A Project Report on

SMART SOIL MONITORING SYSTEM

Submitted for the partial fulfillment of the requirement of the degree of

BACHELOR OF TECHNOLOGY

In

ELECTRONICS AND COMMUNICATIONS ENGINEERING

Submitted by

ARPAN BISWAS	17600321003
SUBHRAJYOTI MANDAL	17600321004
SHIBAM MISHRA	17600321017
ANTARIP MANNA	17600321048
RIPA GHOSH	17632322012

Under the Supervision of

Mr. Subhojit Malik, Assistant Professor

Department of Electronics and Communications Engineering

Hooghly Engineering & Technology College

Affiliated to

MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY, KOLKATA, WEST BENGAL

HOOGHLY ENGINEERING & TECHNOLOGY COLLEGE

Vivekananda Road, Pipulpati, Hooghly-712103, West Bengal, India

June, 2025

DECLARATION

We solemnly declare that the project report on "**SMART SOIL MONITORING SYSTEM**" is based on our own work which was continuously carried out during the course of our study under the supervision of Mr. Subhojit Malik, Assistant Professor, Electronics and Communications Engineering Department, Hooghly Engineering & Technology College.

We can confidently say that the report made and all the conclusions drawn are the actual outcome of our project work. We also declare that:

- The work contained in the report is original and has been done by us under the generalsupervision of our respected supervisor.
- The work has not been submitted by any other institution for any other purpose in thisuniversity.
- We have thoroughly followed the guidelines provided by the university while writing theproject report.
- We have properly explained every minute thing related to our project work and also giventhe proper sources in the references.

Names:

1. **ARPAN BISWAS**
2. **SUBHRAJYOTI MANDAL**
3. **SHIBAM MISHRA**
4. **ANRTIP MANNA**
5. **RIPA GHOSH**

ACKNOWLEDGEMENT

We are honored in documenting this project report on "**SMART SOIL MONITORING SYSTEM**" as a requirement for the award of B.Tech Degree in Electronics and Communications Engineering. This document is a fruit of labour from the students and the professors. Though we, as the students, give all our joint effort to produce this work piece, it would have been impossible without the extensive help from the teachers' side. It would be great pleasure to express our sincere thanks to **Mr. Subhojit Malik, Assistant Professor, Electronics and Communications Engineering Department, Hooghly Engineering & Technology College** who provided us a helping hand in this project. The suggestion given by him undoubtedly helped. The unflagging patience, enthusiastic guidance and immense knowledge that he shared with us proved highly beneficial and were the sole reason in making our project both possible and successful.

We are obliged to the Head of department of Electronics and Communications Engineering Department, Hooghly Engineering & Technology College, for his cordial support, valuable information and guidance which helped us in completing this task through various stages. The blessing help and guidance given by sir time to time shall carry us a long way in the journey of life. We would also like to thank our institute and other faculty members without whom this project would have been distant reality. The guidance and support from all members who contributed to this project, was vital for the completion of project.

Date: Name and Signature of Group Members

……………………………………….

Arpan Biswas (17600321003)

……………………………………….

Subhrajyoti Mandal (17600321004)

…………………………………………

Shibam Mishra (17600321017)

…………………………………………

Antarip Manna (17600321048)

………………………………………..

Ripa Ghosh (17632322012)

HOOGHLY ENGINEERING & TECHNOLOGY COLLEGE

VIVEKANANDA ROAD, PIPULPATI, HOOGHLY, WEST BENGAL, PIN-712103

Department of Electronics and Communications Engineering

CERTIFICATE

Certified that the Project Work entitled "**SMART SOIL MONITORING SYSTEM**"is a bona fide work carried out by **Arpan Biswas (17600321003)** in fulfillment for the award of Bachelor of Technology (B. Tech) in Electronics and Communications Engineering (ECE) Department of the Maulana Abul Kalam Azad University of Technology (MAKAUT), West Bengal. It is certified that all the corrections/suggestions indicated for internal assessment have been incorporated in the report and deposited in the departmental library. The Project Report has been approved as it satisfies the academic requirement in respect of Project Work prescribe for the said degree.

