Event-DrivenSmartAgricultureMonitoring UsingAWSCloudServices

MScResearchProject
MSc in Cloud Computing

VIVEKANANDAKANUKULA
StudentID:23340312

SchoolofComputing
NationalCollegeofIreland

Supervisor: Jorge Mario Cortes Mendoza

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	VIVEKANANDA KANUKULA
Student ID:	23340312
Programme:	MSc in Cloud Computing
Year:	2025
Module:	MSc Research Project
Supervisor:	Jorge Mario Cortes Mendoza
Submission Due Date:	15/09/2025
Project Title:	Event-Driven Smart Agriculture Monitoring Using AWS Cloud Services
Word Count:	6067
Page Count:	19

Signature:	vivekananda
Date:	15th September 2025

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Event-Driven Smart Agriculture Monitoring Using AWS Cloud Services

VIVEKANANDA KANUKULA
23340312

Abstract

Ineffective irrigation methods typically reduce agricultural productivity, leading to water waste and crop stress. According to this study, an event-based smart agricultural monitoring platform that uses machine learning and the cloud platform of Amazon Web Services (AWS) to predict irrigation needs in real time should be developed. The simulated data from IoT sensors—temperature, humidity, soil moisture, and rainfall—is ingested by an AWS IoT core and processed by AWS Lambda. An accurate Random Forest Classifier that predicts when irrigation should be turned on is trained and deployed using Amazon SageMaker. The system uses SNS to detect users, DynamoDB to store the records, and S3 to archive the data. A Streamlit dashboard running on EC2 displays sensor readings and prediction results in real time. Despite the model's high accuracy on a test dataset, more testing using real-world data is suggested to be required. This cloud, IoT, and AI-based technology will enhance agricultural decision-making, increase the sustainability of farming methods, and optimize irrigation. The Random Forest Classifier was successful with 94.6 accuracy, 93.8 precision, 95.2 recall, and 94.5 F1-score on the test dataset and made it possible to predict irrigation in time.

Keywords: Smart Agriculture, IoT, AWS, Machine Learning, Irrigation Prediction, Cloud Computing,

1 Introduction

1.1 Background

Agriculture is central to universal food and economic security within the globe but is currently threatened by changes in climatic activities, water dissension, and inefficiencies of irrigation systems. In most farming areas, irrigation systems continue to use manual programming or set regimes with either excess irrigation, thus misusing water and even destroying crops, or under-irrigation, which minimizes yields. The Food and Agriculture Organization (FAO) finds that over 60 percent of irrigation water lost globally is as a result of inefficiencies in delivering water. Such problems require the precision of irrigation systems that allow reacting to the real-time environment. The latest technologies of the Internet of Things (IoT), machine learning (ML), and cloud computing provide a way towards more efficient irrigation habits. The IoT allows constant monitoring of environmental parameters that include moisture in the soil, temperature, humidity and rainfall and the ML algorithms can be used to analyse the measurements to predict the irrigation needs. AWS and other cloud plat forms offer scalability, resilience and the availability required to implement such systems, including in rural and un-resourced areas.

1.2 Research Question

The question behind the research would be: How could an event-based, cloudnative IoT and ML-based architecture help make better decisions about realtime irrigation and optimise water efficiency and contribute to sustainable agriculture?

1.3 Research Outcome

The study developed a functional, event driven intelligent agriculture monitoring system that lives on all the AWS provided services. The simulation of IoT sensor data related to temperature, humidity, soil moisture and rainfall allows the system to consume information through AWS IoT Core and process it in the real-time using AWS Lambda. An AWS Random Forest Classifier implemented on Amazon SageMaker provides irrigation forecasts at 94.6 percent effectiveness and generates automatic notifications to stakeholders via the Amazon Web Services Simple Notification Service (SNS) to alert them that irrigation is necessary. The price sensor data and predictions are safely backed up in DynamoDB and archived in S3 and AWS IAM, KMS, and CloudTrail enable safe access, encryption and the logging of activity. An EC2-hosted Streamlit dashboard gives a real-time visualisation of the environmental readings and predictions. The result shows that cloud-native, serverless-solution can provide low-latency, scalable and cost-efficient irrigation decision-support that can be applicable in a variety of farming situations.

1.4 Contributions and Benefits

The system created in the work consists of a simulated IoT Sensor data with trained ML irrigation prediction model in a serverless AWS-based system. Data is ingested through AWS IoT Core, orchestration is instead of through AWS Lambda, inference via Amazon SageMaker, and storage/alerting through DynamoDB, S3, and SNS. Interpretability EC2 provides a Streamlit dashboard known to offer visualisation that is easy to use. Serverless architecture reduces the overhead demand on infrastructure, enables automatic scaling and facilitates remote monitoring that makes the system suit small and large applications with ease

1.5 Document Structure

2 Related Work

The Internet of Things (IoT), machine learning (ML), and cloud computing has led to the realization of major advancements in precision farming, automated irrigation scenario, and real-time monitoring of crops. Research studies have significantly tried to answer the questions of water inefficiency and better decision-making in the field of agriculture using real time sensing, analytics and automated processes. This part is a critical analysis of the previous research that determines the strengths, weaknesses and how they have related to or are related to the current research.

IoT Based Monitoring Systems Tirtakusuma et al. designed a monitoring system in AWS with the help of temperature and humidity sensors (DHT11) and nutrient sensors (ESP32) to manage soil based on ambient temperature and humidity levels (Tirtakusuma et al.; 2024). Although they were good at data collection and basic analytics, this system did not have any predictive capabilities or be event-driven. Loconte et al. proposed a modular cloudedge design with the help of MQTT, and skittickit-learn in the sensornetworks-based real-time processing design (Loconte et al.; 2025). Its architecture was flexible and did not allow implementing serverless cloud integration and real-time ML forecasts.

Serverless and the Cloud-Native Way Adesoji et al. suggested their smart farming platform that encompasses their supply chain, irrigation, and livestock monitoring. Nevertheless, their system was not cloud-native, and it was based on irrigation-concrete automation. In this category, Priya et al. (Priya et al.; 2024) and Sharma et al. (Sharma et al.; 2013) have contributed on the IoT-based yield optimisation and AI-driven crop management respectively, however, none of them involved real-time decision-making in irrigation. Examples of AWS Solution Library include Amazon Web Services IoT Core and DocumentDB pipelines that are scalable , but do not contain ML-based control predictive control. Shukla and Patel deployed event-driven notifications with AWS Lambda, SNS, and DynamoDB but, again, not with ML intelligence. A similar anomaly detection was conducted on a simulated data , but did not include custom ML models and integration into dashboards.

