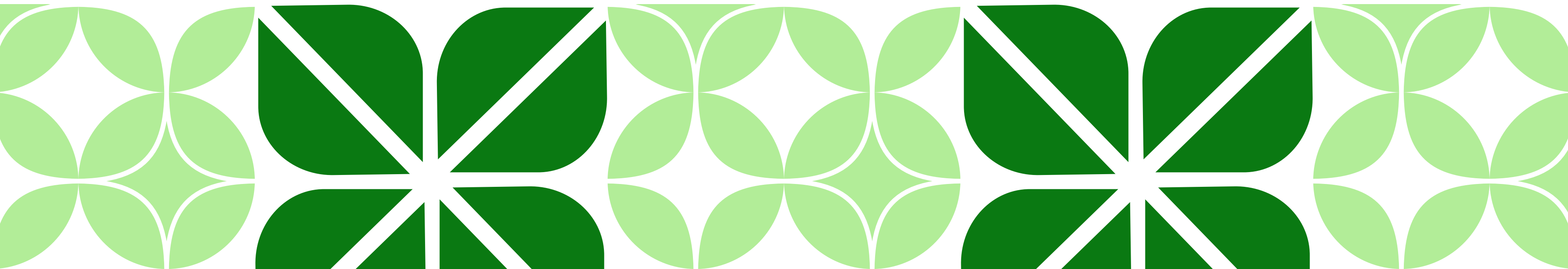


# CULTIVATION CONTROL PROGRAM

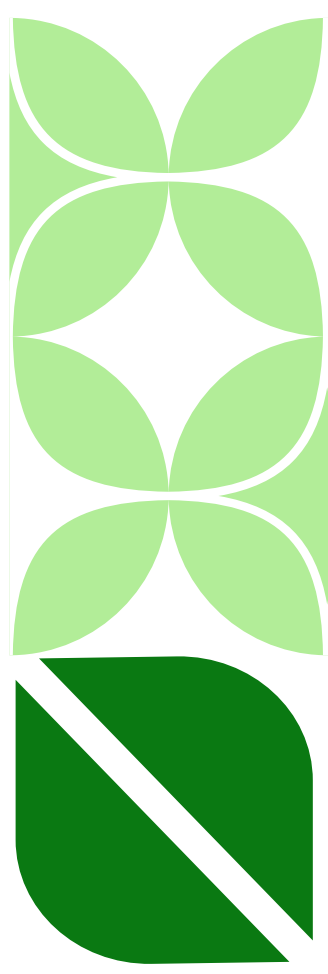
Serverless Computing A.Y. 22/23 - Baggiano Antonio - 0522501526





# ABSTRACT

The Cultivation Control Program (CCP) is a smart agricultural monitoring system designed to collect and analyze environmental data (temperature, humidity) from multiple IoT devices in different fields.





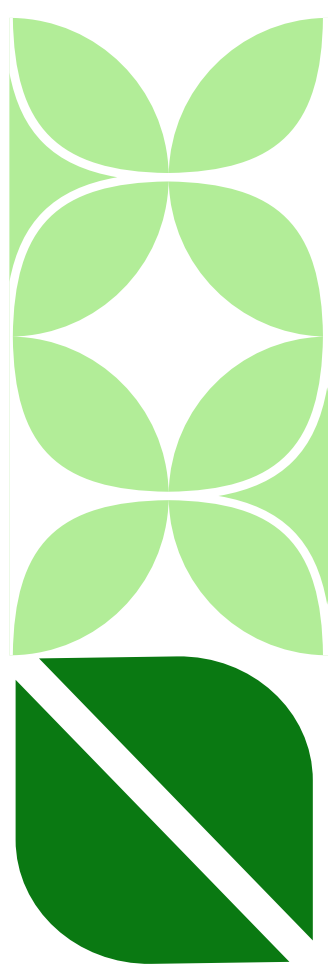
# PROBLEM

## ■ What we have to do

We operate a farming business with four greenhouse fields, each designated for a different crop. Each of the four crops requires a unique temperature and humidity environment to thrive.