…………………………….	……………………………………	……………………………......
Mr. Subhojit Malik	Dr. Ankan Bhattacharya	Prof. (Dr.) B. P Pattanaik
Supervisor	Head of the Department	Principal

………………………………………...

Examiner 1

……………………………………………….

Examiner 2

ABSTRACT

With advancement of technology things are becoming simpler and easier for us. Automation plays an increasingly important role in the world economy and in daily experience. Last few decades has witnessed a rapid development in robotic technology. Different types of intelligent machines which facilitate various tasks in industry environment are becoming popular. The project presented here, focuses on designing an "**SMART SOIL MONITORING SYSTEM**".

- This work presents a Smart Soil Monitoring System using IoT Technology designed to enhance agricultural productivity by providing real-time data on key soil parameters. The system integrates various sensors—such as moisture, temperature, pH and nutrient sensors—with a microcontroller and wireless communication modules (e.g., Wi-Fi or LoRa). These sensors collect and transmit data to a cloud-based platform where it is processed, analyzed and made accessible via a web or mobile application.

- Farmers can monitor soil conditions remotely and receive alerts or recommendations for irrigation, fertilization and other interventions. This IoT-based approach minimizes manual effort, reduces resource wastage and enables data-driven decision-making for sustainable farming. The proposed system aims to improve crop yield, optimize water usage and promote environmentally friendly agricultural practices

CONTENTS

INTRODUCTION

In recent years, the need for **sustainable agricultural practices** and **efficient land management** has become increasingly urgent due to rising global population, climate change and the degradation of natural resources. These growing concerns have brought **soil health monitoring** to the forefront of agricultural innovation, as healthy soil is fundamental to crop production, ecosystem balance and long-term environmental sustainability. With the pressure to produce more food while preserving the environment, it is critical to adopt technologies that provide accurate, real-time insights into soil conditions.

The **Smart Soil Health Measuring Project** is a forward-looking initiative designed to tackle these challenges head-on by integrating cutting-edge technologies such as the **Internet of Things (IoT)**, **data analytics** and **real-time monitoring systems**. The primary goal of the project is to provide an advanced yet cost-effective framework for monitoring and improving soil health across different agricultural landscapes. This project envisions a smarter, more data-driven approach to farming where decisions are guided by timely, accurate and comprehensive soil data.

At the core of this project is a **network of sensors** capable of measuring a wide range of critical soil parameters. These include **soil moisture content**, **temperature**, **pH levels**, **electrical conductivity** and **nutrient concentrations**, all of which play a vital role in determining soil fertility and suitability for crop cultivation. By utilizing affordable, open-source hardware platforms such as **Raspberry Pi** and **Arduino**, the project ensures that the technology remains **accessible**, **customizable** and **scalable**, making it suitable not only for large-scale agricultural operations but also for smallholder farmers, researchers and environmentalists in both rural and urban settings.

One of the most important aspects of this initiative is the **deployment of soil moisture sensors**, which continuously collect data on the amount of water present in the soil. This real-time information helps farmers better understand when and how much to irrigate their fields, thus **minimizing water waste**, improving crop health and contributing to **water conservation efforts**. In regions where water is scarce or costly, this can lead to significant economic and environmental benefits. Furthermore, the integration of **temperature sensors** allows for close monitoring of soil thermal conditions, which is crucial for understanding **plant growth cycles**, **germination timing** and **microbial activity**, all of which affect overall soil vitality.

In conclusion, the **Smart Soil Health Measuring Project** represents a transformative step forward in the pursuit of **sustainable agriculture**. By harnessing modern technologies to monitor, analyse and interpret soil health data, the project lays the foundation for **smarter decision-making**, **improved agricultural productivity** and **greater environmental stewardship**. As the agricultural sector faces growing challenges from **climate change**, **resource scarcity** and **population growth**, innovative solutions like this are essential. This project not only supports the health and resilience of our soils but also paves the way for a more secure and sustainable food system for future generations.