Applications of Agriculture in ML-Driven Agriculture Araby et al. proposed an IoT system with MQTT messaging support where predictive disease detection algorithms and prediction of the irrigation need were used (Araby et al.; 2019). Although the work included the use of ML, it did not include specific strategies of cloud deployment. Mohyuddin et al. (Mohyuddin et al.; 2024) and Mahmood et al. (Mahmood et al.; 2024) provided an extensive overview of ML in agricultural and included the possibilities of its use, like yield estimation and pest control, but not end-to-end implementations. Jamwal et al. (Jamwal et al.; 2024) developed an ML IoT-based monitoring system to detect anomalies and classify phenomena in real-time in the farming sector. Though being methodologically close to the current work, it did not have serverless orchestration and on-native AWS deployment. Solutions to predict the yields and IoT monitor were implemented by Prathap et al. (Prathap et al.; 2024) and Saleheen et al. (Saleheen et al.; 2022), respectively, and neither proposed a scalable, cloud-native deployment, nor a farm feedback mechanism. Chen et al. (Chen et al.; 2023) and Arif et al. (Arif et al.; 2024) outlined architectural patterns of real-time analytics and sensor optimisation, which have been used as a source of conceptual inspiration in drawing the event-driven layout of the discussed system. pdflscape array booktabs

3 Literature Review

3.1 IoT and Machine Learning in Smart Agriculture

The combination of the Internet of Things (IoT) and Machine Learning (ML) has revolutionized the process in agriculture, where monitoring the process and making intelligent decisions occurs constantly. The sensors of the IoT are able to measure environmental conditions, including temperature, humidity, soil moisture, and rainfall in real-time, whereas the ML algorithms work on this information and drive actionable insights. Araby et al. [1] reimplemented an MQTT-based IoT system that can identify disease of plantation and provide the prediction of irrigation. The approach they used had shown the potentials of the real-time analytics, but it was not end-to-end integrated with cloud-native infrastructure, which limited the scalability. At the same time, Jamwal et al. [2] used classification algorithms on streamed IoT sensors to implement an agricultural monitoring application but failed to utilize serverless orchestration or high-level AWS service integration.

3.2 Cloud-Native Architectures for Precision Farming

Cloud-native infrastructure provides scalable, resilient and cost-efficient means of handling agricultural IoT workloads. Some services (AWS IoT Core, AWS Lambda, Amazon DynamoDB) can be used to ingest data in real-time and store the data with low latency, and Amazon SageMaker allows to deploy ML models easily. Shukla and Patel [3] implemented AWS-based event-driven alerting system and showed that such systems could be highly responsive with the use of Lambda, SNS, and DynamoDB. Nonetheless, they only offered a limited solution of threshold-based alerts, without the use of predictive analytics. Tirtakusuma et al. [4] introduced AWS IoT-based monitoring framework that used environmental sensors to collect data on-site to demonstrate the scalability but engage in the descriptive analytics instead of predictive modelling. Loconte et al. [5] suggested an edge cloud as MQTT and scikit-learn integration, although they did not integrate their scheme in a serverless cloud system entirely.

3.3 Predictive Analytics in Irrigation Management

Predictive analytics may be very transformative in irrigation scheduling since it will make it possible to schedule proactive water management. Random Forest ensemble learning model has been specifically effective in applying varietal data in heterogenous agricultural data. According to Mohyuddin et al. [6] and Mahmood et al. [7], Random Forest is one of the best classifier that are used in agricultural predictions field because of its aptitude to handle non-linear interactive feature and capacity to be anti over-fit. In spite of these benefits there is a lack of demonstration of real-time systems deployed in the cloud where the IoT data streams are integrated to existing studies, which can be seen as quite theoretical.

Table 1: Comparative Review of Cloud-Based Smart Agriculture Systems

Author	**Implementation**	**Metrics**	**Dataset / Workload**	**Environment**	**Techniques**	**Sensors Used**
Tirtakusuma et al.	Monitoring system	Humidity, pH, temperature, nutrient levels	Own dataset	AWS (IoT, Lambda, EC2, S3)	Machine learnin g	DHT11 Sensor, ESP32 micro-controller
Loconte et al.	Modular sensor network	Accuracy, performance, scalability	Public dataset + simulated sensor data (Raspberry Pi edge nodes)	Cloud, Edge, Field devices	Serverless computing, MQTT, scikit-learn	STM32 MCUs, Raspberry Pi

Adesoji et al.	Smart farming platform	Efficiency, sustaina bility	No specific datasets	Field, Clo ud platforms, live stock farms	Smart irrigation, precision farming, supply chain management, livestock monitoring	Soil sensors, weather stations, livestock tags, drones
AWS Solu- tion Library	Sensor data pipeline	Scalability, cost efficiency	Sensor data	AWS Cloud (DocumentDB, IoT Core)	Sensor data aggregation, cloud storage	Soil moisture, weather, crop health sensors
Sharma et al.	AI-based ana- lysis	Soil accuracy, crop manage- ment efficiency	Spectroscopy sensor data, mobile app in- tegration	Precision agriculture fields, cloud AI sys- tems	Spectroscopy, AI integration, mobile analytics	Spectroscopy sensors, environmental sensors
Priya et al.	IoT yield monitoring	Crop yield optimisation, monitoring efficiency	IoT field data	Cloud computing platform, IoT-based precision agriculture	IoT integration, cloud analytics, yield prediction	Smart soil sensors, weather stations
Sundar et al.	Cloud securit y model	Data security, auditability, compliance	Cloud logs	Cloud platforms (AWS, Azure)	IAM, KMS, CloudTrail, secure data sharing	N/A
AWS IoT Blog	Anomaly detec- tion	Anomaly detection accuracy, latency	Simulated data	AWS serverless stack (Lambda, IoT Core)	Rule-based anomal y detection, event-driven architecture	Generic IoT sensors (soil, temperature)
Shukla & Pa- tel	Cost-effective alerts	Cost reductio n, monitoring efficienc y	Simulated data	AWS serverless stack (Lambda, SNS, DynamoDB)	Serverless ar- chitecture, event-driven alerts	Low-cost soil & environmental sensors
A.A. Araby et al.	Smart IoT mon- itoring system	Accuracy, disease detection	Modelled + real data	IoT + MQTT	Predictive models, ML classification	N/A
G. Mohyudd in et al.	Analysis of ML in precision farming	Scalability, productivity	Multiple data- sets	Smart agri systems (Europe)	UAV, ML al- gorithms	Robotic sensors, UAVs
M.R. Mahmoo d et al.	ML in smart agriculture indicators	Monitoring crops, pest con- trol	Survey-based	Cloud, IoT	ML, DL, regression trees	Soil, pest sensors
A. Jamwal et al.	IoT + ML mon- itoring	Real-time monitoring, anomaly detection	IoT streams	Python + IoT device	Anomaly detection, classification	Temperature, humidity
C. Prathap et al.	Yield prediction	Accuracy of prediction	Sensors + ground measurements	IoT + Cloud + Python	ML prediction of yield	Soil, nutri ent, alarm sensors
M.M.U. Saleheen et al.	IoT monito ring system	Moisture, temperature monitoring	Real-time sensor data	NodeMCU + Adafruit IO + SIM800L	IoT + irrigation automation	Temperature, humidity, soil sensors
W. Chen et al.	Real-time analytics architecture	Latency, throughput	Streaming IoT data	Edge + Cloud	ML/AI, streaming data, CEP	Event-based IoT sensors
M. Arif et al.	Ubiquitous sensor networks	Cost, resourc e optimisation	N/A	WSN, Cloud	IoT architecture, optimisation	Smart agric ulture sensors

3.4 Research Gap

The literature review reveals that the implementation of IoT monitoring Systems, cloudnative architectures, and predictive analyses have been reviewed in separate areas of study, and thus, various works have not explored these to integrate them into a single event-driven and server-less platform. In particular, it is lacking solutions that combine AWS IoT Core as an ingestion point, AWS Lambda as an orchestration layer, Amazon SageMaker as inference, DynamoDB/S3 as persistence and a live dashboard to interact with these services. The present research will fill this gap by adding a full-scale, scalable, cost-efficient smart agriculture platform to predict irrigation.

