PERIÓDICO TCHÊ QUÍMICA
ARTIGO ORIGINAL

UM SISTEMA AUTOMATIZADO DE MONITORAMENTO E IRRIGAÇÃO DE CULTURAS COM ANÁLISE PREDITIVA

AN AUTOMATED CROP MONITORING AND IRRIGATION SYSTEM WITH PREDICTIVE ANALYSIS

Sangeeta Borkakoty
University of Science & Technology Meghalaya, Department of Computer Science, India

Dr. Atowar ul Islam *
University of Science & Technology Meghalaya, Department of Computer Science, India

Devaj Neogi
University of Science & Technology Meghalaya, Department of Computer Science, India

Nilotpal Deka
University of Science & Technology Meghalaya, Department of Computer Science, India

** Corresponding author*
e-mail: atowar91626@gmail.com

Received on December 10, 2023; received as revised form June 01, 2024; accepted June 12, 2024.

RESUMO

Introdução: O aumento da população global e a diminuição das terras agrícolas desafiam o abastecimento de alimentos, agravado pela degradação da terra e escassez de água. A agricultura de precisão, usando soluções de IoT, oferece uma promessa para abordar essas questões, aumentando a produtividade, a lucratividade e a sustentabilidade ambiental. O artigo propõe um sistema automatizado de monitoramento e irrigação baseado em IoT com análise preditiva como uma solução. **Objetivos**: Desenvolver um modelo de irrigação baseado em IoT com computação em nuvem, um aplicativo móvel para monitoramento remoto e controle de irrigação, e um método de análise preditiva para previsão das condições climáticas. **Métodos**: Sensores IoT conectados a um Node MCU monitoram em tempo real a temperatura, umidade e umidade do solo. Uma bomba de água ajusta a irrigação de acordo. Os dados são enviados para o ThingSpeak para visualização e para o Firebase para armazenamento, acessíveis por meio de um aplicativo móvel. A análise preditiva combina dados de sensores e dados históricos do clima usando o algoritmo Random Forest para otimizar a irrigação. Isso determina a frequência, duração e tempo ideais de irrigação com base nos níveis previstos de temperatura e umidade do solo, garantindo a umidade ideal do solo para o crescimento das plantas. **Resultados**: O sistema exibiu resultados encorajadores. Aproveitando IoT, computação em nuvem e análise preditiva, ele mostrou potencial transformador. Oferecendo monitoramento preciso da umidade do solo, acesso instantâneo a dados e conselhos personalizados, o sistema pode elevar o gerenciamento de culturas, agilizar o consumo de água e aumentar a produtividade entre os agricultores indianos. **Discussão**: O sistema automatizado de monitoramento e irrigação de culturas tem potencial transformador na agricultura. Alguns escopos e aprimoramentos futuros podem incluir a ampliação da compatibilidade de culturas, integração de dados de satélite, aprimoramento do monitoramento de pragas, melhoria da conectividade, enriquecimento do aplicativo móvel, expansão por meio de parcerias e refinamento dos modelos preditivos. **Conclusões**: O sistema de monitoramento de culturas baseado em IoT revoluciona a agricultura indiana, otimizando a irrigação e integrando previsões climáticas. Com aprimoramentos de aplicativos móveis e melhorias futuras, ele promete uma agricultura sustentável e o empoderamento dos agricultores.

Palavras-chave: *Internet das Coisas, sensores IOT, random forest, aprendizado de máquina, análise preditiva.*

ABSTRACT

Background: The rising global population and decreasing agricultural land challenge the food supply, worsened by land degradation and water scarcity. Using IoT solutions, precision farming offers promise for addressing these issues by enhancing productivity, profitability, and environmental sustainability. The paper proposes an IoT-based automated monitoring and irrigation system with predictive analysis as a solution. **Aims:** To develop an IoT-based

DOI: 10.52571/PTQ.v21.n47.2024_03_ATOWAR_pg_24_41.pdf.

irrigation model with cloud computing, a mobile app for remote monitoring and irrigation control, and a predictive analysis method for forecasting weather conditions. **Methods:** IoT sensors are connected to a node MCU, and they monitor real-time temperature, humidity, and soil moisture. A water pump adjusts irrigation accordingly. Data is sent to ThingSpeak for visualization and Firebase for storage and is accessible via a mobile app. Predictive analysis combines sensor and historical weather data using the Random Forest algorithm to optimize irrigation. This determines the ideal irrigation frequency, duration, and timing based on predicted temperature and soil moisture levels, ensuring optimal soil moisture for plant growth. **Results:** The system exhibited encouraging outcomes. It displayed transformative potential by harnessing IoT, cloud computing, and predictive analytics. Offering precise soil moisture monitoring, instant data access, and tailored advice, the system can elevate crop management, streamline water consumption, and boost productivity among Indian farmers. **Discussion:** The automated crop monitoring and irrigation system has transformative potential in farming. Some future scope and enhancements may include broadening crop compatibility, integrating satellite data, enhancing pest monitoring, improving connectivity, enriching the mobile app, scaling up through partnerships, and refining the predictive models. **Conclusions:** The IoT-driven crop monitoring system revolutionizes Indian agriculture by optimizing irrigation and integrating weather forecasts. With mobile app enhancements and future improvements, it promises sustainable farming and empowerment for farmers.

Keywords: *Internet of Things, IOT sensors, random forest, machine learning, predictive analysis.*

1. INTRODUCTION:

The world population is growing exponentially. The Food and Agriculture Organization predicts that food and feed production will need to increase by 70% by 2050 to meet global demand ("The state of the world's land and water resources for food and agriculture," n.d.). However, agricultural land is shrinking, and natural resources are being depleted, creating a critical need to improve agricultural productivity (Kumar et al., 2008). The application of IoT (Internet of Things) solutions in agriculture, known as precision farming, can help bridge the gap between supply and demand, ensuring high productivity, profitability, and environmental protection (Verdouw et al., 2016). The adoption of IoT devices in agriculture is expected to reach 75 million by 2020, with the global smart agriculture market tripling by 2025 (Adesta et al., 2017).

IoT technology enables farmers to reduce waste, improve yields, and make efficient use of resources such as water and electricity. Smart farming solutions use sensors to monitor crop fields, providing data on humidity, soil moisture, light, temperature, and crop health. Farmers can remotely monitor the field status and take necessary actions based on the data, such as initiating smart irrigation when soil moisture drops. These IoT solutions provide better control over livestock raising and growing processes, making them more predictable and efficient.

The challenges facing agricultural growth include population growth, limited arable land, access to water, and changing dietary patterns. Rapid population growth, uneven land distribution, and land degradation hinder efforts to increase food production and meet basic needs. Rising incomes and dietary changes increase global food demand, necessitating increased agricultural output. However, factors such as climate change, underinvestment, and urbanization pose challenges to agricultural production.

Total factor productivity (TFP) (Le Mouël & Forslund, 2017) measures agricultural productivity that considers inputs such as land, labor, fertilizer, and machinery. Improving TFP through advanced technology, improved seed varieties, mechanization, and efficient resource use is crucial to sustainably meeting the needs of a growing population. However, TFP growth is not keeping pace with targets, and low-income countries rely on changing land use, which leads to deforestation and land degradation.

The world has lost a third of arable land in the past 40 years due to erosion and pollution (Quinton & Fiener, 2024). Continuous tillage, heavy fertilizer use, and deforestation (Lawrence & Vandecar, 2015) have degraded soils and affected their ability to store water. Soil erosion is occurring at a rate much faster than soil formation, posing a threat to agricultural productivity (Wu et al., 2018). As populations increase, per-capita farmland is decreasing, exacerbating the challenge of meeting food demand.

Water scarcity is another significant concern for agriculture (Seckler et al., 1999). Irrigated agriculture is more productive than rain-fed agriculture, but competition for water resources is increasing due to population growth, urbanization, and climate change. Agriculture currently accounts for around 70% of global freshwater withdrawals, and reallocating water from agriculture to other sectors may be necessary in water-stressed regions (Giovannucci et al., 2012).