WORKING PRINCIPLE

The Smart Soil Health Measuring Project is designed to monitor and analyze key soil health parameters using a combination of sensors, microcontrollers and communication technologies. The fundamental working principle is based on real-time data collection, digital signal processing, wireless data transmission and intelligent decision-making to improve agricultural productivity and sustainability. This system makes use of modern electronics and open-source platforms like Raspberry Pi and Arduino, enabling a cost-effective and scalable solution for a wide range of users including farmers, researchers and environmentalists.

1. Sensor Integration and Placement

At the core of the project lies a network of soil sensors strategically placed in the field. These sensors are responsible for continuously measuring various essential soil parameters:

- **Soil Moisture Sensor:** Detects the amount of water content in the soil. This is crucial for irrigation planning.
- **Temperature Sensor (e.g., DHT11 or DS18B20):** Measures the current soil temperature, which affects seed germination and microbial activity.
- **pH Sensor:** Measures the acidity or alkalinity of the soil, which determines nutrient availability to plants.
- **Electrical Conductivity (EC) Sensor:** Measures the ion concentration, helping assess soil salinity.
- **NPK Sensor (Nitrogen, Phosphorus, Potassium**): Provides nutrient concentration levels, essential for determining soil fertility.

These sensors are embedded at appropriate depths and positions in the soil to ensure accurate and representative readings.

2. Data Acquisition and Signal Conditioning

Each sensor generates a signal (analog or digital) corresponding to the parameter it measures. These signals are then sent to a microcontroller unit, such as Raspberry Pi Pico W or an Arduino board.

If the sensor sends analog signals, an Analog-to-Digital Converter (ADC) is used to convert them into digital values.

Digital data is then filtered and processed using software libraries or pre-set calibration curves to transform raw values into real-world units (e.g., temperature in °C, moisture in %, pH level).

This step ensures that the collected data is clean, accurateand ready for analysis.

3. **Central Processing and Control**

The microcontroller serves as the central processing unit of the system. It performs several critical tasks:

- Continuously collects and stores real-time data from all connected sensors.
- Processes data using embedded algorithms to detect trends or thresholds.
- Triggers alerts or automated responses (e.g., turning on a water pump if moisture is low).

In some versions, the data is also logged over time for future reference and trend analysis.

4. **Display and User Feedback**

To provide immediate feedback to users, the processed data is displayed on a 0.96-inch OLED display. Each parameter (moisture, temperature, pH, etc.) is shown clearly on the screen, allowing users to:

- Monitor soil health in real time.
- Make manual decisions based on current readings.
- Take timely actions like irrigating or fertilizing crops.
- This makes the system user-friendly, even for farmers with minimal technical background.

5. **Wireless Communication and Cloud Integration**

If the system includes a microcontroller with Wi-Fi capability (such as Raspberry Pi Pico W), the data can be transmitted wirelessly to a cloud platform, mobile app, or web dashboard. This allows:

- Remote monitoring of soil conditions from any location.
- Storage and analysis of historical data.
- Alerts and notifications to be sent directly to the user's phone or computer.
- This feature adds great value in large-scale farms or when the farmer is not physically present in the field.

6. Data Analysis and Decision Making

With access to real-time and historical data, farmers or agricultural experts can analyze trends and make informed decisions. Examples include:

- Scheduling irrigation based on soil moisture trends to reduce water wastage.
- Adjusting fertilizer type and quantity based on NPK levels.
- Modifying planting cycles according to temperature and pH data.
- Advanced versions of this project can even integrate machine learning algorithms to provide smart recommendations and predict future soil behavior.