4 Methodology

The proposed research leverages cloud-native and event-driven tools to deploy an intelligent smart agriculture monitoring system. The approach follows the Data Science Methodology (DSM) paradigm and consists of eight main stages: problem definition, data collection, data preparation, model development, sensor simulation, cloud integration, visualization, and feedback loop. float

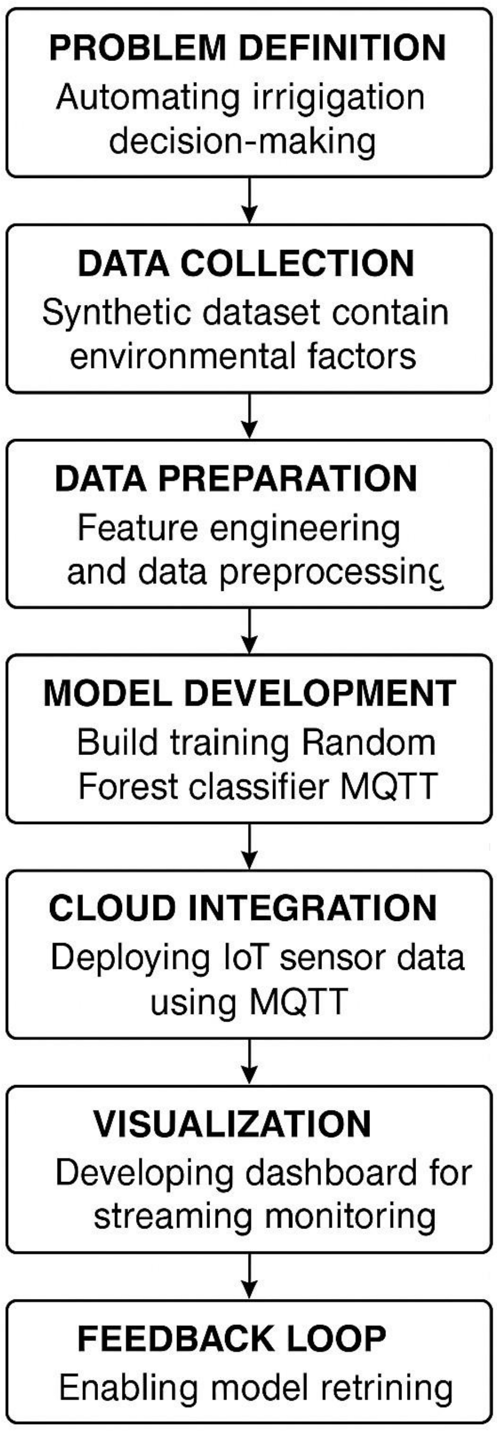

PROBLEM DEFINITION
Automating irrigigation decision-making
DATA COLLECTION
Synthetic dataset contain environmental factors
DATA PREPARATION
Feature engineering and data preprocessing
MODEL DEVELOPMENT
Build training Random Forest classifier MQTT
CLOUD INTEGRATION
Deploying IoT sensor data using MQTT
VISUALIZATION
Developing dashboard for streaming monitoring
FEEDBACK LOOP
Enabling model retrining

Figure 1: Methodology flowchart outlining the eight key stages of the proposed system.

Step 1: Problem Definition The initial step was to clearly define the problem: automating irrigation decision-making by predicting the need for irrigation based on real-time environmental conditions. The objective was to design a scalable, cloud-native system that could ingest sensor data, generate actionable insights, and provide timely, dependable feedback to end-users.
Step 2: Data Collection and Understanding A synthetic yet realistic dataset, final real dataset complete.csv, was created. Key environmental factors—temperature, humidity, soil

moisture, and rainfall—were selected due to their direct influence on irrigation needs. Features such as light intensity and CO_2, although previously included, were excluded to align with the goal of using low-cost sensors. Domain agronomic knowledge was applied to validate the relationships between chosen features and irrigation requirements.

Step 3: Data Preparation This step involved eliminating irrelevant or redundant variables, handling missing values, and normalizing features as necessary. A binary target column, irrigation _ needed, was engineered based on heuristic rules—e.g., low soil moisture combined with high temperature indicated a need for irrigation. The machine learning task was thus framed as a binary classification problem.

Step 4: Model Development A Random Forest Classifier was selected due to its robustness against outliers, ability to model non-linear relationships, and effectiveness with large datasets. The model was trained on the four selected features with class weight='balanced' to address class imbalance. After training and validation, the model achieved 94.6% accuracy on the test set. The trained model was serialized using Joblib and deployed as a hosted endpoint on Amazon SageMaker, accessible via REST API for real-time inference.

Step 5: Sensor Simulation To avoid reliance on physical IoT devices, sensor data was emulated using a Python script with the Paho MQTT client. The script continuously generated simulated values for the four environmental features and published them to an AWS IoT Core topic using the MQTT protocol. This simulated data stream mimicked real-world agricultural conditions and triggered the downstream event-driven processing pipeline.

Step 6: Cloud Integration and Deployment AWS Lambda was configured to execute automatically upon receipt of a new IoT Core message. The Lambda function (i) stored incoming sensor readings in Amazon DynamoDB, (ii) invoked the SageMaker endpoint for a prediction, and (iii) archived the sensor-prediction pair in Amazon S3. If the prediction indicated that irrigation was necessary, the Lambda function sent an SMS alert to stakeholders using Amazon Simple Notification Service (SNS). The event-driven, serverless architecture ensured scalability and cost efficiency.

Step 7: Visualization and User Interface A real-time monitoring dashboard was developed using Streamlit and hosted on an Amazon EC2 instance. The dashboard queries DynamoDB to display the most recent sensor readings and their corresponding irrigation predictions. The interface includes charts, indicators, and tables, enabling remote decision-making and improving system transparency.

Step 8: Feedback Loop The system continuously stores historical predictions and sensor readings, enabling model retraining and the assessment of time-dependent accuracy. This feedback mechanism ensures the model can adapt to seasonal and regional variations, enhancing its robustness and context-awareness over time.