Several works have been done in this regard. A few prominent and recent works include:

Na et al. (2016) employ antimony electrodes for pH measurement. The inverse relationship between soil resistance and soil moisture is utilized for estimating soil moisture content, with corresponding circuitry developed accordingly. Soil temperature is determined using the DS18B20 sensor, which operates on the Dallas one-wire protocol. This system is integrated with Bluetooth technology to facilitate data transfer to a nearby mobile phone.

Ananthi et al. (2017) tested soil using various sensors, including pH, temperature, and humidity sensors. Based on the results, farmers can select and cultivate crops that are most suitable for their soil conditions.

Boobalan et al. (2018) propose a system that incorporates a Raspberry Pi, various sensors, a Pi camera, and a motor driver. The Pi camera captures video and transmits it to the cloud via the Raspberry Pi.

Raut et al. (2018) describe an automatic irrigation system that also measures the levels of the three major macronutrients—nitrogen (N), phosphorus (P), and potassium (K)—in the soil, thereby saving farmers time, money, and energy.

Bhatnagar and Chandra (2020) propose a soil health monitoring system that enables farmers to monitor soil moisture, soil temperature, and soil pH using an Android smartphone.

Sarma et al. (2023) suggest an IoT-based agriculture environment and security monitoring system to address challenges in the agricultural sector, including water constraints, soil degradation, and climate change.

1.1 Aim of the project

To address the challenges faced by the agriculture sector, there is a need to accelerate productivity growth in agriculture through technological advancements, sustainable land management practices, and efficient resource use. Improving TFP, reducing soil erosion, and implementing water management strategies are essential for meeting future food demand and ensuring food security. Farmers can optimize resource use, improve yields, and contribute to a sustainable agricultural future by adopting IoT solutions and precision farming techniques.

Integrating sustainable sensors, cloud-based data analytics, weather predictions, and emerging technologies like the Internet of Things (IoT), Cloud Computing, Big Data, and Predictive Analysis holds immense potential in effectively addressing the problem at hand. Moreover, by incorporating a mobile app into the system, the real-time status of the field becomes easily accessible, allowing for remote control of the irrigation system.

Continuous real-time monitoring of the field throughout the year is made possible by deploying sustainable sensors. These sensors collect crucial data on environmental factors such as temperature, humidity, soil moisture, and sunlight exposure. Leveraging the power of the IoT, these sensors are seamlessly connected, enabling the transmission of real-time data for analysis.

The collected data can be efficiently processed and analyzed using cloud-based infrastructure and data analytics tools. Cloud Computing provides scalability and computational resources for handling large datasets, while advanced analytics techniques, including machine learning, enable the extraction of valuable insights.

Considering weather predictions in the analysis adds another layer of understanding to the field conditions. Predictive Analysis methods help anticipate future trends and potential risks, enabling proactive decision-making.

To enhance accessibility and control, connecting the system to a mobile app is highly beneficial. This integration allows users to access the real-time status of the field remotely. Additionally, it enables convenient control of the irrigation system through the app, empowering users to make adjustments based on the collected data and analysis.

By combining these technologies and functionalities, the system becomes a comprehensive solution. It offers real-time monitoring and analysis and empowers users with remote control capabilities, ultimately leading to more efficient resource management, optimized irrigation practices, and improved agricultural outcomes.

2. MATERIALS AND METHODS:

Our automated crop monitoring and irrigation system is a sophisticated solution designed to optimize agricultural practices and maximize crop yields. This comprehensive system comprises various components and technologies, each crucial in ensuring efficient and effective irrigation management.

2.1. Materials

Figure 1 shows the block diagram of the system.

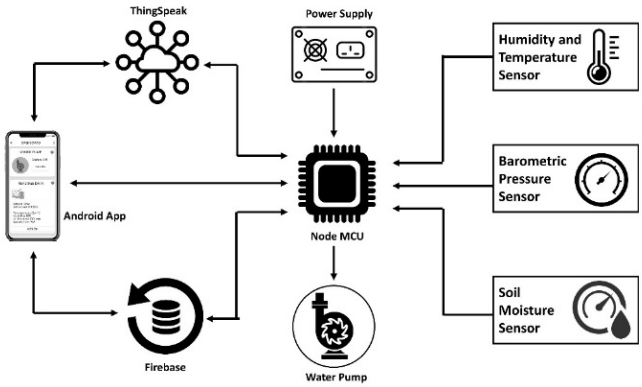

Figure 1. Block Diagram of the System.

The following are the components of our system:

- **Node MCU:** At the core of our system lies the Node MCU (Dutta & Khurana, 2021). NodeMCU stands for Node Microcontroller Unit. It is a powerful microcontroller based on the ESP8266 WiFi module (Mesquita et al., 2019). This compact and versatile device serves as the central control unit, responsible for gathering data from various sensors, processing it, and communicating with the cloud and mobile app.
- **Soil moisture sensor:** To monitor soil moisture levels, we employ a high-precision soil moisture sensor FC-28 (Bhadani & Vashisht, 2019). This sensor measures the moisture content in the soil, providing valuable insights into the watering needs of the crops. By regularly collecting and analyzing this data, farmers can ensure optimal irrigation levels and prevent under or overwatering.
- **Humidity and temperature sensor:** In addition to monitoring soil moisture, our system also tracks humidity levels in the farm environment. DHT11 humidity and temperature sensor (Gay & Gay, 2018) is deployed to measure the amount of moisture present in the air. This data helps farmers understand the atmospheric conditions and make informed decisions regarding irrigation scheduling.
- **Barometric Sensor:** To gauge atmospheric pressure, which can impact crop growth and water requirements, a BMP280 barometric sensor (Kusuma et al., 2023) is integrated into our system. Farmers can anticipate weather patterns by monitoring atmospheric pressure changes and adjust their irrigation plans accordingly. This information proves invaluable in ensuring that crops receive the appropriate amount of water at the right time.
- **ThingSpeak:** To facilitate data management and analytics, our system incorporates ThingSpeak (Nakhuva & Champaneria, 2015), an open-source Internet of Things (IoT) platform. This cloud-based platform receives and stores the data collected by the sensors, allowing for real-time monitoring and analysis. ThingSpeak also offers visualization tools that enable users to track historical data trends, helping farmers identify patterns and make data-driven decisions.
- **Firebase:** To enhance the functionality and scalability of our system, we integrate Firebase (Moroney & Moroney, 2017), a popular mobile and web application development platform. Firebase serves as a real-time database, enabling seamless synchronization of data across multiple devices and providing instant updates to users. By leveraging Firebase's capabilities, our system ensures that users have access to the latest information regarding their crops and irrigation processes.
- **Android Mobile Application:** To provide users with a user-friendly interface and seamless control over the system, we have developed an Android mobile application. This intuitive and feature-rich app empowers farmers to monitor real-time data, visualize historical trends, access crop yield predictions, and control the water pump. With just a few taps on their mobile devices, farmers can remotely start or stop the water pump, ensuring the irrigation process aligns with the specific needs of their crops.
- **Water pump:** The pump itself is a vital system component. It is responsible for delivering water to the field based on the data collected from the soil moisture sensor. Integrating the water pump into our automated system eliminates the need for manual irrigation management and reduces the risk of human error. This automated approach guarantees consistent and optimized watering, promoting healthy crop growth and

minimizing water waste.

Figure 2 shows the working of the prototype of our system.