CONNECTION DIAGRAM

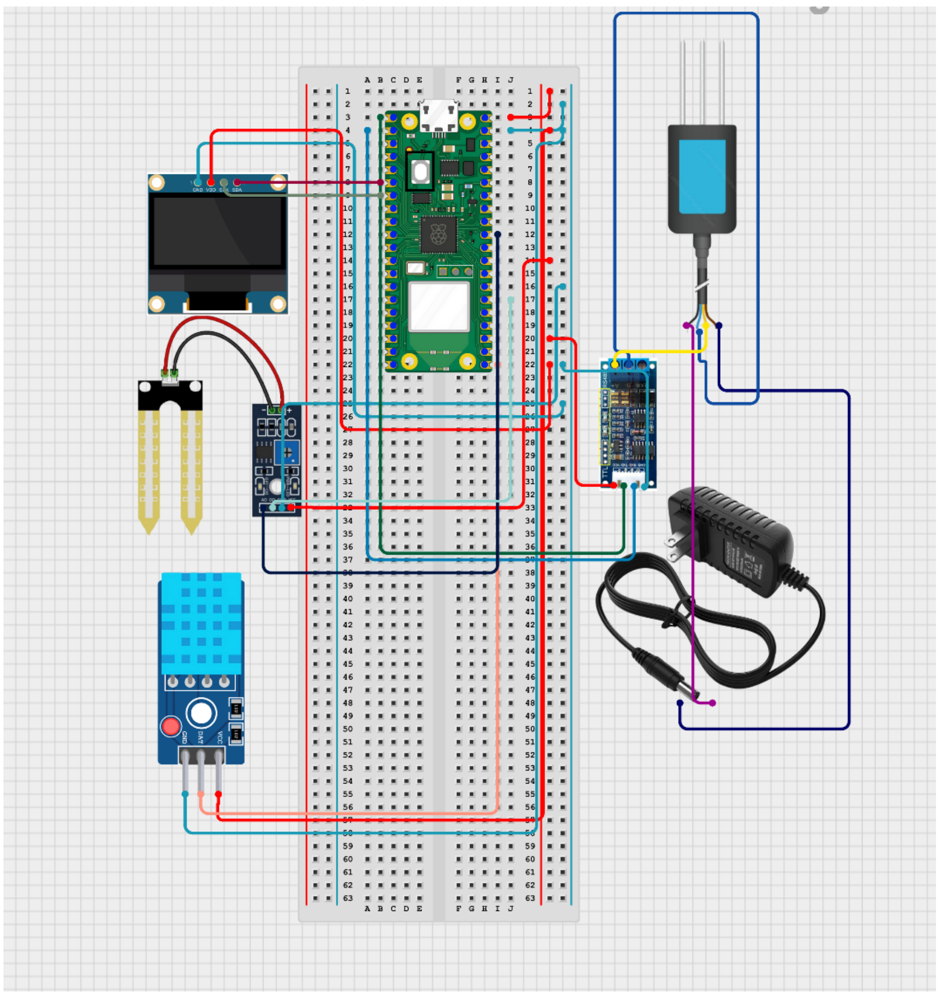


Fig. 1: Connection Diagram

PROPOSED MODEL

Smart Soil Health Measuring Project is centered around a networked system of sensors and IoT-enabled devices strategically deployed across agricultural fields. The model leverages low-cost, open-source hardware such as Raspberry Pi and Arduino to create a robust, scalable and accessible solution tailored for diverse agricultural needs. This sensor network is designed to measure crucial soil parameters, including moisture content, temperature, pH, electrical conductivity and nutrient levels, offering a comprehensive insight into soil health dynamics.

The model begins with soil moisture sensors, which provide real-time data on water availability within the soil, allowing for optimized irrigation practices. By continuously tracking moisture levels, farmers can make informed decisions on when and how much to irrigate, minimizing water waste while ensuring that crops receive the necessary hydration. Additionally, temperature sensors within the model monitor variations in soil temperature, a critical factor that influences both plant growth rates and microbial activity.

To support nutrient management, pH and electrical conductivity sensors are integrated to detect soil acidity levels and nutrient concentrations, enabling precision in fertilizer application. This data is transmitted in real-time to a central database, where it is analyzed using data analytics tools. Through this centralized system, farmers can access an intuitive, user-friendly dashboard on mobile or desktop, providing actionable insights for informed decision-making.

The proposed model is highly modular, allowing for easy scaling and customization based on field size, crop type and local conditions. This adaptability makes the Smart Soil Health Measuring Project an innovative and accessible solution for modern agricultural challenges, with the potential to significantly enhance resource efficiency, crop yield and sustainable land management.