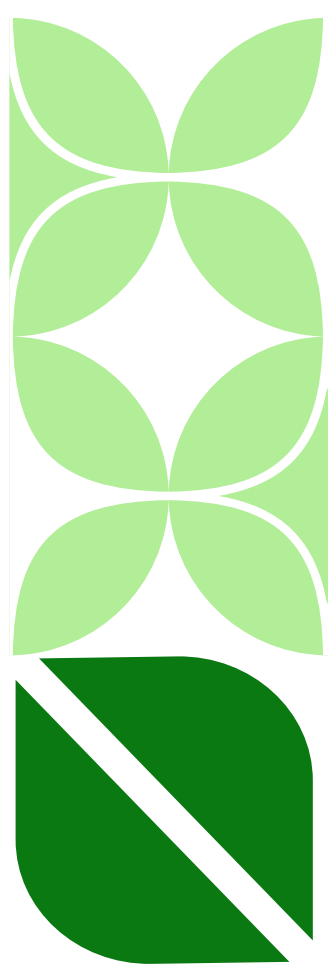
## ■ The core Problem

The core challenge is managing fluctuations in temperature and humidity, alerting operators whenever the specific thresholds for each crop are exceeded.





# GOAL

- Automate the collection of temperature and humidity data for each cultivated greenhouse.
  - Automate the delivery of alerts when specified thresholds are reached.
- 