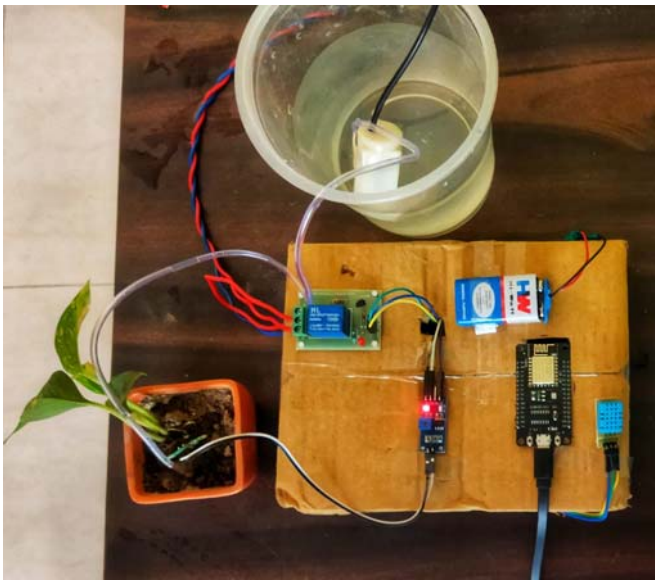

Figure 2: Prototype of the system

2.2. Methods

The workflow begins with data collection from soil moisture, humidity, and pressure sensors, followed by data processing and transmission to cloud platforms such as ThingSpeak and Firebase. Through an intuitive mobile app, farmers can access real-time data, control the water pump remotely, and gain valuable insights into crop conditions. With optimization and decision-making capabilities, this system enhances sustainable agriculture practices, helping farmers make informed choices and achieve maximum crop productivity:

2.2.1. Data Collection

The system utilizes various sensors, including a soil moisture sensor, humidity sensor, and barometric pressure sensor, to collect data related to soil moisture, atmospheric humidity, and atmospheric pressure. These sensors are connected to the Node MCU, which acts as the central control unit of the system.

The FC-28 soil moisture sensor is connected to the analog pin A0 of the NodeMCU. The DHT11 humidity and temperature sensor is connected to digital pin D2.

To initialize the I2C (Inter-Integrated Circuit) communication interface on the NodeMCU microcontroller, pin 4 is used for SDA and pin 5 is used for SCL. I2C is a popular communication protocol used to connect multiple peripheral devices with a microcontroller or a microprocessor. It allows multiple devices to communicate using only two wires: SDA (Serial Data Line) and SCL (Serial Clock Line). This allows the NodeMCU to communicate with I2C-compatible devices such as the BMP280 sensor.

Lastly, the power source is connected to the NodeMCU and the water pump is connected to the digital pin D1.

The MicroPython firmware version 1.22.2 (Tollervey, 2017; Gaspar et al., 2020) is flashed to the NodeMCU board. MicroPython is a lean and efficient Python 3 programming language implementation optimized to run on microcontrollers and embedded systems.

The following Python modules ("MicroPython libraries — MicroPython latest documentation," n.d.). are used to gain access to the necessary functions and classes to control the microcontroller's pins, communicate with I2C (Inter-Integrated Circuit) devices, perform serial communication, read analog values, work with time-related operations, and interface.

- **machine.Pin:** This module provides an interface to control the pins of the microcontroller. It allows you to configure pins as inputs or outputs, set their state, and read their values. It is commonly used for interacting with sensors, actuators, and other hardware peripherals connected to the microcontroller.
- **machine.I2C:** The I2C (Inter-Integrated Circuit) module enables communication with devices that support the I2C protocol. It allows data to be sent and received over the I2C bus, which is a popular communication interface for sensors, displays, and other devices.
- **machine.UART:** The UART (Universal Asynchronous Receiver-Transmitter) module provides access to the serial communication interface on the microcontroller. It allows data to be sent and received over serial connections, such as with a computer or other devices.
- **machine.ADC:** The ADC (Analog-to-Digital Converter) module provides access to the analog-to-digital converter on the microcontroller. It allows to read analog voltages from sensors or other analog sources.
- **time:** The time module provides functions to work with time-related operations, such as adding delays, measuring time intervals, and getting the current time.

- **bmp280:** This module provides a driver to interact with the BMP280 barometric pressure and temperature sensor. It allows data such as temperature and atmospheric pressure to be read from the sensor.
- **dht:** This module provides a driver to interact with DHT series humidity and temperature sensors, such as the DHT11 and DHT22. It allows data such as temperature and humidity to be read from the sensor.

2.2.2 Data Processing and Transmission

The Node MCU processes the data collected from the sensors, performing necessary calculations and formatting. The processed data is then transmitted to the cloud platform, ThingSpeak and Firebase for storage and further analysis.

2.2.2.1 Data Upload to ThingSpeak

Firstly, we set up the credentials for ThingSpeak: our Channel ID and API Keys. We also define an interval of 30 seconds. We set up a loop to send the sensor data to our ThingSpeak channel every 30 minutes.

2.2.2.2 Data Upload to Firebase

Firebase, a real-time database, is used to synchronize the data across multiple devices and provide instant updates to users.

To connect our device to Google Firebase we have used the Firebase Admin SDK (Yahiaoui, 2017). The Admin SDK allows to interact with Firebase services, including the Real-time Database and Firestore (Kesavan et al., 2023), a flexible, scalable, and server-less cloud database provided by Google as part of the Firebase platform.

2.2.3 Mobile Application

An Android mobile app is developed to provide a user-friendly interface for farmers to interact with the system. The mobile app allows farmers to monitor real-time data, visualize historical trends, and control the water pump remotely. Through the app, users can access the ThingSpeak visualizations and the Firebase Real-time Database to gain insights to the data.

The mobile application has been developed using AppGyver (Oteyo et al., 2021), a low-code WYSIWYG (What You See Is What You Get) app development platform that allows users to create web and mobile apps using a visual interface. It offers a range of pre-built components and supports integration with various APIs.

The app shows the status of the water pump (ON/OFF) along with an option to toggle the status. It also displays the temperature, humidity, atmospheric pressure, and soil moisture levels in real time. Figure 3 shows the mobile application dashboard interface.

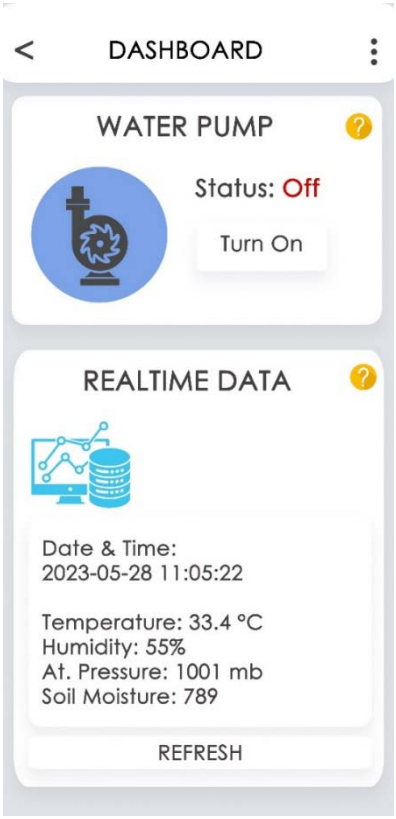

Figure 3: Mobile application dashboard interface

2.2.4 Irrigation Control

The water pump, a critical component of the system, is controlled based on the data collected from the soil moisture sensor. The system analyzes the soil moisture data to determine the optimal irrigation requirements. Farmers can remotely control the water pump through the mobile app via the Node MCU, selecting from options such as turning it on, or off.

For the coding part, we use relay module and MQTT. A relay module is an electrical switch that uses an electromagnet to control the flow of current to an external device or circuit. It acts as a digital switch that can be controlled by a microcontroller or a device like the NodeMCU. When energized, the relay closes the circuit, allowing current to flow through it and activating the connected device. When de-energized, the relay opens the circuit, interrupting the current flow and deactivating the connected device.

MQTT (Message Queuing Telemetry Transport) is a lightweight messaging protocol commonly used in IoT (Internet of Things) applications. It is designed to be efficient, simple to implement, and well-suited for constrained

devices with low bandwidth and limited processing capabilities.

2.2.5 Predictive Analysis and Crop Yield Prediction

Predictive analysis (Joshi et al., 2020) is a powerful methodology that utilizes historical data and statistical algorithms to forecast future outcomes or trends. In the context of agriculture and environmental management, predictive analysis plays a crucial role in optimizing decision-making processes. By analyzing historical trends, environmental conditions, and irrigation practices, the system can forecast the future yield of crops. These predictions assist farmers in making informed decisions regarding crop management and resource allocation.