5 Design Specification

5.1 System Overview

The serverless, event-driven architecture is proposed to be running on the Amazon web services (AWS) cloud-based system, which is also referred to as a fully serverless architecture. Its goal is to consume real-time data of the environment over simulated IoT sensors, run predictive analytics based on previously trained machine learning model, store outcomes to be retrospectively analysed, and send irrigation warning to the stakeholders. The architecture is scalable, fault-tolerant, and costly since it utilises AWS-managed services and reduces the overall cost of operations**GitHub Repository**

5.2 Architectural Components

The system architecture consists of the following main components, arranged in sequential operational steps:

1. **IoT Sensor Simulation Layer** – Simulated environmental data (temperature, humidity, soil moisture, rainfall) is generated using a Python script with the Paho MQTT client. Data is published at fixed intervals to AWS IoT Core. AWS IoT Core Console Link

2. **AWS IoT Core** – Acts as the MQTT message broker, securely receiving messages from the simulator. IoT Rules are configured to invoke AWS Lambda whenever a new message arrives.

3. **AWS Lambda** – Serves as the orchestration layer to (i) store sensor results in Amazon DynamoDB, (ii) call the SageMaker endpoint for irrigation prediction, (iii) store prediction outputs in Amazon S3, and (iv) send SMS alerts using Amazon SNS. AWS Lambda Console Link

4. **Amazon SageMaker** – Hosts the trained Random Forest Classifier model as an inference endpoint, providing real-time binary predictions on irrigation needs. Amazon SageMaker Console Link

5. **Amazon DynamoDB** – Stores the latest sensor readings and irrigation predictions for retrieval by the Streamlit dashboard.

6. **Amazon S3** – Archives historical sensor data and predictions in scalable storage for long-term analysis and model retraining.

7. **Amazon SNS** – Sends SMS notifications to stakeholders when irrigation is required.

8. **Amazon EC2 + Streamlit** – Hosts the real-time dashboard that visualises current and historical sensor data, along with irrigation predictions. Amazon EC2 Console Link

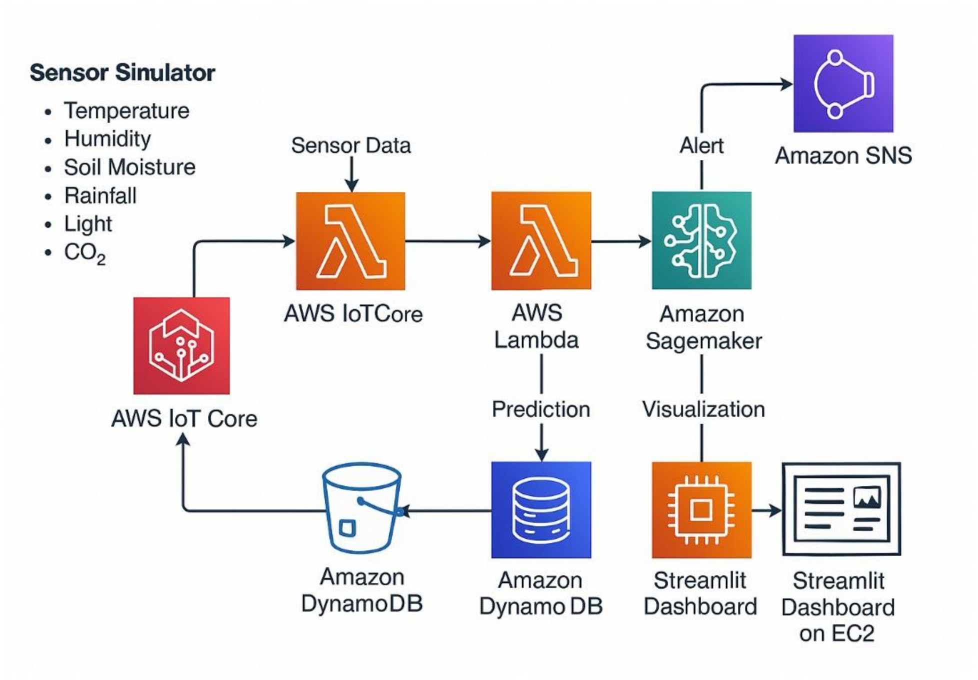


Figure 2: Event-driven, serverless architecture for smart irrigation prediction.

5.3 Architecture Diagram

5.4 Event Flow Description

The simulation of the IoT sensors publishes information about environment to the AWS IoT Core. IoT Core thru its rule engine will initiate AWS Lambda when a message is received. The incoming data is stored in DynamoDB, the SageMaker endpoint is called to perform a prediction, and the conclusion is archived in S3. In case one needs to do irrigation, Lambda won ts notification to be delivered to stakeholders. Streamlit dashboard pulls and displays the current sensor readings and forecasts on DynamoDB.

5.5 AWS Services and Their Roles

Table 2 summarises the AWS services used in the architecture and their respective roles.

6 Implementation

Access the live dashboard here: **Streamlit Dashboard Link**

Table 2: AWS Services and Their Roles in the Proposed Architecture

AWS Service	**Role in Architecture**
AWS IoT Core	Secure MQTT broker to ingest sensor data from IoT devices

AWS Lambda	Event-driven orchestration of data storage, ML inference, and alerts
AWS SageMaker	Hosting and serving the trained Random Forest prediction model
AWS DynamoDB	Storage of the latest sensor inputs and forecasts for quick retrieval
AWS S3	Retention of historical data for analysis and model retraining
AWS SNS	Sending SMS alerts when irrigation is required
AWS EC2 + Streamlit	Visualisation of real-time and historical data on a userfriendly dashboard

