutions can begin at the household level.

Moreover, the project seeks to support the commercialization and upscaling of these learning outcomes. The primary target group is urban households with limited space who are interested in growing chemical-free vegetables for home consumption or small-scale business. The project utilizes readily available small greenhouse structures commonly sold online—typically 1.5x2 or 2x3 meters in size—which are adapted to integrate with the smart farm system developed under this initiative. The design focuses on cost effectiveness and ease of maintenance, promoting a replicable and scalable model that can be expanded and developed further for household businesses or community-based enterprises in the future.

2. Objectives

1. To utilize a coding-based curriculum as a foundation for learning, studying, and designing a smart greenhouse system.
2. To develop a prototype of a smart greenhouse system for household vegetable cultivation.
3. To conduct real vegetable gardening experiments within the developed system and evaluate its performance.

3. Research Methodology

This study employed a mixed-methods approach, combining quantitative and qualitative techniques to achieve the project objectives. The research was conducted in three structured phases as follows:

Phase 1: Learning and system design using a coding-based curriculum

In the first phase, participants—including secondary school teachers and students engaged in structured learning through the "IoT Coding in AgriTech" curriculum, developed by the Coding Coaching and Coding Learning initiative. The curriculum emphasized project-based learning (PBL) and included key topics

such as Internet of Things (IoT) fundamentals, Arduino ESP32 microcontroller programming in C, sensor integration, and basic smart farm system design. Learning outcomes were assessed through pre- and post tests, and participants' system design prototypes were evaluated based on feasibility, completeness, and innovation by a panel of three experts using a structured rubric.

Phase 2: Development of the smart greenhouse system prototype

Following the design phase, a smart greenhouse system prototype was developed for small-scale household vegetable farming, featuring a $1.4 \times 1.4 \times 1.95$-meter frame equipped with an SHT30 sensor for ambient temperature and humidity, and an RS485 soil moisture sensor. The system included automated irrigation via a DC water pump and ventilation through a 4-inch fan, both controlled by relay modules. An Arduino ESP32 [3, 4] board handled real-time data processing and Wi-Fi communication, while the Blynk application provided a web-based dashboard for remote monitoring and control. The control program was written using the Arduino IDE, and the system was tested to ensure reliable performance under typical environmental conditions.

Phase 3: Experimental cultivation and system performance evaluation

The prototype was tested in a 35-day cultivation experiment with leafy vegetables including lettuce, spring onions, and holy basil. The internal environment (temperature, humidity, soil moisture) was monitored continuously using data loggers, while plant growth data (height, leaf count, biomass) were recorded every 5 days. To evaluate learning outcomes, a paired-sample t-test was used to compare pre- and post-training scores in IoT knowledge and coding skills, with significance set at $p < 0.05$. Usability and satisfaction were measured using a 5-point Likert scale questionnaire (Cronbach's alpha = 0.89), and observational checklists were used to assess system reliability and ease of maintenance. Water usage was measured daily to compare efficiency against traditional watering practices.

4. System Design and Prototype

The smart greenhouse prototype was designed to integrate key components for efficient environmental monitoring and control. It utilizes a SHT30 sensor to measure temperature and humidity, along with a soil moisture sensor connected via an RS485 module to ensure precise soil condition tracking. An Arduino ESP32 microcontroller functions as the central controller, enabling real-time data transmission over Wi-Fi and managing automated operations. The system includes a small DC water pump for mist irrigation and a 4-inch fan for automatic ventilation, both controlled via relay modules. For user interaction, data visualization and remote control are facilitated through a web-based dashboard, allowing users to monitor conditions and manage the system conveniently via smartphones or computers.

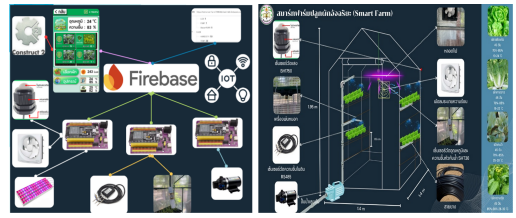

Fig. 1. Design of a smart farm system for indoor vegetable cultivation.