To implement predictive analysis into the system, we have used the following steps:

Over a period of time, the data collected from the sensors is combined with historical weather obtained from a public weather database OpenWeather's API, One Call API version 3.0 ("OpenWeatherMap.org.," n.d.) to create a comprehensive dataset. The dataset has the following features as shown in Figure 4.

```
Column Names:
Index(['Date',    'Time',    'Temperature',
'Humidity', 'Soil Moisture',
       'Wind  Speed',  'Wind  Direction',
'Rainfall'],
      dtype='object')

Data Types:
Date              object
Time              object
Temperature        int64
Humidity           int64
Soil Moisture      int64
Wind Speed         int64
Wind Direction    object
Rainfall         float64
dtype: object
```

Figure 4: Feature of the dataset

The dataset is divided into two subsets: training and test sets. The training set (80% of the data) is used to train the model, while the test set (the remaining 20%) is kept aside to evaluate the model's performance.

The random forest (Breiman, 2001) algorithm as the machine learning model for prediction due to its ability to handle both regression tasks (predicting continuous variables) and classification tasks, as well as its robustness and feature importance analysis. In the random forest algorithm, multiple decision trees are trained independently and combined to make predictions.

The random forest model is trained using the training set. The model learns the relationship between the input features (such as Temperature, Humidity, etc.) and the target variables (Temperature and Soil Moisture).

The performance of the trained model is accessed using the test set. The Mean Squared Error (MSE) (Hodson et al., 2021) evaluation metric was used to gauge the accuracy of the model's temperature and soil moisture levels predictions.

The Mean Squared Error (MSE) is a common metric used to evaluate the performance of regression models, particularly in machine learning tasks. It measures the average of the squared differences between the predicted values and the actual values. Mathematically, the MSE is calculated shown in Equation 1:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad \text{..... (1)}$$

Where:

n - number of samples or data points in the dataset.

y_i - the actual value of the target variable for the i^{th} data point.

$\hat{y}_i$ - the predicted value of the target variable for the i^{th} data point.

The MSE computes the squared differences between each predicted value and its corresponding actual value, then averages these squared differences across all data points.

An algorithm is developed that takes the predicted temperature and soil moisture levels as inputs. These predictions determine the optimal irrigation frequency, duration, and timing to maintain the desired soil moisture level and support plant growth.

2.2.5 Source Codes

The implementation of the system involves two key components: real-time control of irrigation using the NodeMCU microcontroller, and predictive analysis. Here we provide the source code for both the components.

2.2.5.1 Arduino Code

The following code connects the sensors to the Node MCU and periodically uploads the data to ThingSpeak and Firebase database. The code for this section is written in MicroPython version 1.22.2 using the Arduino IDE.

Firstly, the necessary libraries are imported.

```
import dht
import bmp280
import machine
import time
from umqtt.simple import MQTTClient
import urequests
import firebase_admin
from firebase_admin import credentials, db
```

The sensor connections are then set up with the NodeMCU. Analog pin A0 is used for reading soil moisture levels. Digital pin

```
# Setting up the sensor connections with
NodeMCU

# assigned to the analog pin A0 for
reading soil moisture levels
sensorpin1 = machine.A0

# assigned to digital pin 2 for the DHT11
temperature and humidity sensor
sensorpin2 = machine.Pin(2)

dht_sensor = dht.DHT11(sensorpin2)

# initializing I2C communication using
pins 5 (SCL) and 4 (SDA)
i2c = machine.I2C(scl=machine.Pin(5),
sda=machine.Pin(4))

# initializing the BMP280 sensor for
atmospheric pressure readings
bmp = bmp280.BMP280(i2c=i2c)
```

After that, we define the functions to read data from the sensors.

```
# Defining the functions to read data from
the sensors

def read_soil_moisture():
    # Reading the soil moisture sensor and
converts the raw value to a percentage
    value = sensorpin1.read()
    moisture_percentage = (value / 1023) *
100
    return moisture_percentage

def read_temperature_humidity():
    # Reading temperature and humidity
from DHT sensor
    dht_sensor.measure()
    temperature = dht_sensor.temperature()
    humidity = dht_sensor.humidity()
    return temperature, humidity

def read_atmospheric_pressure():
    # Reading atmospheric pressure from
BMP280 sensor
    pressure = bmp.pressure
    return pressure
```

Functions are defined to print the read data.

```
# Printing the sensor readings
def print_sensor_data():
    timestamp = time.strftime('%Y-%m-%d
%H:%M:%S')
    temperature, humidity =
read_temperature_humidity()
    atmospheric_pressure =
read_atmospheric_pressure()
    soil_moisture = read_soil_moisture()
    print("Timestamp:", timestamp)
    print("Temperature:", temperature,
"°C")
    print("Humidity:", humidity, "%")
    print("Pressure:",
atmospheric_pressure, "hPa")
    print("Soil Moisture:", soil_moisture,
"%")
    print()
```

The pump is set up.

```
# Setting up the Pump
relay_pin = machine.Pin(5,
machine.Pin.OUT)

def turn_on_pump():
    #Turning on the water pump
    relay_pin.value(0)
    print("Pump is ON")

def turn_off_pump():
    #Turning off the water pump
    relay_pin.value(1)
    print("Pump is OFF")
```

Post that step, the MQQT protocol is set up.

```
# MQTT setup
mqtt_broker = 'mqtt.thingspeak.com'
mqtt_port = 1883
mqtt_user = 'smartagriculture'
mqtt_password = 'SmaAg@321'
mqtt_topic = 'water_pump/control'

def mqtt_callback(topic, msg):
    message = msg.decode('utf-8')
    if message == 'ON':
        turn_on_pump()
    elif message == 'OFF':
        turn_off_pump()
    else:
        print("Invalid command.")

client = MQTTClient("nodemcu_client",
mqtt_broker, mqtt_port, mqtt_user,
mqtt_password)
client.set_callback(mqtt_callback)
client.connect()
client.subscribe(mqtt_topic)
```

Now, the ThingSpeak Channel is set up with the API Key.

```
# Setting up the credentials for
ThingSpeak: Channel and API
channel_id = "2179018"
write_api_key = "Y4VCD299459S30JQ"
read_api_key = "H800DIMY47WE2N3D"
interval = 5 * 60
# 5 minutes interval
```

Data is read from the sensors and sent to ThingSpeak every 5 minutes.

```
# Reading the data from the sensors
def read_sensor_data():
    moisture = read_soil_moisture()
    temperature, humidity =
read_temperature_humidity()
    pressure = read_atmospheric_pressure()
    return moisture, temperature,
humidity, pressure

# Defining a function to send data to
ThingSpeak
def send_data_to_thingspeak(timestamp,
moisture, temperature, humidity,
pressure):
    # Sending sensor data to ThingSpeak
    url =
"https://api.thingspeak.com/update?api_key
={}".format(write_api_key)
    payload =
"field1={:.2f}&field2={:.2f}&field3={:.2f}
&field4={:.2f}&field5={:.2f}".format(
        timestamp, moisture, temperature,
humidity, pressure
    )
    try:
        response = urequests.get(url,
params=payload)
        response.close()
        print("ThingSpeak Update
successful.")
    except Exception as e:
        print("ThingSpeak Update error:
{e}")

# Setting up a loop to send the sensor
data to our ThingSpeak channel every 5
minutes
while True:
    timestamp = time.time()
    moisture, temperature, humidity,
pressure = read_sensor_data()
    send_data_to_thingspeak(timestamp,
moisture, temperature, humidity, pressure)
    time.sleep(interval)
```

Next, the Firebase Database is set up.

```
# Firebase setup
cred =
credentials.Certificate('path/to/smart-
agriculture-2e093.json')
firebase_admin.initialize_app(cred,
{'databaseURL': 'https://smart-
agriculture-2e093-default-
rtdb.firebaseio.com/'})
ref = db.reference('/')
```

Now, the weather data has been sent to Firebase.

```
# Sending the weather data to Firebase
database
def send_weather_data(moisture,
temperature, humidity, pressure):
    timestamp = int(time.time())
    data = {
        'Timestamp': timestamp,
        'Soil Moisture': moisture,
        'Temperature': temperature,
        'Humidity': humidity,
        'Pressure': pressure,
    }
    try:
        ref.push(data)
        print("Database Updated.")
    except Exception as e:
        print("Database Update error:
{e}")
```

The main loop continuously reads sensor data, prints it, and sends it to the Firebase database at a predefined interval.

```
# Main loop
while True:
    try:
        print_sensor_data()

send_weather_data(*read_sensor_data())
    except Exception as e:
        print("Error: {e}")
    time.sleep(interval
```

2.2.5.2 Predictive Analysis Code

The code below predicts the temperature and soil moisture levels, and based on these inputs, the optimal irrigation frequency, duration, and timing are calculated to maintain the desired soil moisture level.