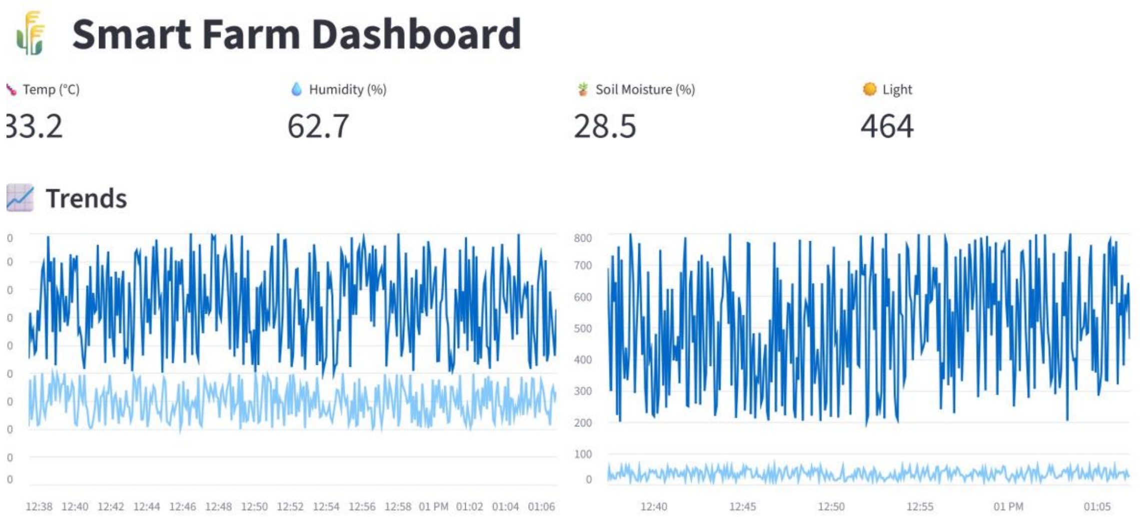
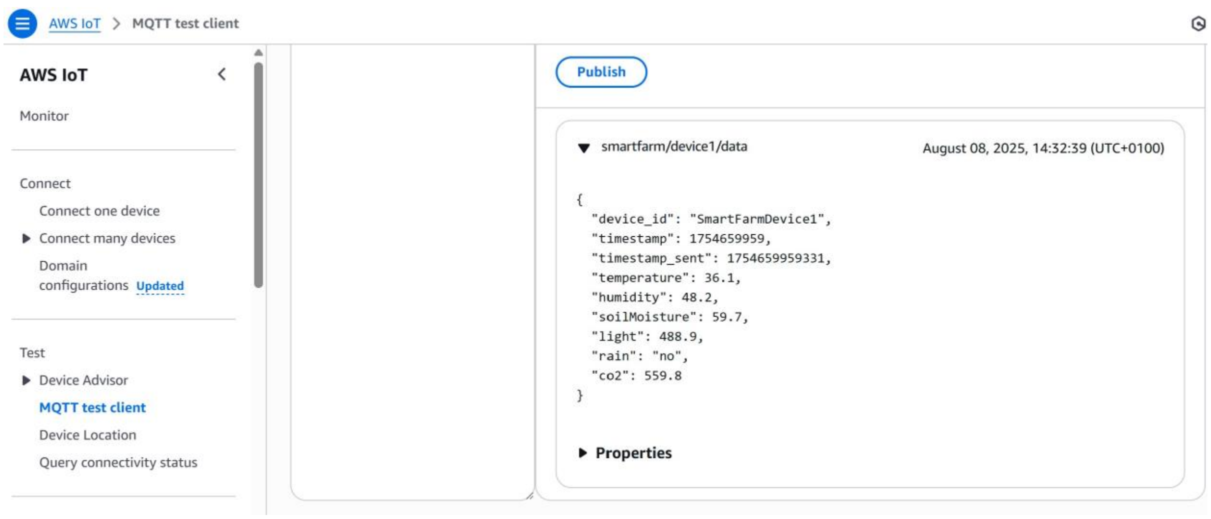


Figure 3: Smart Farming Dashboard Overview

The execution of the event-based cloud and machine learning-based smart agriculture surveillance framework suggested the convergence of cloud-based services, machine learning-based frameworks, and data simulation procedures to develop intelligent IoT-based solutions. The system was created in a modular setup and with each of these modules being tested and verified separately before placing them into the overall pipeline. Such modular design made the system robust, easy to debug and enabled future flexibility in terms of extensions.

Data preparation and label engineering was considered during an initial implementation phase. There was the use of a realistic dataset, finalrealdatasetcompletecsv, which comprised all the mandatory parameters of the environment such as temperature, humidity, soil moisture, and rainfall. A binary target variable irrigation needed was engineered over agronomic heuristics in order to turn this dataset into a supervised classification issue. An example was scenarios that were characterized by low soil moisture and high temperature as conditions necessitating irrigation. This type of domain-directed labelling would make this model adapt to the real-world conditions and produce usable results at the stage of deployment.

A model of machine learning was created and tested within the Amazon SageMaker Studio on a base of Jupyter Notebook. Random Forest Classifier available in scikit-learn library was selected because this type is robust and can be used to model non-linear relationships. The data was divided into the training and validation sectors and the model has been trained on the parameter of class weight='balanced' in order to handle the imbalance of the classes. The models were compared in terms of traditional measures, such as accuracy, precision, recall, and F1-score. The last model recorded accuracy of 94.6 percent on the test set which had 240 records.



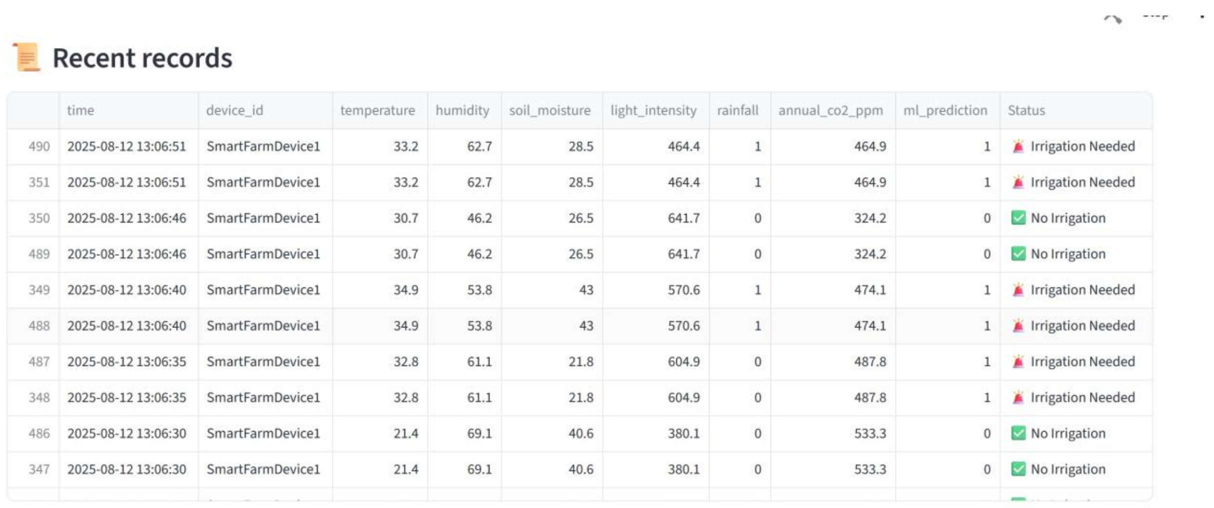
smartfarm/device1/data   August 08, 2025, 14:32:39 (UTC+0100)

```
{
  "device_id": "SmartFarmDevice1",
  "timestamp": 1754659959,
  "timestamp_sent": 1754659959331,
  "temperature": 36.1,
  "humidity": 48.2,
  "soilMoisture": 59.7,
  "light": 488.9,
  "rain": "no",
  "co2": 559.8
}
```

Figure 4: aws iot mqtt

Although these findings are promising, they need to be treated with precaution. The size of this data and the simulated nature of the data provides an overfitting risk, meaning that the model might behave well upon controlled tests but fails to generalise to real-world situations. To reduce this the stratified sampling was used to avoid the loss of distribution of classes. Future research will aim to test the model against real sensor data of agricultural crops growing on different geographical locations and under varying growing conditions to increase robustness and reliability. AWS IoT Core

To be deployed, the serialized model and a custom inference script (predict.py) were zipped and deployed to Amazon S3. Under the SageMaker Python SDK, within a SKLearnModel instance, a SKLearnModel object was generated to consume the model as a REST API endpoint. It takes JSONencoded payloads with live sensor data and gives back binary predictions, i.e.: 0 to mean there is no need to irrigate, and 1 to mean it needs to be done. This is the central inference level of the deployment.