Components Required

SL.	Component	Specification	Cost
1.	Raspberry Pi Pico	Dual core ARM Cortex- M0+processor running up to 133 MHz with 264 KB of on-chip SRAM	350.00
2.	DHT 11 SENSOR	The sensor can measure temperature from 0°C to 50°C and humidity from 20% to 90% with an accuracy of ±1°C and ±1%.	50.00
3.	BREADBOARD	A breadboard is commonly rated for five volts at one amp or fifteen volts at one-third of an amp	50.00
4.	JUMPER WIRE	The nominal conductor diameter shall be 0.6mm with a tolerance of +0.01mm. Each conductor shall be uniformly covered with fire-resistive PVC insulation.	30.00
5.	OLED-0.96”	Resolution is 4K (3840 x 2160 pixels) or 8K (7680 x 4320 pixels) for high-end models	130.00
6.	NPK SENSOR	High measurement accuracy, fast response speedand good interchangeability.	2900.0
7.	TTL Convertor	Built-in USB to TTL Transfer chip. Designed to be used for USB to TTL electronic projects.	50.00
8.	12-volt Charger	12 Volts DC (regulated) 100–240 Volts AC (universal input) Typically 0.5A to 5A depending on the use case (e.g., 12V/1A, 12V/2A, 12V/5A)	150.00

FUNCTIONAL DESCRIPTION

➢ ***RASPBERRY PI PICO*:**

Raspberry Pi Pico W brings wireless connectivity to the best- selling Raspberry Pi Pico product line. Built around our RP2040 silicon platform, Pico products bring our signature values of high performance, low cost and ease of use to the microcontroller space. With a large on-chip memory, symmetric dual-core processor complex, deterministic bus fabric and rich peripheral set augmented with our unique Programmable I/O (PIO) subsystem, RP2040 provides professional users with unrivalled power and flexibility. Offering detailed documentation, a polished Micro Python port and a UF2 bootloader in ROM, it has the lowest possible barrier to entry for beginner and hobbyist users. RP2040 is manufactured on a modern 40nm process node, delivering high performance, low dynamic power consumption and low leakage, with a variety of low-power modes to support extended-duration operation on battery power. Raspberry Pi Pico W offers 2.4GHz 802.11 b/g/n wireless LAN support and Bluetooth 5.2, with an on-board antenna and modular compliance certification. It is able to operate in both station and access-point modes. Full access to network functionality is available to both C and Micro Python developers. Raspberry Pi Pico W pairs RP2040 with 2MB of flash memory and a power supply chip supporting input voltages from 1.8–5.5V. It provides 26 GPIO pins, three of which can function as analogue inputs, on 0.1”-pitch through-hole pads with castellated edges.

Specification:

- Form factor: 21 mm × 51 mm
- CPU: Dual-core Arm Cortex-M0+ @ 133MHz
- Memory: 264KB on-chip SRAM; 2MB on-board QSPI flash
- Interfacing: 26 GPIO pins, including 3 analogue inputs
- Peripherals: • 2 × UART
    - 2 × SPI controllers
    - 2 × I2C controllers
    - 16 × PWM channels
    - 1 × USB 1.1 controller and PHY, with host and device support
    - 8 × PIO state machines
- Connectivity: 2.4GHz IEEE 802.11b/g/n wireless LAN, on-board antenna
- Bluetooth 5.2

    - Support for Bluetooth LE Central and Peripheral roles.
    - Support for Bluetooth Classic
- Input power: 1.8–5.5V DC.
- Operating temperature: -20°C to +70°