The coding has been done in Python 3.12.2 using Jupyter Notebook.

Firstly, the necessary libraries are imported.

```
import pandas as pd
from sklearn.model_selection import
train_test_split
from sklearn.ensemble import
RandomForestRegressor
from sklearn.metrics import
mean_squared_error
```

The dataset that has been formatted into a CSV (Comma Separated Values) file is now loaded.

```
# Loading the dataset
data = pd.read_csv('weather_data.csv')
```

The data is split into features and target variables.

```
# Splitting the data into features and
target variables
X = data[['Temperature', 'Humidity', 'Wind
Speed', 'Wind Direction', 'Rainfall']]
y_temperature = data['Temperature']
y_soil_moisture = data['Soil Moisture']
```

Then, the dataset is split into training and testing data. 80% of the data will be used for training and the remaining 20% for testing. The 'random_state' parameter specifies the seed value for the random number generator. This seed is used to ensure that the data split is reproducible.

```
# Splitting data into training and testing
sets
X_train, X_test, y_temp_train,
y_temp_test, y_soil_train, y_soil_test =
train_test_split(X, y_temperature,
y_soil_moisture, test_size=0.2,
random_state=42)
```

The random forest model for temperature and soil moisture predictions.

Here, the 'n_estimators' parameter specifies the number of decision trees to be used in the random forest. In this case, n_estimators=100 means that the random forest will consist of 100 decision trees.

```
# Training the random forest model for
temperature prediction
rf_temp =
RandomForestRegressor(n_estimators=100,
random_state=42)
rf_temp.fit(X_train, y_temp_train)
```

```
# Training the random forest model for
soil moisture prediction
rf_soil =
RandomForestRegressor(n_estimators=100,
random_state=42)
rf_soil.fit(X_train, y_soil_train)
```

Temperature and soil moisture predictions are made using the trained models on the testing data.

```
# Making predictions
temp_predictions = rf_temp.predict(X_test)
soil_predictions = rf_soil.predict(X_test)
```

Mean squared error (RMSE) is calculated for both temperature and soil moisture predictions.

```
# Evaluating the models
temp_rmse =
mean_squared_error(y_temp_test,
temp_predictions, squared=False)
soil_rmse =
mean_squared_error(y_soil_test,
soil_predictions, squared=False)

print(f"Temperature RMSE: {temp_rmse}")
print(f"Soil Moisture RMSE: {soil_rmse}")
```

An algorithm is now created to adjust irrigation frequency and duration based on temperature and soil moisture predictions. It categorizes irrigation frequency and duration into "High/Low/Normal" and "Long/Short/Normal," respectively, based on the predictions.

```
# Algorithm for adjusting irrigation
frequency, duration, and timing
def adjust_irrigation(temp_prediction,
soil_moisture_prediction):
    if temp_prediction > 30:
        irrigation_frequency = "High"
    elif temp_prediction < 10:
        irrigation_frequency = "Low"
    else:
        irrigation_frequency = "Normal"

    if soil_moisture_prediction < 20:
        irrigation_duration = "Long"
    elif soil_moisture_prediction > 80:
        irrigation_duration = "Short"
    else:
        irrigation_duration = "Normal"

    return irrigation_frequency,
irrigation_duration
```

Lastly, the algorithm is tested by providing five test cases. For each test case, temperature and soil moisture predictions are used to recommend irrigation frequency and duration.

```
# Testing the algorithm
print("Testing:")
for i in range(5):
    temp_pred = temp_predictions[i]
    soil_pred = soil_predictions[i]
    irrigation_frequency,
irrigation_duration =
adjust_irrigation(temp_pred, soil_pred)
    print(f"Test {i+1}:")
    print(f"  - Temperature Prediction:
{temp_pred:.2f}°C")
    print(f"  - Soil Moisture Prediction:
{soil_pred:.2f}%")
    print(f"  - Recommended Irrigation
Frequency: {irrigation_frequency}")
    print(f"  - Recommended Irrigation
Duration: {irrigation_duration}")
```

3. RESULTS AND DISCUSSION:

3.1. Results

In the context of Indian farming, the automated crop monitoring and irrigation system has shown promising results during the testing phase. By utilizing IoT technology, cloud computing, and predictive analytics, the system has the potential to revolutionize agriculture practices and significantly benefit Indian farmers. Let's explore how the system performed and its positive impact on crop monitoring and irrigation.

The experimental evaluation of the system was conducted in two distinct phases: Weather Monitoring and Irrigation Phase and the Predictive Analysis Phase.

3.1.1 Weather Monitoring and Irrigation Phase

In the Weather Monitoring and Irrigation phase, the system demonstrated remarkable proficiency in acquiring real-time data and effectively controlling the water pump as depicted in Figure 5. The ThingSpeak content was updated correctly as shown in Figure 6. Across different settings, the system consistently provided accurate temperature, pressure, soil moisture, and humidity readings. The efficient water pump control mechanism ensured optimal irrigation management, enhancing crop health and productivity.

```
Timestamp: 2024-03-12 08:00:00
Temperature: 25 °C
Humidity: 60 %
Pressure: 1013.25 hPa
Soil Moisture: 42.73 %

ThingSpeak Update successful.
Database Updated.
Pump is OFF

Timestamp: 2024-03-12 08:05:00
Temperature: 25.1 °C
Humidity: 59.5 %
Pressure: 1013.2 hPa
Soil Moisture: 42.69 %

ThingSpeak Update successful.
Database Updated.
Pump is OFF

Timestamp: 2024-03-12 08:10:00
Temperature: 25.2 °C
Humidity: 59 %
Pressure: 1013.1 hPa
Soil Moisture: 42.66 %

ThingSpeak Update successful.
Database Updated.
Pump is OFF
```

Figure 5: Output of Arduino Code

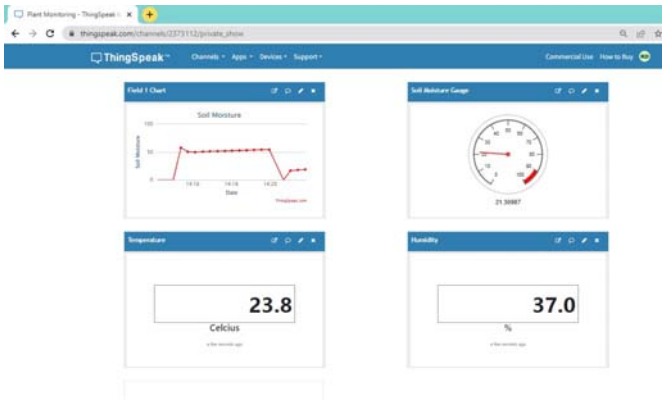

Figure 6: ThingSpeak Output

3.1.2 Predictive Analysis Phase

3.1.2.1 Testing Algorithm for Weather Prediction and Irrigation Recommendations

In the Predictive Analysis phase, the algorithm was first tested to check if the weather prediction and irrigation recommendations were working correctly. They were both found to be acceptable, as shown in Figure 7.

```
Test 1:
  - Temperature Prediction: 31.94°C
  - Soil Moisture Prediction: 68.31%
  - Recommended Irrigation Frequency: Normal
  - Recommended Irrigation Duration: Normal
Test 2:
  - Temperature Prediction: 31.20°C
  - Soil Moisture Prediction: 66.67%
  - Recommended Irrigation Frequency: Normal
  - Recommended Irrigation Duration: Normal
Test 3:
  - Temperature Prediction: 31.83°C
  - Soil Moisture Prediction: 66.35%
  - Recommended Irrigation Frequency: Normal
  - Recommended Irrigation Duration: Normal
Test 4:
  - Temperature Prediction: 31.15°C
  - Soil Moisture Prediction: 66.10%
  - Recommended Irrigation Frequency: Normal
  - Recommended Irrigation Duration: Normal
Test 5:
  - Temperature Prediction: 31.31°C
  - Soil Moisture Prediction: 66.03%
  - Recommended Irrigation Frequency: Normal
  - Recommended Irrigation Duration: Normal
```