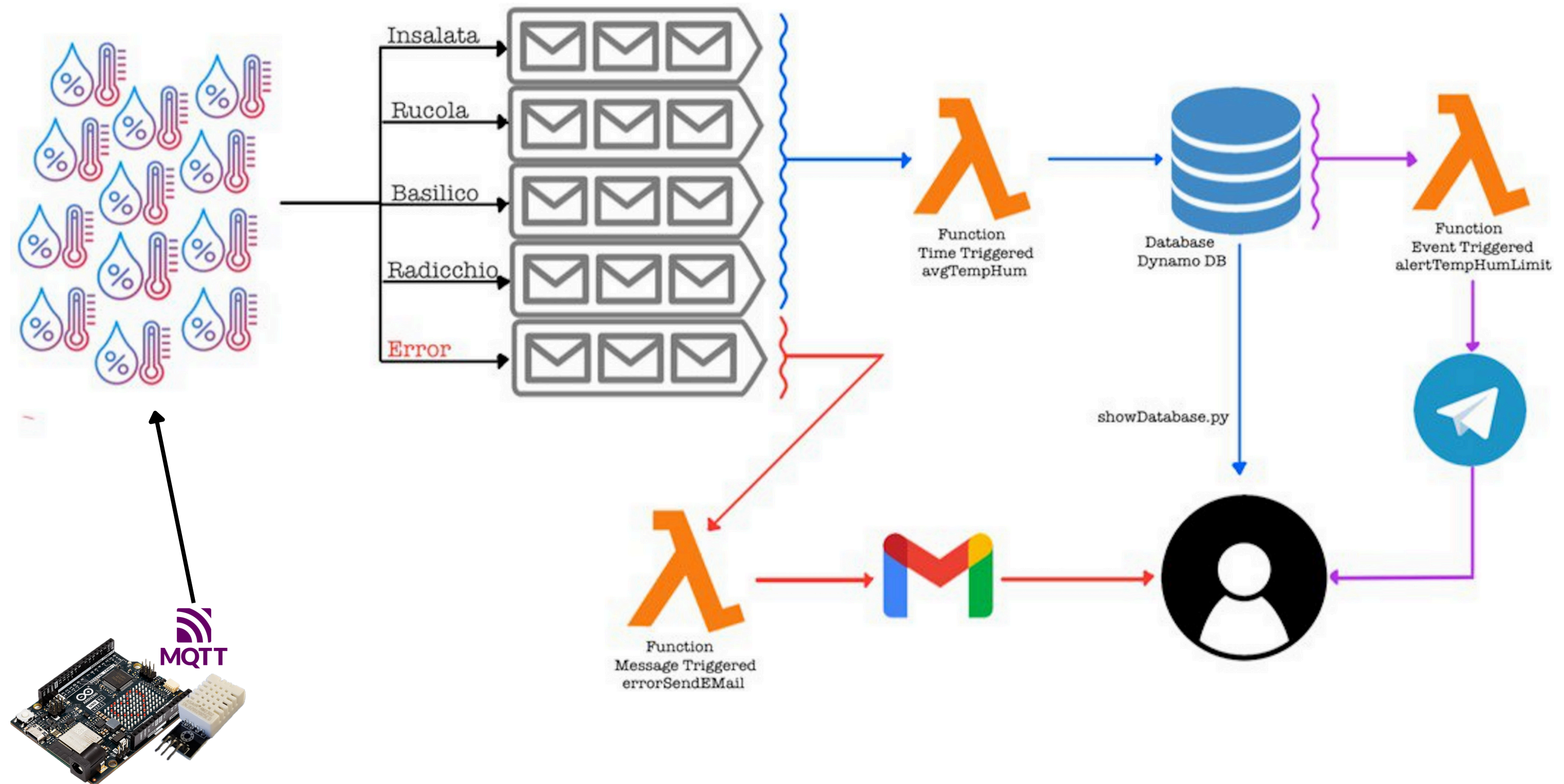


# METHODOLOGY

We will use DHT22 sensors installed on Arduino UNO WiFi Rev4 boards to gather data, processing it with a full range of AWS tools. Alerts will be automatically generated through AWS functions whenever specified thresholds are exceeded.