Fig. 1 illustrates a smart greenhouse system developed for efficient vegetable cultivation using IoT-based automa-

tion technology. This compact setup is designed to support the growth of crops such as green oak lettuce, Chinese cabbage, regular cabbage, and Chinese kale. The system integrates multiple sensors and automated devices to create optimal growing conditions, including a BH1750 light sensor for monitoring light in tensity, an RS485 soil moisture sensor, an SHT30 temperature and humidity sensor, a misting device for humidity control, a high-pressure water pump for irrigation, LED plant growth lights for supplemental lighting, a ventilation fan for air circulation, and a water distribution system. These components are connected and managed through a centralized platform using Firebase and a mobile application developed with Construct 3. The system supports both manual and automatic operation modes. For instance, it can automatically activate LED lights when the ambient light level drops below 5 lux, or initiate misting when humidity falls below 70%. Through continuous monitoring and real-time control of environmental factors such as light, humidity, temperature, and soil moisture, the system enhances growing efficiency, increases crop yield, and reduces manual labor, offering a scalable model for smart farming applications. To implement these functions, the system utilizes embedded code running on an Arduino ESP32 microcontroller. This code integrates sensor data with cloud services and remote notification platforms. The following snippet demonstrates the core logic of the environmental control program using Arduino, Blynk, and Firebase:

```
#define BLYNK_PRINT Serial
#include <WiFi.h>
#include <BlynkSimpleEsp32.h>
#include <Wire.h>
#include "Adafruit_SHT31.h"
#include <FirebaseESP32.h> //
    Include Firebase library
//Blynk credentials
char auth[] = "The_Blynk_Auth_Token
    ";
char ssid[] = "The_SSID";
char pass[] = "The_WIFI_Password";
// Firebase credentials
#define FIREBASE_HOST "the-project-
    id.firebaseio.com"
#define FIREBASE_AUTH "
    The_Firebase_Database_Secret"
FirebaseData firebaseData;
Adafruit_SHT31 sht30 =
    Adafruit_SHT31();
#define soilMoisturePin 34
#define relayPin 5
int soilMoistureThreshold = 400;
float tempThreshold = 30.0;
float humidityThreshold = 60.0;
void setup() {
 Serial.begin(115200);
 Blynk.begin(auth, ssid, pass);
 WiFi.begin(ssid, pass);
 Firebase.begin(FIREBASE_HOST,
     FIREBASE_AUTH);
 Firebase.reconnectWiFi(true);
 pinMode(relayPin, OUTPUT);
 digitalWrite(relayPin, LOW);
 if (!sht30.begin(0x44)) {
  Serial.println("SHT30 not found."
      );
  while (1) delay(1);
 }
}
void loop() {
 Blynk.run();
 int soilMoisture = analogRead(
     soilMoisturePin);
 float temp = sht30.readTemperature
     ();
 float humidity = sht30.
     readHumidity();
 Serial.print("Soil Moisture: ");
 Serial.println(soilMoisture);
 Serial.print("Temperature: ");
 Serial.print(temp); Serial.print("
      C\t");
 Serial.print("Humidity: ");
 Serial.print(humidity); Serial.
     println(" %");
 // Send live data to Blynk Virtual
      Pins
 Blynk.virtualWrite(V1,soilMoisture
     );
 Blynk.virtualWrite(V2,temp);
 Blynk.virtualWrite(V3,humidity);
 // Send data to Firebase
 Firebase.setFloat(firebaseData,"/
     greenhouse/temperature",temp);
 Firebase.setFloat(firebaseData,"/
     greenhouse/humidity",humidity)
     ;
 Firebase.setInt(firebaseData,"/
```

```
        greenhouse/soilMoisture",
        soilMoisture);
    // Watering logic based on sensor
        thresholds
    if (soilMoisture >
        soilMoistureThreshold || temp
        > tempThreshold || humidity <
        humidityThreshold) {
     digitalWrite(relayPin,HIGH);
     Serial.println("Watering
         Activated...");
     Blynk.notify("Smart Greenhouse:
         Watering Started");
     Firebase.setString(firebaseData,"
         /greenhouse/status", "
         Watering ON");
     delay(5000);
     digitalWrite(relayPin,LOW);
     Serial.println("Watering
         Deactivated.");
     Blynk.notify("Smart Greenhouse:
         Watering Stopped");
     Firebase.setString(firebaseData,"
         /greenhouse/status", "
         Watering OFF");
    }
    delay(10000);  // Wait for 10
        seconds before next reading
}
```

This program implements an automated smart greenhouse system using an ESP32 microcontroller integrated with Blynk for mobile monitoring and Firebase Realtime Database for cloud-based data logging. It continuously reads environmental parameters—including temperature and humidity from an SHT30 sensor and soil moisture from an analog sensor—then transmits this data to both the Blynk dashboard and Firebase database for real-time visualization and historical tracking. The system uses predefined threshold values to determine when to activate a water pump via a relay, issuing push notifications through Blynk and updating status logs in Firebase. This dual integration not only enables local control and alerts but also supports cloud-based diagnostics and future data analytics, enhancing the reliability and scalability of small-scale digital agriculture systems.