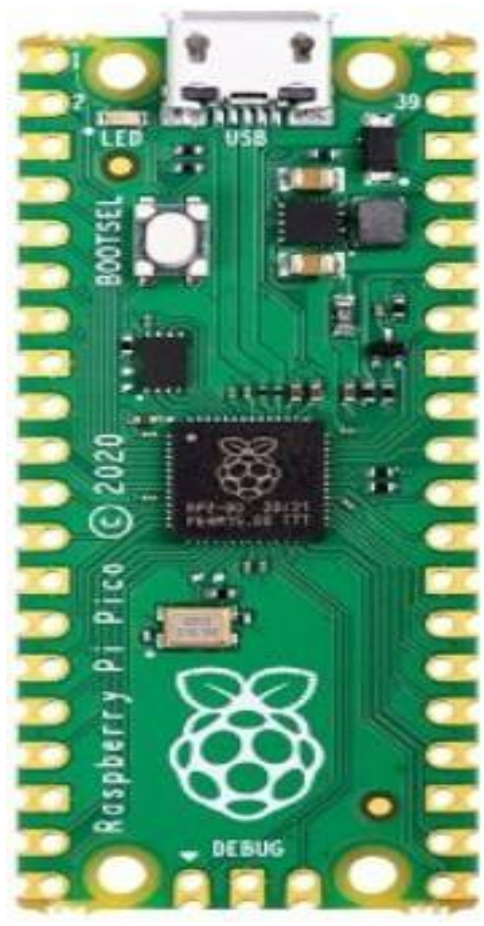


Fig. 2: Rasberry PI Pico

NPK SENSOR

NPK sensors are specialized devices used in agriculture to measure soil nutrient levels, specifically nitrogen (N), phosphorus (P)and potassium (K), which are essential for plant growth. By monitoring these nutrients, NPK sensors help farmers and researchers optimize fertilization, ensuring crops receive the right nutrients for healthy growth while avoiding over-fertilization that can harm the environment.

These sensors typically use optical or electrochemical methods to detect nutrient levels in the soil. They provide real-time or near-real-time data, making them valuable for precision agriculture. NPK sensors can integrate with other IoT devices, enabling automated nutrient management for sustainable farming practices.

Features:

1. 4-20mA Output Soil Sensor
2. 2M Cable with 3-pin Probes
3. Measures Soil Moisture, Temperatureand Humidity
4. Suitable for Agriculture and Environmental Monitoring
5. 4-20mA Output for Compatibility

Fig 3:NPK Sensor

TTL Convertor

A TTL (Transistor-Transistor Logic) converter is a device used to adapt voltage levels between different systems, typically converting TTL-level signals (usually 5V or 3.3V) to other logic levels, such as RS-232 or USB. TTL converters are essential for communication between microcontrollers, computers and peripheral devices that operate at different voltage standards.

Common types include TTL to USB converters, which enable microcontrollers to communicate with computers via USB and TTL to RS-232 converters, which allow for serial communication. TTL converters are widely used in embedded systems, robotics and DIY electronics projects to ensure reliable, error-free data transfer across various components.

TTL(Serial) Converter Module is two data transmission indicators can monitor data transfer status in real-time. It can Adopt imported controller RS232 TTL, which can stabilize the flash with high-speed 500mA self-recovery fuse for protection.

Features

1. Built-in USB to TTL Transfer chip.
2. Designed to be used for USB to TTL electronic projects.
3. TTL interface output, easy to connect to your MCU.
4. Dual 3.3V and 5V Power output, work with 3.3v and 5 V target device.
5. The mini module is designed specifically for STC download and ARDUINO PRO supports all series of STC
6. Supports WIN7/VISTA/MAC/LINUX (32 bit /64-bit system)

.

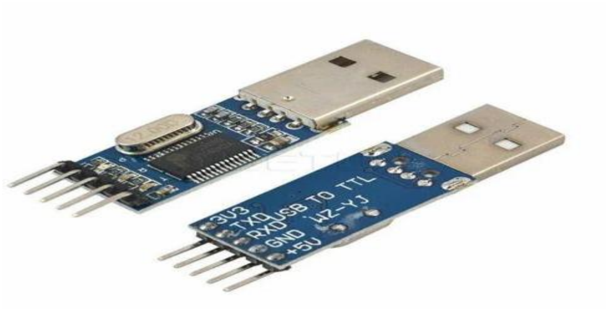


Fig 4: TTL Convertor

DTH11 SENSOR

The DHT11 is a commonly used Temperature and humidity sensor that come with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as data.

DHT11 Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. By using the exclusive digital-signal- acquisition technique and temperature & humidity sensing technology, it ensures high reliability and excellent long-term stability. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component and connects to a high performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness.