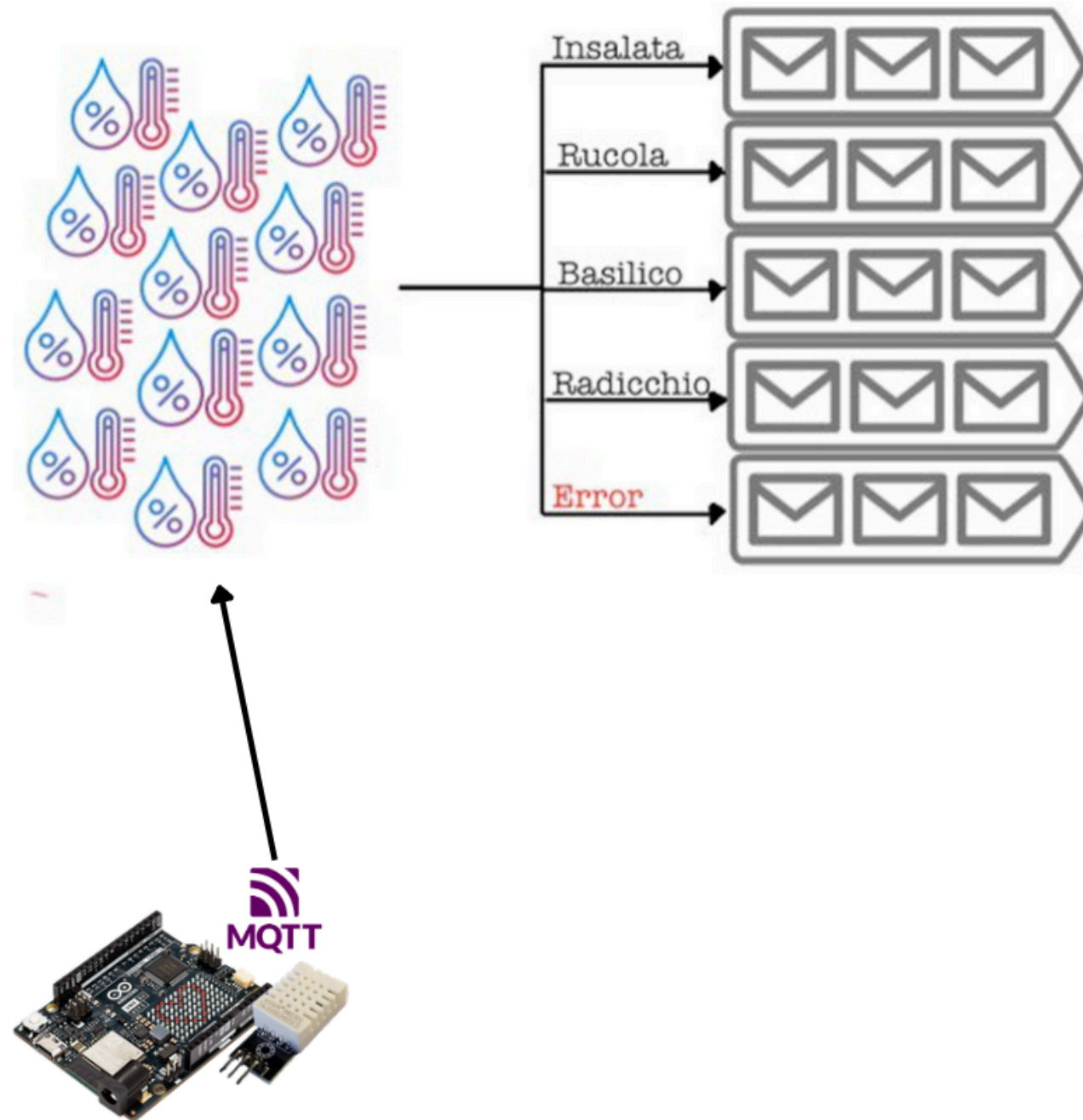


# ARCHITECTURE





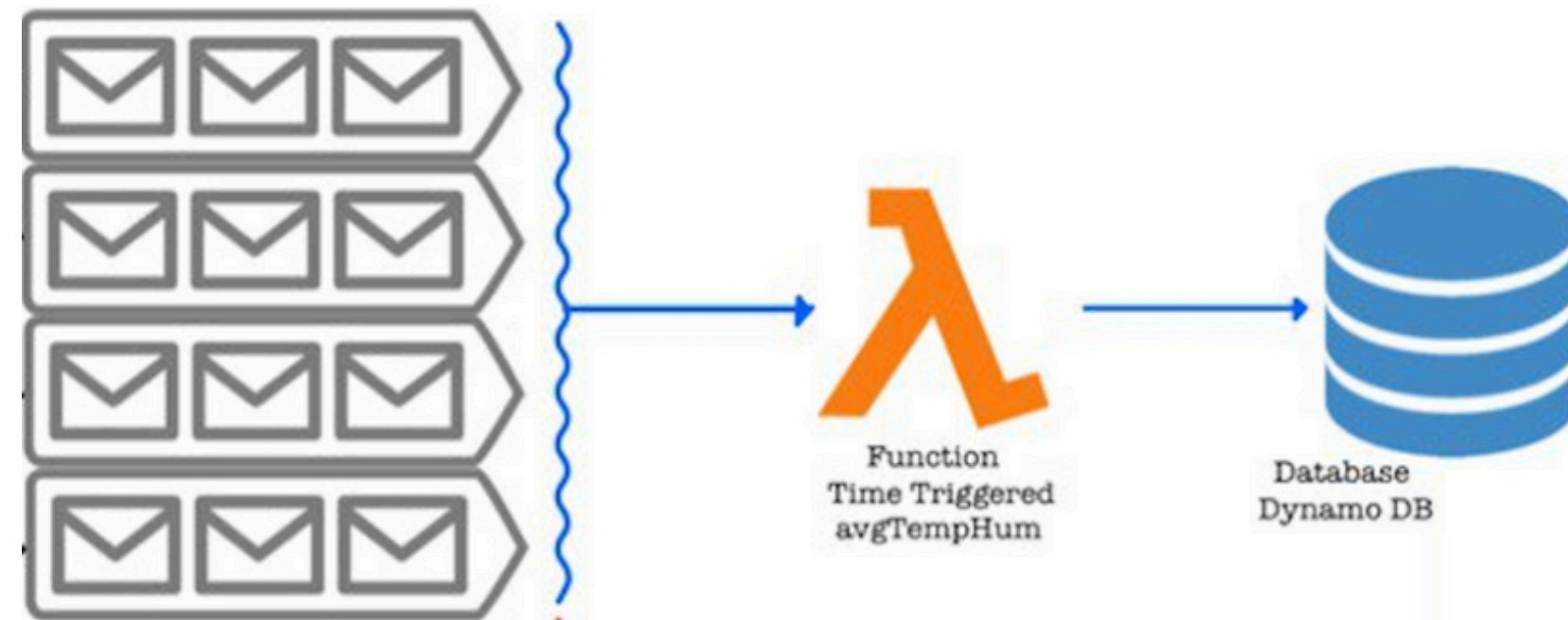
# DATA ACQUISITION



Three Arduino UNO WiFi Rev4 boards are placed within each greenhouse, regularly collecting temperature and humidity data using DHT22 sensors. These data points are transmitted in JSON format via the MQTT protocol to a broker, organized by crop-specific topics. A consumer listening to each topic then inserts the data into AWS SQS queues, separated by crop type. Since the Arduinos may