Fig.2 demonstrates the integration of a smart greenhouse system for vegetable cultivation using IoT technology. The left and right panels exhibit project posters out lining the objectives, system architecture, and educational benefits. At the center, a compact greenhouse structure is shown with multiple plant beds and automated components such as lighting, ventilation, and irrigation systems. The system is equipped with a user-friendly digital interface that enables real-time monitoring and control. Users can remotely operate essential devices—namely grow lights, fans, and water pumps—via intuitive toggle switches. Additionally, the interface allows users to select specific plant types for each planting bed and configure automated operation schedules based on plant requirements. These features support precision agriculture by adjusting environmental conditions such as light intensity, humidity, and watering cycles according to the selected crop, enhancing both productivity and learning outcomes in smart farming applications.

Fig. 2. Smart greenhouse control interface for vegetable cultivation.

Exploratory Student-Led Experiment on Plant Health Monitoring via Image Processing

As part of the educational component of the project, a separate exploratory experi-

ment was conducted by student participants to assess the feasibility of integrating image processing techniques for plant health monitoring . The objective was to detect common leaf abnormalities, such as chlorosis (yellowing), necrosis (dry patches), and spot lesions, using computer vision techniques [5–7]. A dataset of 500 images of leafy vegetables was collected using an ESP32-CAM camera module under consistent lighting and background conditions within the smart greenhouse environment.

The labeling process was conducted in collaboration with agricultural experts, who categorized the images into four classes: healthy, yellowing, necrotic, and spotted leaves. To ensure reproducibility and unbiased evaluation, the dataset was randomly divided into training (60%), validation (20%), and testing (20%) subsets using a stratified sampling strategy (Table 1). The training and validation subsets were used for CNN model development and hyperparameter tuning with TensorFlow Lite, while the testing subset, consisting solely of original images, was reserved for independent performance evaluation. The dataset was intentionally balanced across all classes using a stratified sampling approach to ensure fair model training and evaluation.

Although the original dataset size was relatively limited, data augmentation techniques were applied exclusively to the training subset to enhance data diversity and model generalization. These techniques included random rotation (±15°), horizontal and vertical flipping, brightness adjustment, and slight zooming. Using the augmented training dataset, a convolutional neural network (CNN) model was trained with TensorFlow Lite, while the validation set was used for hyperparameter tuning and early stopping. This experimental design ensured that the reported classification results reflect the model's true generalization capability without data leakage.

The CNN architecture consisted of three convolutional blocks. Each block included a 3×3 convolutional layer with ReLU activation, followed by a 2×2 max-pooling layer. The number of filters was set to 32, 64, and 128 for the first, second, and third convolutional layers, respectively. Dropout layers with a rate of 0.25 were applied after the second and third convolutional blocks to reduce overfitting. The extracted feature maps were flattened and passed to a fully connected dense layer with 128 neurons, followed by a softmax output layer for four-class classification.

The model was trained using the Adam optimizer with a learning rate of 0.001 and the categorical cross-entropy loss function. Overall classification performance was evaluated using accuracy, confusion matrix analysis, and class-wise precision, recall, and F1-score metrics based on the independent test dataset.

Although the dataset was collected as part of an educational project, the authors plan to release an anonymized version of the image dataset for academic use upon reasonable request. This approach aligns with open science practices and supports future comparative studies in smart agriculture and plant health monitoring.

The softmax function was used in the output layer to perform multi-class classification and is defined as follows:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \quad for\ i = 1, 2, \ldots, K \tag{4.1}$$

where z_i is the logit (output) for class i, and K is the total number of classes. The model was trained using the categorical

cross-entropy loss function:

$$L = -\sum_{i=1}^{K} y_i \log(\hat{y}_i), \qquad (4.2)$$

where y_i is the true label and $\hat{y}_i$ is the predicted probability for class i.

The model achieved an overall classification accuracy of 87.5% on the test dataset. Class-wise precision scores were as follows: healthy (90.2%), yellowing (85.0%), necrosis (84.5%), and spotted (86.3%). The experiment demonstrated the potential to implement a lightweight, on-device diagnostic system for early detection of plant stress symptoms. Integration with the existing smart greenhouse system via a mobile dashboard (e.g., Blynk or Firebase) could enable real-time alerts to users. These findings suggest that image-based monitoring can serve as a complementary tool to sensor data, enhancing the reliability of home-scale digital agriculture systems.