Figure 7: Output of Predictive Algorithm

3.1.2.2 Performance Evaluation

After that, the system's performance was evaluated over a span of one week. Data acquired from the system's sensors was amalgamated with historical weather data sourced from OpenWeather's API during this period. This integration resulted in the creation of a comprehensive dataset. A dataset comprising 100 sets of temperature and soil moisture values, synchronized at identical timestamps was utilized for actual and predicted values. The system's predictive capabilities were assessed, revealing minimal deviation between the predicted and actual temperature and soil moisture levels, as shown in Figures 8(a) and 8(b).

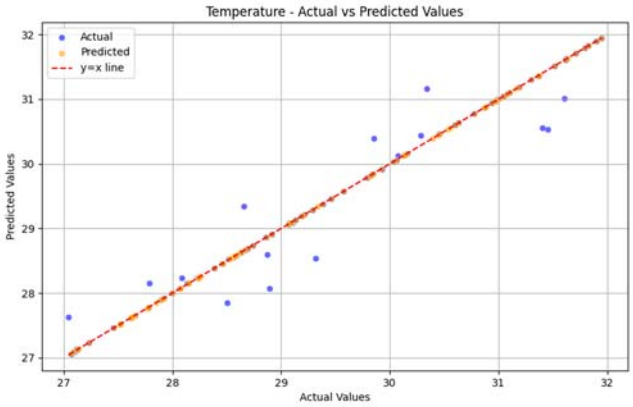

Figure 8(a): Temperature Predictions v/s Actual Data

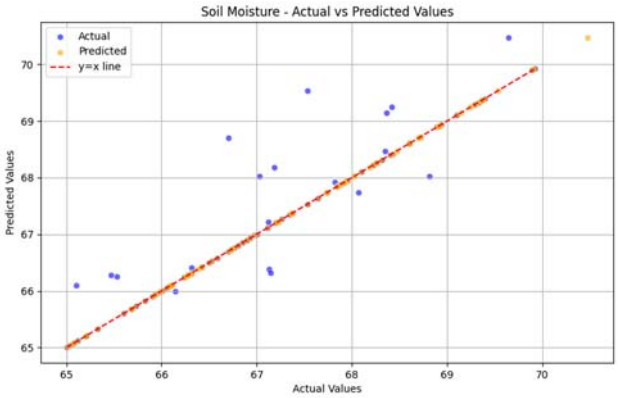

Figure 8(b): Soil Moisture Predictions v/s Actual Data

3.1.2.3 Statistical Analysis of the Results

Statistical analysis of the results involves examining various measures such as mean values, standard deviations, and confidence intervals to gain insights into the data's characteristics and the reliability of the predictions. Here's how each of these statistical measures contributes to the analysis.

Mean Values

The mean value represents the average of a dataset and provides a central tendency measure. Calculating the mean values of actual and predicted data helps understand the typical values observed for temperature and soil moisture levels. Comparing the mean values of actual and predicted data can indicate if the prediction model tends to overestimate or underestimate the variables.

The mean values for the temperature and soil moisture predictions as compared to the means of the actual values, are given in Table 1.

	Temperature	Soil Moisture
Actual	29.33097	67.23058
Predicted	29.31658	67.31588

Table 1: Mean values

Standard Deviation

Standard Deviations: Standard deviation quantifies the spread or dispersion of data points around the mean. A low standard deviation indicates that the data points are close to the mean, while a high standard deviation suggests greater variability. Analyzing the standard deviations of actual and predicted data helps assess the consistency and reliability of the predictions. Lower standard deviations imply more accurate and consistent predictions, while higher

deviations may indicate greater uncertainty.

The standard deviations for the temperature and soil moisture predictions, as compared to of the actual values, are given in Table 2.

	Temperature	Soil Moisture
Actual	1.453039693	1.311055
Predicted	1.436644586	1.349004

Table 2: Standard deviation

Confidence Intervals

Confidence intervals provide a range of values within which the true population parameter is likely to lie with a certain level of confidence. Calculating confidence intervals for predicted values allows us to estimate the range within which the true values are expected to fall. A narrower confidence interval indicates higher precision and confidence in the predictions, while a wider interval suggests greater uncertainty. Analyzing the overlap or disparity between confidence intervals of actual and predicted data provides insights into the accuracy and reliability of the prediction model. We will consider the mean difference and standard error for the two sets of actual and predicted values of temperature and soil moisture predictions.

The mean difference is a measure that quantifies the average disparity or discrepancy between two sets of values. It is calculated by finding the arithmetic mean of the differences between corresponding values in the two datasets.

The standard error measures the variability or uncertainty associated with an estimate or statistic, such as the mean difference. It represents the standard deviation of a statistic's sampling distribution, indicating how much the sample statistic tends to vary from the true population parameter on average. A smaller standard error suggests that the sample statistic is more likely to be close to the population parameter, while a larger standard error indicates greater uncertainty or variability in the estimate.

The confidence intervals are depicted in Figure 9.

```
Temperature Predictions

Mean Difference: 0.01438613861386166
Standard Error: 0.023571424932026137
Confidence          Interval:          (-
0.03237889710055178, 0.061151174328275104)

Soil Moisture Predictions

Mean Difference: -0.08530000000000029
Standard Error: 0.03955781474084111
Confidence          Interval:          (-
0.16379128657341727,                    -
0.006808713426583318)
```

Figure 9. Confidence Intervals.

3.2. Discussion

The automated crop monitoring and irrigation system demonstrated promising results during the testing phase in the Indian farming scenario. By leveraging IoT technology, cloud computing, and predictive analytics, the system showcased its potential to revolutionize agricultural practices. With accurate monitoring of soil moisture, real-time access to data, and personalized recommendations, the system has the power to enhance crop management, optimize water usage, and improve overall productivity for Indian farmers. As the system moves from testing to implementation, its benefits have the potential to transform the agricultural landscape in India, empowering farmers and promoting sustainable farming practices.

3.2.1 Analysis of the results

The results indicate that the predictive model performs well in estimating the target variable, with predicted values closely resembling the actual observations. The consistency in predictions and the lack of statistically significant differences between actual and predicted values further support the model's accuracy and reliability.

- The negligible difference between the means suggests that the predictive model is performing reasonably well in capturing the central tendency of the data.
- The predictive model demonstrates consistency in its predictions, as can be seen from the lower standard deviation of the predicted values.
- The calculation of confidence intervals show that while there is a slight discrepancy between the actual and predicted values, the difference is relatively small and not statistically significant based on the confidence

interval. Therefore, the predictive model appears to provide reasonably accurate estimates, with the predicted values closely aligning with the actual observations.

3.2.2 Areas of improvement

While the system has achieved positive results during the testing phase, there are several areas of future scope and potential improvements that can further enhance its effectiveness and address specific challenges faced by Indian farmers.