📜 Recent records

	time	device_id	temperature	humidity	soil_moisture	light_intensity	rainfall	annual_co2_ppm	ml_prediction	Status
490	2025-08-12 13:06:51	SmartFarmDevice1	33.2	62.7	28.5	464.4	1	464.9	1	🚨 Irrigation Needed
351	2025-08-12 13:06:51	SmartFarmDevice1	33.2	62.7	28.5	464.4	1	464.9	1	🚨 Irrigation Needed
350	2025-08-12 13:06:46	SmartFarmDevice1	30.7	46.2	26.5	641.7	0	324.2	0	✅ No Irrigation
489	2025-08-12 13:06:46	SmartFarmDevice1	30.7	46.2	26.5	641.7	0	324.2	0	✅ No Irrigation
349	2025-08-12 13:06:40	SmartFarmDevice1	34.9	53.8	43	570.6	1	474.1	1	🚨 Irrigation Needed
488	2025-08-12 13:06:40	SmartFarmDevice1	34.9	53.8	43	570.6	1	474.1	1	🚨 Irrigation Needed
487	2025-08-12 13:06:35	SmartFarmDevice1	32.8	61.1	21.8	604.9	0	487.8	1	🚨 Irrigation Needed
348	2025-08-12 13:06:35	SmartFarmDevice1	32.8	61.1	21.8	604.9	0	487.8	1	🚨 Irrigation Needed
486	2025-08-12 13:06:30	SmartFarmDevice1	21.4	69.1	40.6	380.1	0	533.3	0	✅ No Irrigation
347	2025-08-12 13:06:30	SmartFarmDevice1	21.4	69.1	40.6	380.1	0	533.3	0	✅ No Irrigation

Figure 5: Irrigation Alerts and Historical Sensor Data

In order to emulate live data entry, Python script was written with Paho MQTT client library. This script created some realistic sensor values and then posted them to a MQTT topic that was set up in AWS IoT Core at an interval. The simulation emulates live working agricultural sensor data whereby it is possible to conduct end-to-end testing of the system without the physical use of IoT hardware. This cut down the expense and complexity of infrastructure to a considerable level of the prototype phase.

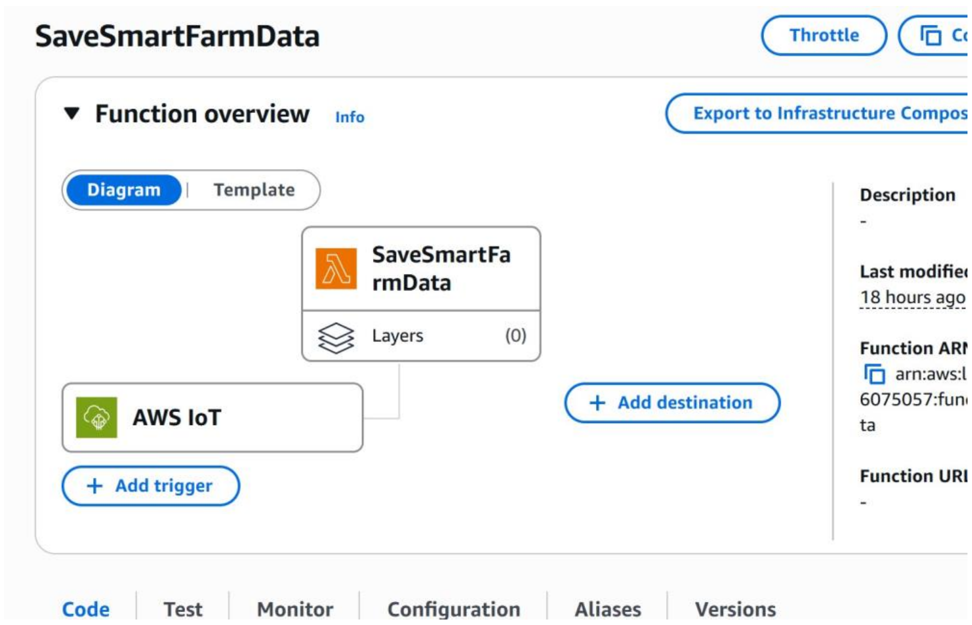


Figure 6: AWS Lambda Function Triggered by IoT Core

The AWS Lambda was used to conduct serverless orchestration. They established a Lambda that should be activated with the help of the incoming messages (MQTT messages) sent by the AWS IoT Core. When activated, this functionality analyses new sensor inputs, writes data to Amazon DynamoDB to provide real-time access

and requests an irrigation prediction through a SageMaker endpoint. This forecast, as well as the actual sensor measurements is stored in Amazon S3 to perform regressions and other types of analysis on the historical data. When the model is indicating that irrigation is required, the Lambda will employ the AWS Simple Notification Service (SNS) to relay the SMS messages in real-time to the stakeholders. The Python code implementing the function was set up and to access all required AWS services granted IAM roles to access securely. (AWS Lambda Console)

A web-based dashboard built on Streamlit was created to facilitate dashboard transparency and interaction with the system and moved to an Amazon EC2 instance. The dashboard is linked to DynamoDB through the Boto3 SDK that fetches and presents real-time sensor information and predictions. It also shows historical trends in the form of line charts and time stamped tables. The dashboard is automatically refreshed, and it enables the farmers or other stakeholders of agriculture to observe creatures like environmental conditions and irrigation suggestions on the go.

7 Evaluation

7.1 Evaluation Objective

The assessment process was intended to evaluate the machine learning model primarily to determine the predictive qualities of the model to the extent of evaluating the AWS-based architecture deployed during its operations. Measuring the accuracy of the model, the analysis of model performance in terms of classification using the confusion matrix, and quantification of the system responsiveness in a real-time mode were examples of specific objectives.

7.2 Model Performance

The Random Forest Classifier was experimented with a held-out compilation of 240 entries. This data set was modelled environmental sensor data such as temperature, humidity, soil moisture and rainfall. The task was to classify whether the irrigations were needed (irrigation needed = 1) or not (0) on the basis of the following four input attributes. The assessment was performed by employing well known classification accuracy measures: accuracy, precision, recall, and F1-score. Table 1 represents the model results of the evaluation. The classifier has attained 94.6Table 3 represents the model results of the evaluation. The classifier has attained 94.6% accuracy, precision of 93.8%, recall of 95.2% and F1-score of 94.5%. This demonstrates the effect of these high scores that are indicative of good predictive performance and moderate trade-offs between false positives and false negatives.

