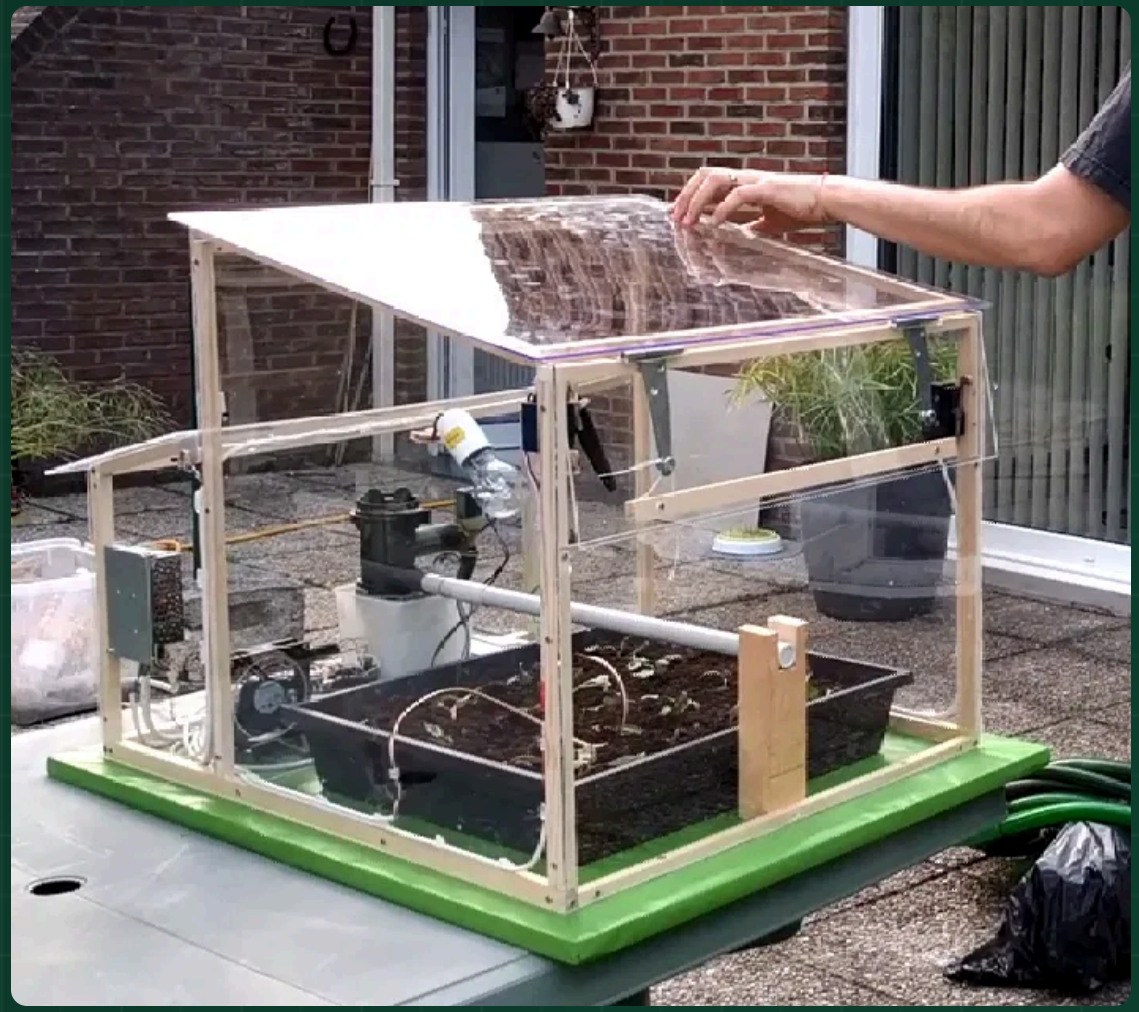
Automating a Mini Greenhouse with IoT

A Smart Solution for Controlled Agriculture



Presenter: Omid Mohebi

Date: December 2025

The Challenge

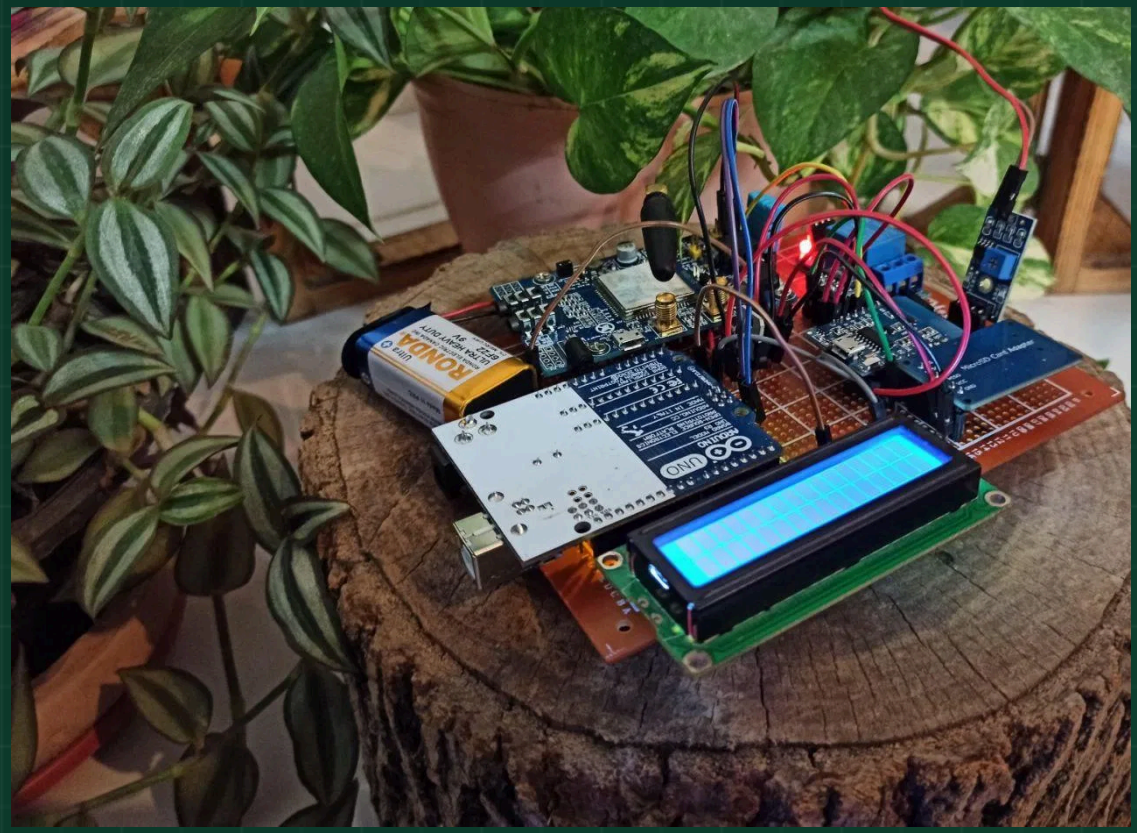
Maintaining optimal and consistent environmental conditions for plant growth is crucial in greenhouse management. Manual monitoring and adjustment of temperature, light, and water levels are time-consuming, prone to human error, and inefficient.

The Solution

An automated, low-cost Internet of Things (IoT) system built around an Arduino microcontroller. This system continuously monitors and regulates the environment, ensuring a stable and healthy ecosystem for the plants.

Key Objectives

- Temperature Regulation: Maintain stable temperature to prevent thermal stress
- Light Management: Regulate light intensity and provide necessary shading
- Hydration Control: Ensure adequate water/moisture levels for plant health



System Architecture

A Closed-Loop Control System

Component Function	Specific Components	Role in the System
Input (Sensors)	LM35, LDR, Potentiometer	Collect real-time environmental data (Temp, Light, Water Level)
Processing (Controller)	Arduino UNO	Executes the control logic based on sensor readings
Output (Actuators)	Fan, Lamp/Heater, Water Pump, Shading Motor	Modify the environment to meet set-point conditions
Drivers/Interface	L293D, ULN2003A, Relay	Safely interface the low-power Arduino with high-power actuators

Environmental Monitoring: The Sensor Suite

LM35 Temperature Sensor

Purpose: Measures ambient temperature

Arduino Pin: A4

Data Type: Analog (Voltage proportional to Temp)

LDR (Light Dependent Resistor)

Purpose: Measures light intensity

Arduino Pin: A0

Data Type: Analog (Resistance changes with light)

Potentiometer (Water Level)

Purpose: Simulates/Measures water level or soil moisture

Arduino Pin: A5

Data Type: Analog (Resistance changes with level)

Environmental Control

Actuators and Drivers

The system uses a variety of actuators to modify the greenhouse environment. These high-power devices are safely interfaced with the low-power Arduino through dedicated driver circuits.