occasionally produce errors, when these occur, the consumer inserts the event into a dedicated “Error” queue.

Note: The GitHub project includes the option to either connect real Arduino boards or populate the queues artificially with randomized data.

# AVERAGE CALCULATION

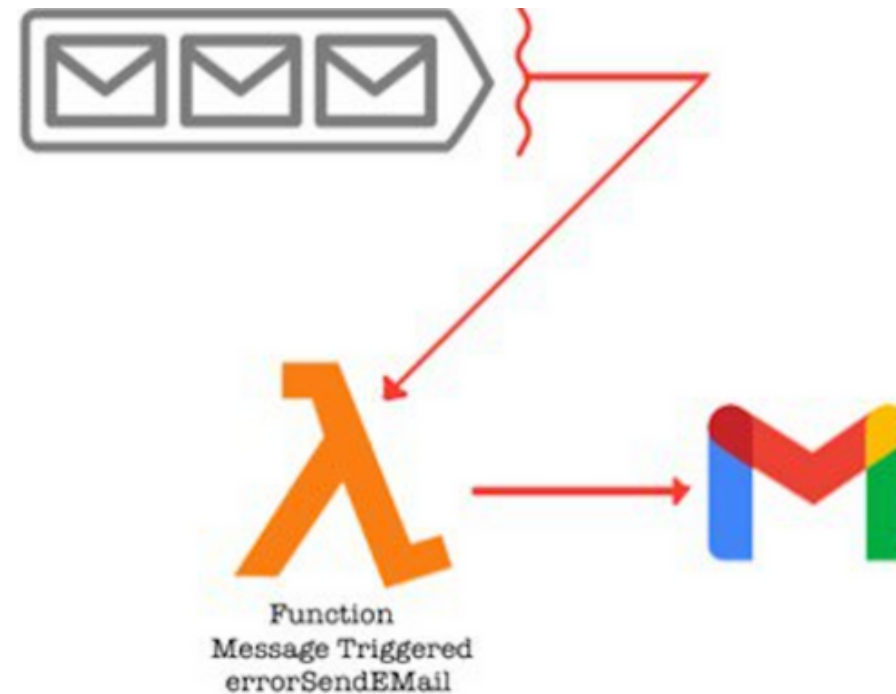


A primary Lambda function consumes messages from the SQS queues associated with the three greenhouses, calculates the average values, and then inserts the results into a DynamoDB database.