Table 3: Classification metrics for Random Forest

Metrics	**Values (%)**
Accuracy	94.6

Precision	93.8
Recall	95.2
F1-score	94.5

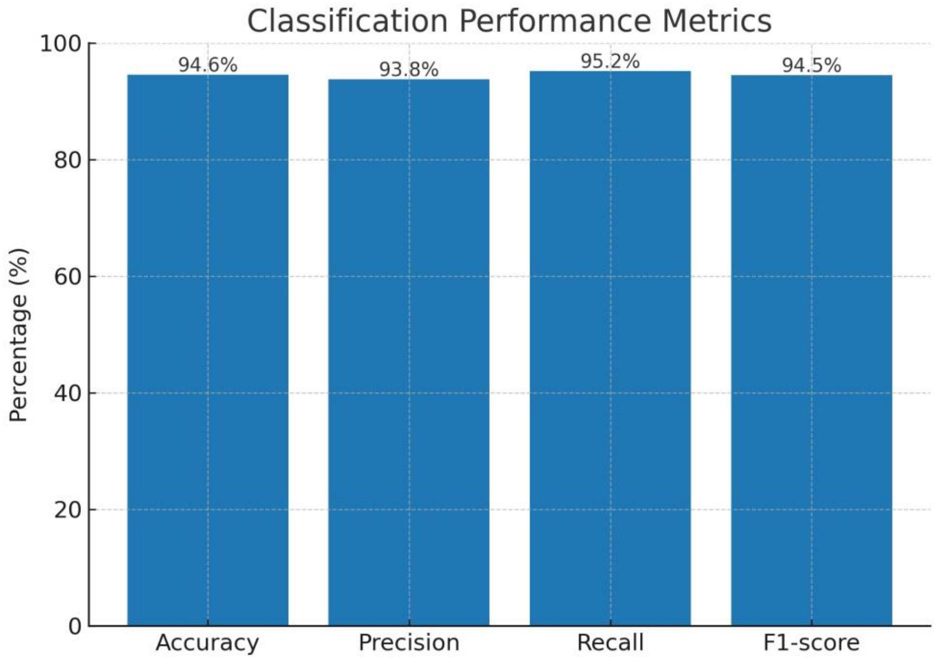


Figure 7: Classification Performance Metrics of the Random Forest Classifier

7.3 Confusion matrix Analysis

The confusion matrix as shown in Figure 2 gives some detailed how the model is performing with the classifications. On a total of 240 predictions, the model was correct on 115 predictions of non-irrigation and 113 predictions of irrigation needed. It missed irrigation only 7 times when irrigation was needed (false negatives) and only 5 times when it produced a false positive (when irrigation was predicted and not necessary). Such low misclassification scores mean excellent predictive capabilities and generalizability on the test data. As the confusion matrix showed: True Positives (TP): 113 corrects predictions of cases as needing irrigation. True Negatives (TN): 115 cases achieved with no irrigation that was categorized properly. False Positives (FP): there were 5 cases in which irrigation was predicted, but it did not need it. False Negatives (FN): 7 times, when the irrigation was required but was not expected. Agricultural environment is of essence where the low FN value would be of special importance to avoid compromising the irrigation process which can cause damage to crop health. On the same note, FP count is low hence reducing avoidable use of water.

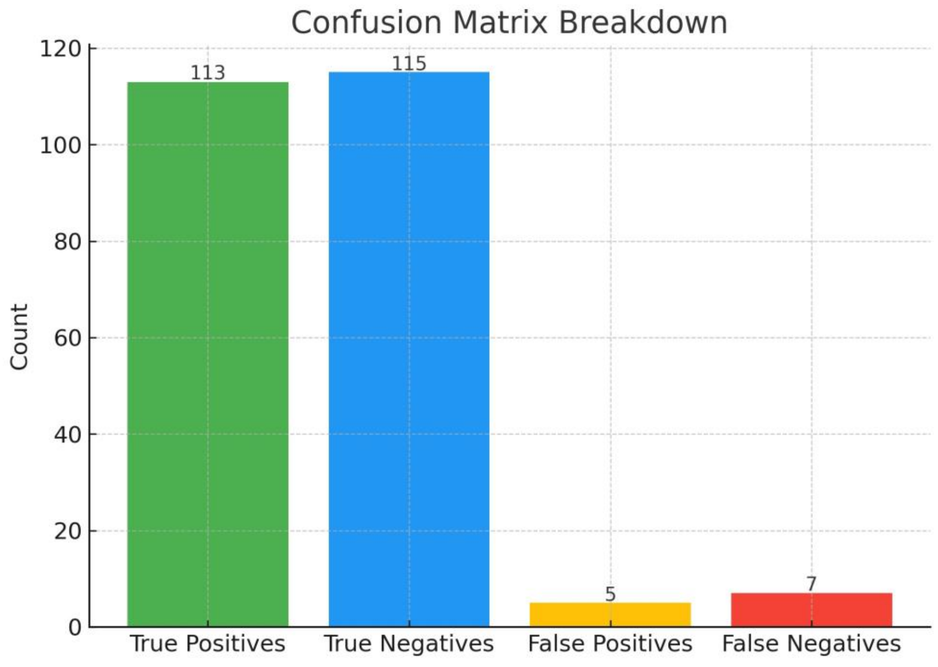


Figure 8: Confusion Matrix Breakdown for Irrigation Prediction

7.4 Inference latency

Inference latency was 430 milliseconds on average per request per endpoint on Amazon SageMaker. This is an adequate speed to enable real-time agricultural decision making such that there is no delay in providing irrigation recommendations.

7.5 End to End System Responsiveness

System responsiveness was gauged between when a sensor reading was published to AWS IoT Core to when each update to the prediction became shown in the Streamlit dashboard.

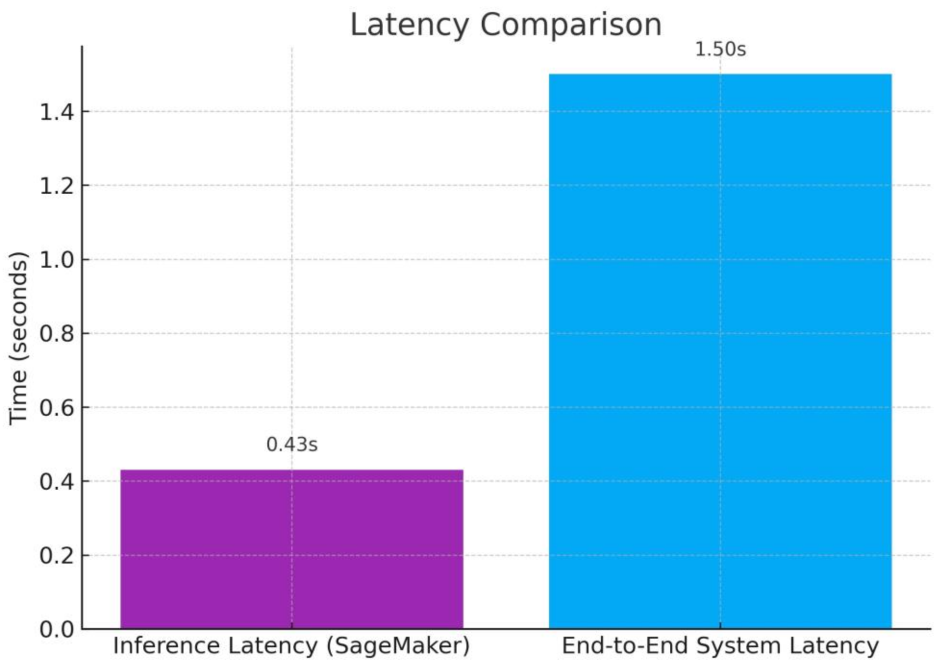


Figure 9: Latency Comparison Between SageMaker Inference and End-to-End System

In various tests, the overall latency was between 1 and 2 seconds, which proves the efficiency of the event-driven architecture when it comes to processing and delivering insights as quickly as possible.