Actuator	Purpose	Driver/Interface	Arduino Pin
Fan	Cooling (when hot)	ULN2003A/Relay	7
Lamp/Heater	Heating (when cold) or supplemental light	ULN2003A/Relay	5
Water Pump	Irrigation (when water level is low)	ULN2003A/Relay	6
Shading Motor	Opens/closes shading to regulate light	L293D Motor Driver	2, 3

Control Logic 1: Temperature Regulation

The system maintains temperature stability by toggling the fan and lamp/heater based on a fixed threshold of 20°C

TEMPERATURE CONTROL LOGIC

Condition	LM35 Reading	Action	Output State
Cold	Temp ≤ 20°C	Turn on the Lamp (Heater)	`lampPin` HIGH, `fanPin` LOW
Hot	Temp > 20°C	Turn on the Fan (Cooling)	`lampPin` LOW, `fanPin` HIGH

ARDUINO IMPLEMENTATION

```
if (temp <= 20) {
  digitalWrite(lampPin, HIGH);
  digitalWrite(fanPin, LOW);
  Serial.print("Its COLD, Turn on the Heater : ");
} else {
  digitalWrite(lampPin, LOW);
  digitalWrite(fanPin, HIGH);
  Serial.print("Its HOT, Turn ON the FAN : ");
}
```

Control Logic 2: Light and Shading Management

Light intensity is managed by activating a shading mechanism and supplemental lighting. The LDR analog reading threshold is set at **500**.

Condition	LDR Reading	Action	Output State
Bright	LDR Status ≤ 500	Activate Shading (Close)	`shadingMotorPin1` HIGH, `shadingMotorPin2` LOW
Dark	LDR Status > 500	Deactivate Shading (Open) & Turn on LED	`shadingMotorPin1` LOW, `shadingMotorPin2` HIGH, `ledPin` HIGH

Code Snippet

Arduino Implementation:

```
if (ldrStatus <= 500) {
  digitalWrite(ledPin, LOW);
  digitalWrite(shadingMotorPin1, HIGH);
  digitalWrite(shadingMotorPin2, LOW);
} else {
  digitalWrite(ledPin, HIGH);
  digitalWrite(shadingMotorPin1, LOW);
  digitalWrite(shadingMotorPin2, HIGH);
}
```

Control Logic 3: Water Level Monitoring

Ensuring Adequate Hydration for Plant Health

The potentiometer simulates a water level sensor, ensuring the pump is activated only when necessary. The analog reading threshold is set at **200**.

Condition	Potentiometer Reading	Action	Output State
Low Water	Pot Status ≤ 200	Turn on the Water Pump	`pumpPin` HIGH
Good Water	Pot Status > 200	Turn off the Water Pump	`pumpPin` LOW

CODE SNIPPET

```
if (potStatus <= 200) {
  digitalWrite(pumpPin, HIGH);
} else {
  digitalWrite(pumpPin, LOW);
}
```

This control logic ensures optimal plant hydration by automatically activating irrigation when water levels drop below the threshold.

Simulation and Validation with Proteus

The entire system was validated using the **Proteus ISIS Simulation Software** before physical implementation. This approach ensures reliability and reduces development costs.

Key Simulation Benefits

Safe Testing: Allows testing of extreme environmental conditions without damaging physical components.

Component Verification: Confirms correct interfacing and operation of the Arduino, L293D, ULN2003A, and all sensors/actuators.

Debugging: Facilitates easy debugging of the Arduino control logic before hardware deployment.

Simulation Components Used

ARDUINO UNO	LM35	LDR
POT-HG	L293D	ULN2003A
MOTOR	10WATT1K	LED-RED
LED-YELLOW		

Conclusion and Future IoT Integration

Project Achievement

The project successfully implemented a multi-parameter, automated control system for a mini-greenhouse using the Arduino platform. The system demonstrates effective closed-loop control of temperature, light, and water levels, with all control logic validated through Proteus simulation before physical implementation.

Future Scope: True IoT Integration

To achieve full Internet of Things capability, the next steps include:

- **Remote Monitoring:** Connect the Arduino to a cloud platform (ThingSpeak, Firebase) to log sensor data and enable real-time environmental tracking.
- **Remote Control:** Enable users to adjust thresholds or manually override actuators via a web or mobile application from anywhere.
- **Advanced Control:** Integrate a Real-Time Clock (RTC) for time-based scheduling and add a humidity sensor for comprehensive environmental management.