To assess the classification performance of the trained CNN model beyond accuracy, the **F1-score** was calculated for each class. The F1-score combines both **precision** and **recall** into a single metric that balances the trade-off between false positives and false negatives, especially useful in multi-class classification problems like plant health monitoring.

$$F1 - score = 2 \times \frac{Precision \times Recall}{Precision + Recall}, \qquad (4.3)$$

where:

$$Precision = \frac{TP}{TP + FP}, \qquad (4.4)$$

$$Recall = \frac{TP}{TP + FN}. \qquad (4.5)$$

These scores confirm that the model achieves consistent and balanced performance across all classes, with F1-scores ranging between 84%–90%, reflecting high reliability in detecting common plant stress symptoms through image-based diagnostics, as shown in Tables 2-3 .

5. Results

The project demonstrated notable success in both educational outcomes and technical implementation. Post-project evaluations indicated a statistically significant improvement ($p < 0.05$) in participants' knowledge of IoT and agricultural automation. The average self-assessed coding proficiency increased from 2.1 to 4.3 (on a 5-point scale), while understanding of smart farming system concepts rose from 2.5 to 4.5. The developed smart greenhouse prototype effectively automated the control of temperature, humidity, light, and irrigation. The system was integrated with the mobile application, enabling real-time monitoring and remote operation via smartphones. User feedback from both teachers and students reflected high levels of satisfaction, with an average score of 4.6 for ease of use and 4.7 for understanding of the system's architecture.

Field trials involving the cultivation of lettuce, spring onions, and holy basil confirmed the system's operational reliability. Environmental conditions within the greenhouse were maintained within optimal ranges—temperature between 24–30°C and humidity between 60–80%—supporting healthy plant growth. Each cultivation cycle yielded approximately 750 grams of produce over a 35-day period. Furthermore, the automated system achieved approximately 30% water savings compared to conventional irrigation methods. Overall user satisfaction with the system was rated at 4.8 out of 5. These results affirm the project's effectiveness in enhancing digital literacy and practical innovation

Table 1. Dataset composition and data splitting strategy.

Class	Total Images	training(60%)	validation(20%)	testing(20%)
Healthy	125	75	25	1
Yellowing	125	75	25	2
Necrotic	125	75	25	5
Spotted	125	75	25	45
Total	500	300	100	100

Table 2. Confusion Matrix of CNN Classification Results (number of images).

Actual\Predicted	Healthy	Yellowing	Necrotic	Spotted
Healthy	45	3	1	1
Yellowing	2	43	3	2
Necrotic	1	2	42	5
Spotted	0	3	2	45

Table 3. F1-Scores based on confusion matrix.

Class	Precision(%)	Recall(%)	F1-Score(%)
Healthy	90.0	90.0	90.0
Yellowing	86.0	86.0	86.0
Necrotic	84.0	84.0	84.0
Spotted	85.0	90.0	87.4

among youth and educators. It successfully bridged technological education with real-world agricultural application, fostering a strong foundation for continued interest and innovation in smart farming solutions.

6. Conclusion and Recommendations

The project successfully achieved its objectives by delivering a replicable smart greenhouse system and enhancing participants' coding and IoT competencies. The integration of automated environmental control, mobile-based monitoring, and real-time data logging to Firebase enabled effective and accessible digital farming practices. The plant health monitoring module using CNN image classification further demonstrated the value of incorporating artificial intelligence into agriculture, achieving over 87% accuracy in diagnosing leaf conditions.

Looking ahead, the system could be extended by integrating additional sensors (e.g., CO□, pH, and soil nutrient sensors), applying edge computing solutions such as Raspberry Pi or Jetson Nano for on-device inference, and supporting long-term data analytics via cloud dashboards. The approach also has potential for scaling into mid-sized greenhouses and community co-farming models. Educationally, the project could be expanded through open-source deployment, coding camps, and collaboration with academic institutions to foster widespread adoption of smart farming in both urban and rural settings.

References

[1] Tzounis A, Katsoulas N, Bartzanas T, & Kittas C. Internet of Things in agriculture, recent advances and future challenges. Biosystems Engineering 2017;164:31-48.

[2] Ministry of Digital Economy and Society, Thailand. (2018). Thailand Digital Economy and Society Development Plan (Digital Thailand).

[3] Arduino. (n.d.). Arduino IDE. Available from: https://www.arduino.cc/en/software

[4] Espressif Systems. (n.d.). ESP32 Series Datasheet. Available from: https://www.espressif.com/en/products/socs/esp32

[5] Singh V, Sharma N, & Singh S. A review of imaging techniques for plant disease detection. Artificial Intelligence in Agriculture 2020;4:229-42.

[6] Sajitha P, Andrushia AD, Anand N, & Naser MZ. A review on machine learning and deep learning image-based plant disease classification for industrial farming systems. Journal of Industrial Information Integration 2024;38:100572.

[7] Saleem MH, Potgieter J, & Arif KM. Plant disease detection and classification by deep learning. Plants 2019;8(11):468.