The Lambda function is triggered by a CloudWatch rule every 5 minutes to ensure that the data in the database is regularly updated. This interval allows for near real-time monitoring by keeping the stored values current every 5 minutes.

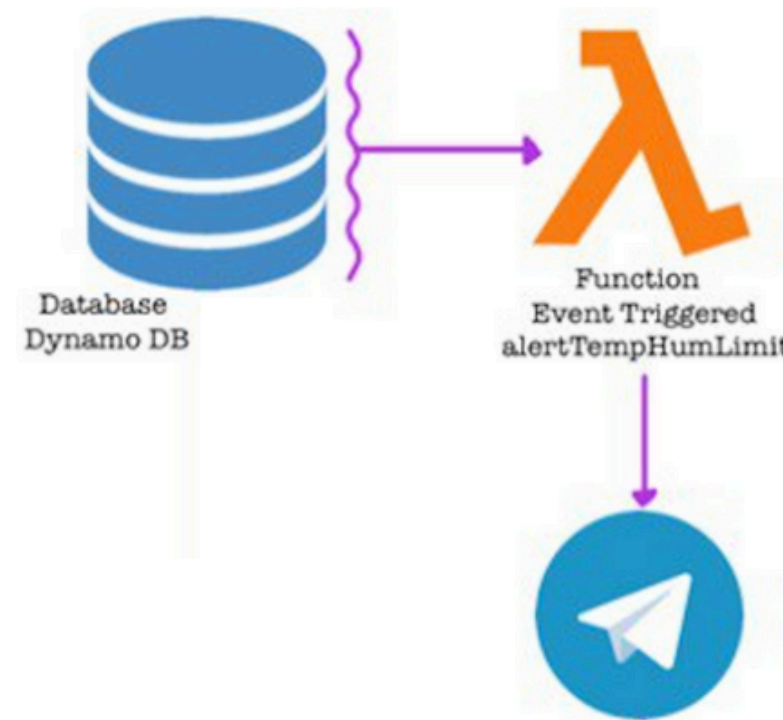


# SENSOR'S ERROR



Another Lambda function is event-triggered to listen for incoming messages in the “Error” SQS queue. Each time a new message arrives, it is immediately consumed by the Lambda, which then translates the message into an email sent to a specified user. I am using a Python library that leverages Gmail’s SMTP service, so a Gmail account is required. Although I could have used AWS SNS, I preferred Gmail for real feedback, as LocalStack only simulates email delivery.