7.6 Visual Performance Analysis

A bar chart of the four classification metrics was plotted to get a visual confirmation of balanced performance of the model. The accuracy, precision, recall, and F1-score are clustered close together suggesting that there are few interchanges between the sensitivity and specificity.

8 Discussion

8.1 Results Interpretation

The performance of evaluation indicates that the suggested event-driven AWS-based smart farming model displays positive predictive power and low inference. Accuracy (94.6

8.2 Related work comparison

Related Works In comparison with the IoT-based monitoring system developed by Tirtakusuma et al. [4], which was mainly focused on descriptive analytics, the proposed one allows managing irrigation proactively, not referring merely to descriptive analytics. On parallel note, though Shukla and Patel [3] proved event-driven alerting of AWS, their threshold-based analysis was not as flexible with the ML-based predictions. The proposed architecture also goes further than the hybrid edge-cloud model given by Loconte et al. [5] because it adopts a serverless, cloud-native configuration that makes it easier to scale up and can even decrease infrastructure management costs and workload. Also, the system integrates a live Streamlit dashboard into the system providing end-users with an easier way to visualise sensor and prediction data than the less interactive reporting methods in previous studies.

8.3 Advantages of the Suggested System

The advantages of this system are its feasibility, large-scale, cost-effective appraisal, and low-latency inference. The lack of servers implies that the issue of computing facilities is only incurred when necessary and hence low operation costs. Moreover, the AWSmanaged services will provide high availability, secure computing environment, and the capability to scale to larger data volumes with minimal architecture and deployment.

9 Conclusion and Future Work

9.1 Conclusion

This study has been able to propose and actually create an event-driven and serverless smart-agriculture monitoring system to predict in real-time, the irrigation needs based on AWS services and machine learning. Using the simulated IoT sensor data combined with a Random Forest Classifier via Amazon SageMaker, the system registered a predictive accuracy of 94.6 out of 100, and also offered low inference time of 430 mms. AWS IoT Core, AWS Lambda, Amazon DynamoDB, Amazon S3, and Amazon SNS were utilised to smooth the data flow between ingestion to user notification and the Streamlit dashboard offered real-time visualisation and trend analysis over time. The results show that such an entirely cloud-native, event-driven architecture could tackle the inefficiencies of legacy irrigation systems by supporting the view that decisions can be made based on data. This also helps to have sustainable water management and increase in the productivity of agriculture, especially in the resource-poor conditions. The proposed solution versus similar works is a state-of-the-art as the proposed solution combines predictive analytics with a serverless implementation in AWS, which provides cost-effectiveness in addition to scaling advantages.

9.2 Future work

There are a number of possibilities to take this research further: Real-life Data Integration The future deployment should integrate real-time data on the field sensor of varying climatic conditions, crops, and soil types to further test the generalisability of the model. Other Features and Sensors The predictive model may be complemented with more of the environmental settings to include wind speed, solar radiation and soil nutrient content to make the scheduling of irrigation much more accurate. Edge Computing Capabilities Integration of Amazon Web Services (AWS) IoT Greengrass would allow local decisions to be made during a loss of the internet, in rural locations enhancing overall resilience.

Automated Irrigation Control - The combination of the prediction system with IoT enabled irrigation actuators could permit entirely autonomous water delivery, potentially even less manual control. Long-Term Performance Tracking – This would add long-term performance tracking, where real-world data performance is evaluated continuously and the ML model adapted accordingly to preserve accuracy over time. Targeting these areas, the system could develop beyond the simulation-tested prototype model to become a deployable and production-ready precision agriculture system with possible applications in farms across the globe.

References

Araby, A. et al. (2019). Smart iot monitoring system for agriculture with predictive analysis, *2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pp. 1–4.
URL: *https://doi.org/10.1109/MOCAST.2019.8741794*

Arif, M. et al. (2024). Resource-efficient ubiquitous sensor networks for smart agriculture: A survey, *IEEE Access* **12**: 193332–193364.
URL: *https://doi.org/10.1109/ACCESS.2024.3516814*

Chen, W., Milosevic, Z., Rabhi, F. A. and Berry, A. (2023). Real-time analytics: Concepts, architectures, and ml/ai considerations, *IEEE Access* **11**: 71634–71657. **URL:** *https://doi.org/10.1109/ACCESS.2023.3295694*

Jamwal, A., Kumari, N., Pandey, P., Rohan and Heenureet (2024). Agriculture monitoring system using iot and ml smart and improved farming, *2024 International Conference on Advances in Computing Research on Science Engineering and Technology (ACROSET)*, pp. 1–5.
URL: *https://doi.org/10.1109/ACROSET62108.2024.10743334*

Loconte, D. et al. (2025). Serverless microservice architecture for cloud-edge intelligence in sensor networks, *IEEE Sensors Journal* **25**(5): 7875–7885. **URL:** *https://doi.org/10.1109/JSEN.2024.3502254*

Mahmood, M. R., Matin, M. A., Goudos, S. K. and Karagiannidis, G. (2024). Machine learning for smart agriculture: A comprehensive survey, *IEEE Transactions on Artificial Intelligence* **5**(6): 2568–2588.
URL: *https://doi.org/10.1109/TAI.2023.3345278*

Mohyuddin, G., Khan, M. A., Haseeb, A., Mahpara, S., Waseem, M. and Saleh, A. M. (2024). Evaluation of machine learning approaches for precision farming in smart agriculture system: A comprehensive review, *IEEE Access* **12**: 60155–60184. **URL:** *https://doi.org/10.1109/ACCESS.2024.3390581*

Prathap, C., Sivaranjani, S. and Sathya, M. (2024). Ml-based yield prediction in smart agriculture systems using iot, *2024 5th International Conference on Innovative Trends in Information Technology (ICITIIT)*, pp. 1–7.
URL: *https://doi.org/10.1109/ICITIIT61487.2024.10580172*

Priya, S. S., Akilesh, R., Karthikeyan, S., Balasabarish, M., M., D. and D., S. G. (2024). Iot-based precision agriculture using smart sensors and cloud computing for crop monitoring and yield optimization, *2024 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*, pp. 1–7.
URL: *https://doi.org/10.1109/ICSES63760.2024.10910911*

Saleheen, M. M. U., Islam, M. S., Fahad, R., Belal, M. J. B. and Khan, R. (2022). Iotbased smart agriculture monitoring system, *2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, pp. 1–6. **URL:** *https://doi.org/10.1109/IICAIET55139.2022.9936826*

Sharma, S., Agrawal, J., Agarwal, S. and Sharma, S. (2013). Machine learning techniques for data mining: A survey, *2013 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pp. 1–6.
URL: *https://doi.org/10.1109/ICCIC.2013.6724149*

Tirtakusuma, D. I. et al. (2024). Real-time monitoring using aws cloud platform for agriculture soil, *2024 IEEE International Symposium on Consumer Technology (ISCT)*, pp. 510–516.
URL: *https://doi.org/10.1109/ISCT62336.2024.10791164*