1. One area of future scope lies in expanding the system's compatibility with different types of crops and farming practices. The system has been primarily tested with staple crops such as rice and wheat. To cater to the diverse agricultural landscape of India, it is crucial to include a wider range of crops, including fruits, vegetables, and cash crops. This expansion would require the development of crop-specific models and algorithms to provide accurate recommendations and predictions tailored to the specific needs of each crop.
2. Another aspect that can be further explored is the integration of satellite imagery and remote sensing technologies into the system. By incorporating satellite data, farmers can gain valuable insights into larger factors influencing crop growth, such as land cover, vegetation indices, and temperature gradients. This integration would enable farmers to monitor crop conditions on a regional or even national scale, helping them make informed decisions about resource allocation and crop planning.
3. Additionally, the system can benefit from the inclusion of pest and disease monitoring capabilities. Indian farmers face significant challenges in pest management, and early detection is crucial to minimize crop losses. The system can provide real-time alerts and recommendations for effective pest control measures by incorporating pest and disease monitoring sensors or leveraging image recognition technologies. This integration would empower farmers to take proactive measures and minimize the impact of pests on their crops.
4. To enhance the system's effectiveness, addressing connectivity issues and ensuring reliable data transmission in remote and rural areas is essential. In India, where internet connectivity can be inconsistent, efforts should be made to develop offline functionalities for the mobile app. This would enable farmers to access crucial data and control features even in areas with limited or no internet connectivity. Additionally, exploring alternative communication technologies, such as low-power wide-area networks (LPWANs) (Chaudhari et al., 2020), can improve data transmission reliability in rural settings.
5. The system's mobile app can be further enriched with additional features and functionalities to meet the diverse needs of Indian farmers. For instance, incorporating market prices and trends can help farmers make informed decisions about crop marketing and improve their profitability. Providing access to relevant agricultural resources, such as best practices, training materials, and weather advisories, can empower farmers with knowledge and support their decision-making processes.
6. In terms of scalability, the system can explore partnerships with government agencies and agricultural cooperatives to reach a larger number of farmers. Collaborating with existing agricultural extension services can facilitate the adoption of the system among farmers and ensure effective dissemination of information and support.
7. Moreover, continuous research and development efforts are necessary to refine the predictive models and algorithms used in the system. By incorporating machine learning and artificial intelligence techniques, the system can continually improve its accuracy in predicting crop yields, identifying optimal irrigation schedules, and providing personalized recommendations. This would require ongoing data collection, analysis, and refinement of the models to adapt to changing climatic conditions, crop varieties, and farming practices.

3.2.3 Broader implications and practical applications

The findings from the research have several broader implications and practical

applications in agriculture.

1. **Optimized Resource Allocation:** Accurate crop yield predictions enable farmers and agricultural practitioners to allocate resources such as water, fertilizers, and pesticides more efficiently. Farmers can adjust input quantities to match the anticipated demand by knowing the expected yield in advance, reducing waste and minimizing costs.

2. **Risk Management:** Reliable crop yield predictions help farmers mitigate risks associated with uncertain weather conditions, market fluctuations, and environmental factors. Farmers can make informed decisions regarding crop selection, planting schedules, and risk management strategies by having a clearer understanding of potential yield outcomes.

3. **Financial Planning and Decision Making:** Crop yield predictions provide valuable insights for financial planning and decision-making processes. Farmers can use yield forecasts to estimate future revenues, negotiate contracts with buyers or suppliers, and secure financing for agricultural operations. This information is essential for budgeting, investment planning, and long-term sustainability.

4. **Research and Innovation:** Findings from such research can contribute to ongoing research and innovation in agricultural science, data analytics, and technology development. Continuous refinement of predictive models, integration of new data sources, and advancements in machine learning and artificial intelligence empower stakeholders to make more accurate and timely predictions, driving improvements in crop productivity, resilience, and sustainability.

4. CONCLUSIONS:

The automated crop monitoring and irrigation system, powered by IoT technology, cloud computing, and data analytics, has the potential to revolutionize Indian farming. During testing, the system effectively monitored real-time soil moisture, temperature, humidity, and pressure, providing accurate field data for efficient irrigation management. Integration of weather predictions further improved water usage by aligning irrigation schedules with upcoming weather patterns. The accompanying mobile app offered farmers a user-friendly interface for monitoring and controlling the system, visualizing historical data, and receiving predictive analytics on crop yields. Future improvements include expanding compatibility with diverse crops, integrating satellite imagery and remote sensing technologies for broader insights, addressing connectivity challenges in rural areas, and enriching the system with features like pest monitoring and market information. Continuous research and development efforts are essential to refine predictive models and algorithms using machine learning and artificial intelligence techniques. By optimizing resource utilization and promoting sustainable farming practices, this system can empower farmers, improve their livelihoods, and contribute to a resilient and prosperous agricultural sector in India.

The automated crop monitoring and irrigation system demonstrated promising results during the testing phase in the Indian farming scenario. By leveraging IoT technology, cloud computing, and predictive analytics, the system showcased its potential to revolutionize agricultural practices. With accurate monitoring of soil moisture, real-time access to data, and personalized recommendations, the system has the power to enhance crop management, optimize water usage, and improve overall productivity for Indian farmers. As the system moves from testing to implementation, its benefits have the potential to transform the agricultural landscape in India, empowering farmers and promoting sustainable farming practices. Analysis of the results indicates that the predictive model performs well in estimating the target variable, with predicted values closely resembling the actual observations. The consistency in predictions and the lack of statistically significant differences between actual and predicted values further support the model's accuracy and reliability. The negligible difference between the means suggests that the predictive model is performing reasonably well in capturing the central tendency of the data. The predictive model demonstrates consistency in its predictions, as can be seen from the lower standard deviation of the predicted values. The calculation of confidence intervals show that while there is a slight discrepancy between the actual and predicted values, the difference is relatively small and not statistically significant based on the confidence interval. Therefore, the predictive model appears to provide reasonably accurate estimates, with the predicted values closely

aligning with the actual observations. Areas of improvement for the system include expanding compatibility with different types of crops, integrating satellite imagery and remote sensing technologies, addressing connectivity issues in rural areas, incorporating pest and disease monitoring capabilities, enhancing the mobile app with additional features, and continuous refinement of predictive models through research and development efforts. The findings from such research have broader implications and practical applications in agriculture, including optimized resource allocation, risk management, financial planning, and decision-making, as well as opportunities for research and innovation to drive improvements in crop productivity, resilience, and sustainability.

5. DECLARATIONS

5.1. Study Limitations

a) Limited crop compatibility: The system has been primarily tested with staple crops like rice and wheat. Its compatibility with a wider range of crops, including fruits, vegetables, and cash crops, is not explored, limiting its applicability across the diverse agricultural landscape of India.

b) Connectivity challenges in rural areas: The manuscript does not address potential connectivity issues or limitations in remote and rural areas of India, where internet connectivity and data transmission reliability may be a concern.

c) Limited data source for predictive analysis: The source of the historical weather data used for training and testing the predictive analysis model is not clearly stated, which could affect the generalizability of the model's performance.

d) Lack of real-world testing and validation: The manuscript primarily focuses on the experimental setup and testing phase, but it does not provide information on real-world deployment, validation, and feedback from farmers, which could help identify additional limitations or areas for improvement.

5.2. Acknowledgements

I want to express my sincere gratitude to everyone who has offered their support during the research journey. While not directly involved in this project, the indirect contributions and unwavering support from various individuals have greatly contributed to its success. I also want to acknowledge the reviewers and editors for their valuable feedback and suggestions, which have significantly enhanced the quality of this research paper.

5.3. Funding source

This research was funded by the authors.

5.4. Competing Interests

The authors declare no potential conflict of interest in this publication.