# OUT-OF-LIMIT ALERT

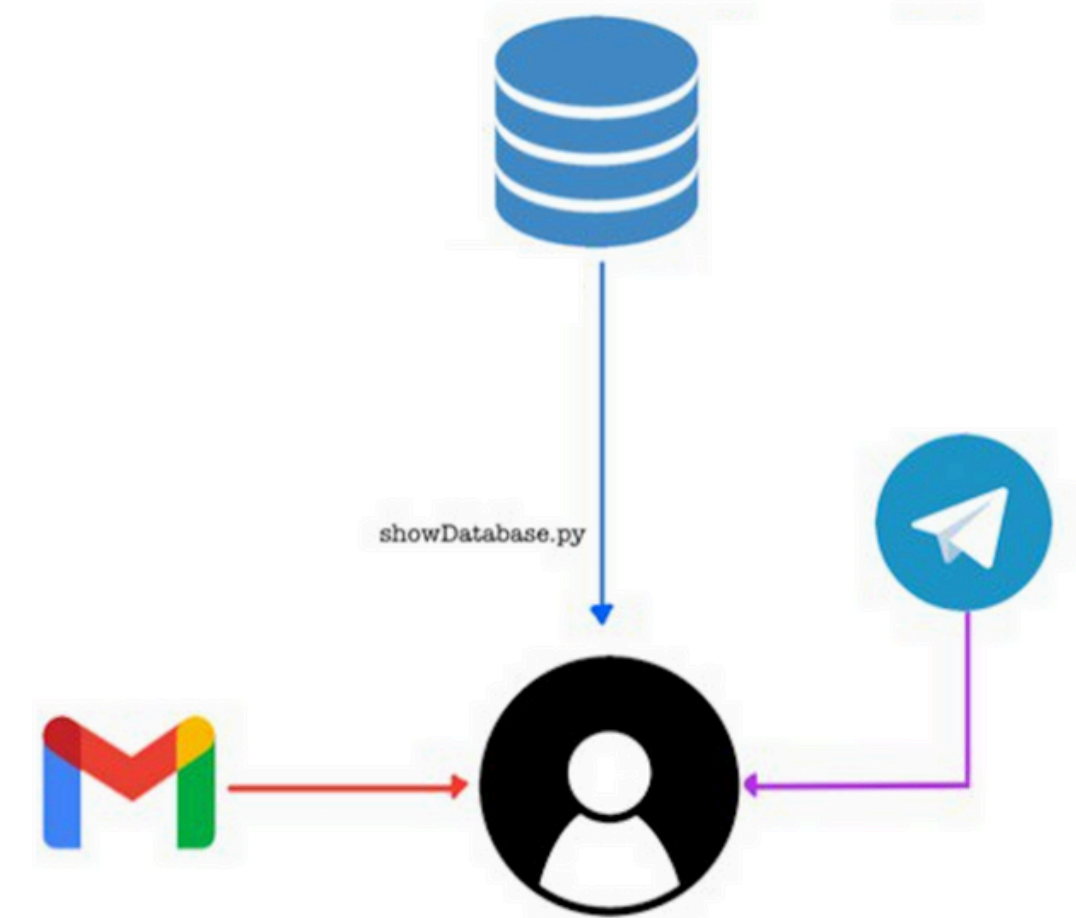


A final Lambda function is event-triggered to monitor updates in the DynamoDB database. When an average value exceeds predefined thresholds for each crop, a notification is sent to a specified user via a Telegram bot.

# RESULTS

The final result is that the user is notified via email in case of errors and via Telegram when thresholds are exceeded. Additionally, the user can check the status independently using two methods: a text-based Python script and a Java GUI.

cultivationName	device_ids	measure_date	temperature	humidity
Insalata	Insalata_0 Insalata_1 Insalata_2	2024-10-12 02:14:46	29.37	62.09
Basilico	Basilico_0 Basilico_1 Basilico_2	2024-10-12 02:14:46	13.43	61.09
Rucola	Rucola_0 Rucola_1 Rucola_2	2024-10-12 02:14:46	26.79	30.96
Radicchio	Radicchio_0 Radicchio_1 Radicchio_2	2024-10-12 02:14:46	11.41	66.64



DynamoDB Viewer

cultivationName	device_ids	measure_date	temperature	humidity
Insalata	Insalata_0 Insalata_1 Ins...	2024-10-28 11:14:03	19.2	45.88
Basilico	Basilico_0 Basilico_1 Basi...	2024-10-28 11:14:03	28.88	46.33
Rucola	Rucola_0 Rucola_1 Rucol...	2024-10-28 11:14:03	20.26	34.26
Radicchio	Radicchio_0 Radicchio_1...	2024-10-28 11:14:03	29.94	65.65

Load Data



# FUTURE UPDATE

- Improvements to Arduino Sketch and Implementation of LoRa Systems for Data Collection: Enhance the current Arduino sketches and explore the integration of LoRa technology to improve data transmission and expand range.
- Collect External Temperature Data for Predictive Modeling: Gather external temperature data outside the greenhouses to enable comparisons and create a predictive model that estimates internal temperatures based on external conditions.
- Integrate Output Devices for Automated Control: Interface with “output” devices, such as sprinkler systems and ventilation units, to automate environmental control within the greenhouses.



**THANK YOU**