6. REFERENCES:

1. Adesta, E. Y. T., Agusman, D., & Avicenna, A. (2017). Internet of things (IoT) in agriculture industries. Indonesian Journal of Electrical Engineering and Informatics (IJEEI), 5(4), 376-382. doi: 10.52549/ijeei.v5i4.373
2. Ananthi, N., Divya, J., Divya, M., & Janani, V. (2017, April). IoT based smart soil monitoring system for agricultural production. In 2017 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR) (pp. 209-214). IEEE. doi: 10.1109/TIAR.2017.8273717
3. Bhadani, P., & Vashisht, V. (2019, January). Soil moisture, temperature and humidity measurement using arduino. In 2019 9th International Conference on

Cloud Computing, Data Science & Engineering (Confluence) (pp. 567-571). IEEE. doi: 10.1109/CONFLUENCE.2019.8776973
4. Bhatnagar, V., & Chandra, R. (2020). IoT-based soil health monitoring and recommendation system. Internet of Things and Analytics for Agriculture, Volume 2, 1-21. doi: 10.1007/978-981-15-0663-5_1
5. Boobalan, J. N. K. T., Jacintha, V., Nagarajan, J., Thangayogesh, K., & Tamilarasu, S. (2018, April). An IOT based agriculture monitoring system. In 2018 international conference on communication and signal processing (ICCSP) (pp. 0594-0598). IEEE. doi: 10.1109/ICCSP.2018.8524490
6. Breiman, L. (2001). Random forests. Machine learning, 45, 5-32. doi: 10.1023/A:1010933404324
7. Chaudhari, B. S., Zennaro, M., & Borkar, S. (2020). LPWAN technologies: Emerging application characteristics, requirements, and design considerations. Future Internet, 12(3), 46. doi: 10.3390/fi12030046
8. Dutta, U., & Khurana, N. (2021). The Internet of Things Using NODEMCU. Blue Rose Publishers.
9. Gaspar, G., Fabo, P., Kuba, M., Flochova, J., Dudak, J., & Florkova, Z. (2020, December). Development of IoT applications based on the MicroPython platform for industry 4.0 implementation. In 2020 19th International conference on mechatronics-mechatronika (ME) (pp. 1-7). IEEE. doi: 10.1109/ME49197.2020.9286455
10. Gay, W., & Gay, W. (2018). DHT11 sensor. Advanced Raspberry Pi: Raspbian Linux and GPIO Integration, 399-418. doi: 10.1007/978-1-4842-3948-3_22
11. Giovannucci, D., Scherr, S. J., Nierenberg, D., Hebebrand, C., Shapiro, J., Milder, J., & Wheeler, K. (2012). Food and Agriculture: the future of sustainability. The sustainable development in the 21st century (SD21) Report for Rio, 20. doi: 10.2139/ssrn.2054838
12. Hodson, T. O., Over, T. M., & Foks, S. S. (2021). Mean squared error, deconstructed. Journal of Advances in Modeling Earth Systems, 13(12), e2021MS002681. doi: 10.1029/2021MS002681
13. Joshi, A., Khosravy, M., & Gupta, N. (2020). Machine learning for predictive analysis. Proceedings of ICTIS. doi: 10.1007/978-981-15-7106-0
14. Kesavan, R., Gay, D., Thevessen, D., Shah, J., & Mohan, C. (2023, April). Firestore: The nosql serverless database for the application developer. In 2023 IEEE 39th International Conference on Data Engineering (ICDE) (pp. 3376-3388). IEEE. doi: 10.1109/ICDE55515.2023.00259
15. Kumar, M. D., Malla, A. K., & Tripathy, S. K. (2008). Economic value of water in agriculture: Comparative analysis of a water-scarce and a water-rich region in India. Water International, 33(2), 214-230. doi: 10.1080/02508060802108750
16. Kusuma, H. A., Oktavia, D., Nugaraha, S., Suhendra, T., & Refly, S. (2023, March). Sensor BMP280 Statistical Analysis for Barometric Pressure Acquisition. In IOP Conference Series: Earth and Environmental Science (Vol. 1148, No. 1, p. 012008). IOP Publishing. doi: 10.1088/1755-1315/1148/1/012008
17. Lawrence, D., & Vandecar, K. (2015). Effects of tropical deforestation on climate and agriculture. Nature climate change, 5(1), 27-36. doi: 10.1038/nclimate2430
18. Le Mouël, C., & Forslund, A. (2017). How can we feed the world in 2050? A review of the responses from global scenario studies. European Review of Agricultural Economics, 44(4), 541-591. doi: 10.1093/erae/jbx006
19. Likas, A., Vlassis, N., & Verbeek, J. J. (2003). The global k-means clustering algorithm. Pattern recognition, 36(2), 451-461. doi: 10.1016/S0031-3203(02)00060-2
20. Mesquita, J., Guimarães, D., Pereira, C., Santos, F., & Almeida, L. (2018, September). Assessing the ESP8266 WiFi module for the Internet of Things. In 2018 IEEE 23rd international conference on emerging technologies and factory automation (ETFA) (Vol. 1, pp. 784-791). IEEE. doi: 10.1109/ETFA.2018.8502562
21. MicroPython libraries — MicroPython latest documentation. (n.d.). https://docs.micropython.org/en/latest/library/index.html
22. Moroney, L., & Moroney, L. (2017). The firebase real-time database. The Definitive Guide to Firebase: Build Android Apps on Google's Mobile Platform, 51-71. doi: 10.1007/978-1-4842-2943-9_3

23. Na, A., Isaac, W., Varshney, S., & Khan, E. (2016, October). An IoT based system for remote monitoring of soil characteristics. In 2016 International conference on information technology (InCITe)-the next generation IT summit on the theme-internet of things: Connect your Worlds (pp. 316-320). IEEE. doi: 10.1109/INCITE.2016.7857638
24. Nakhuva, B., & Champaneria, T. (2015). Study of various internet of things platforms. International Journal of Computer Science & Engineering Survey, 6(6), 61-74. doi:10.5121/ijcses.2015.6605
25. OpenWeatherMap.org. (n.d.). One Call API 3.0 - OpenWeatherMap. https://openweathermap.org/api/one-call-3
26. Oteyo, I. N., Pupo, A. L. S., Zaman, J., Kimani, S., De Meuter, W., & Boix, E. G. (2021, September). Building smart agriculture applications using low-code tools: the case for discopar. In 2021 IEEE AFRICON (pp. 1-6). IEEE. doi: 10.1109/AFRICON51333.2021.9570936
27. Quinton, J. N., & Fiener, P. (2024). Soil erosion on arable land: An unresolved global environmental threat. Progress in Physical Geography: Earth and Environment, 48(1), 136-161. doi: 10.1177/03091333231216595
28. Raut, R., Varma, H., Mulla, C., & Pawar, V. R. (2018). Soil monitoring, fertigation, and irrigation system using IoT for agricultural application. In Intelligent Communication and Computational Technologies: Proceedings of Internet of Things for Technological Development, IoT4TD 2017 (pp. 67-73). Springer Singapore. doi: 10.1007/978-981-10-5523-2_7
29. Sarma, P., & Bayan, T. (2023). IOT-based Agriculture Environment and Security Monitoring System. Periódico Tchê Química, 20(44). doi: 10.52571/PTQ.v20.n44.2023
30. Seckler, D., Barker, R., & Amarasinghe, U. (1999). Water scarcity in the twenty-first century. International Journal of Water Resources Development, 15(1-2), 29-42. doi: 10.1080/07900629948916
31. Singh, M., Rajan, M. A., Shivraj, V. L., & Balamuralidhar, P. (2015, April). Secure mqtt for internet of things (iot). In 2015 fifth international conference on communication systems and network technologies (pp. 746-751). IEEE. doi: 10.1109/CSNT.2015.16
32. The state of the world's land and water resources for food and agriculture | Land & Water | Food and Agriculture Organization of the United Nations | Land & Water | Food and Agriculture Organization of the United Nations. (n.d.). https://www.fao.org/land-water/solaw2021/en/
33. Tollervey, N. H. (2017). Programming with MicroPython: embedded programming with microcontrollers and Python. " O'Reilly Media, Inc.".
34. Verdouw, C., Wolfert, S., & Tekinerdogan, B. (2016). Internet of Things in agriculture. CABI Reviews, (2016), 1-12. doi: 10.1079/PAVSNNR201611035
35. Wu, X. D., Guo, J. L., Han, M. Y., & Chen, G. Q. (2018). An overview of arable land use for the world economy: From source to sink via the global supply chain. Land Use Policy, 76, 201-214. Doi: 10.1016/j.landusepol.2018.05.005
36. Yahiaoui, H. (2017). Firebase Cookbook: Over 70 recipes to help you create real-time web and mobile applications with Firebase. Packt Publishing Ltd.