➢ DHT11 Specifications

- Operating Voltage: 3.5V to 5.5V
- Operating current: 0.3mA (measuring) 60uA (standby)
- Output: Serial data
- Temperature Range: 0°C to 50°C
- Humidity Range: 20% to 90%
- Resolution: Temperature and Humidity both are 16-bit
- Accuracy: ±1°C and ±1%

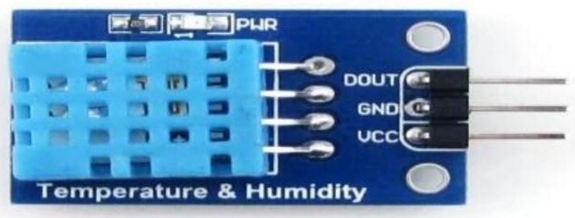


Fig 5 :DHT11 Sensor

BREADBOARD

A breadboard is a reusable platform for building and testing electronic circuits without soldering. It consists of a grid of interconnected holes where components like resistors, capacitors and ICs can be easily inserted. The holes are connected in rows and columns with metal strips beneath, allowing parts to connect electrically.

Breadboards are commonly used in prototyping because they let users modify circuits quickly and test components. Typically organized with power rails along the sides for power distribution and a central area divided into rows, they enable rapid, solder-free assembly. Breadboards are widely used in electronics.

education, experimentation and development.

➢ Features and Specifications

- 2 Distribution Strips, 200 tie-points.
- 630 tie-points in IC/ circuit areas
- ABS plastic with color legend
- Dimension: 6.5*4.4*0.3 inch
- Hole/Pitch Style: Square wire holes (2.54mm)
- ABS heat Distortion Temperature: 84° C (183° F)
- Rating: 300/3 to 5Amps
- Insulation Resistance: 500MΩ / DC500V
- Withstanding Voltage: 1,000V AC / 1 minute

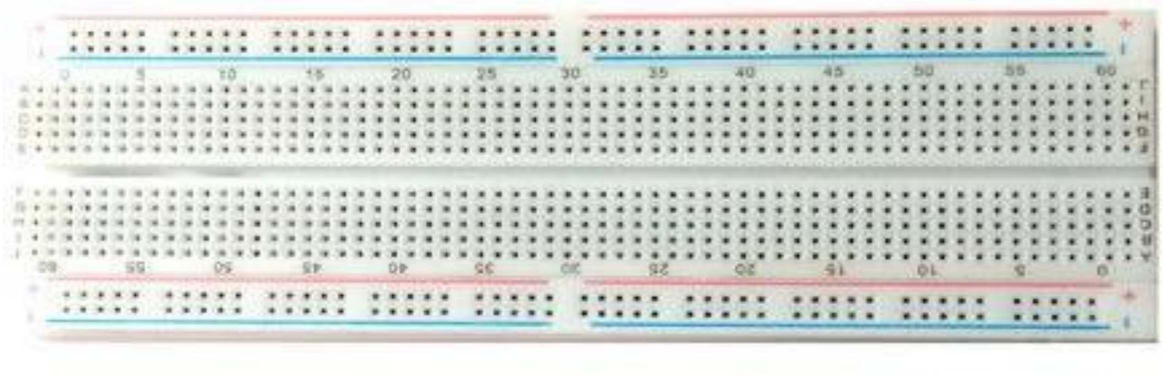


Fig 6 : Breadboard

OLED-96

The OLED-0.96" is a small, high-contrast display module often used in electronics projects, featuring a 0.96-inch OLED screen. It typically has a resolution of 128x64 pixels, providing clear and crisp visuals with a wide viewing angle. Unlike LCDs, OLED displays don't require a backlight; each pixel emits its own light, allowing for deeper blacks and lower power consumption, making it suitable for battery- powered devices.

The OLED-0.96" display usually connects via I2C or SPI interfaces, making it compatible with microcontrollers like Arduino, Raspberry Pi and ESP32. This

display is popular for wearable devices, compact user interfaces and DIY electronics, where space is limited but clarity is essential. Its small size, high brightness and flexibility in displaying graphics and text make it ideal for many creative applications.

➢ Features:

- 96 x 64 resolution, 65K true to life colors, PMOLED screen.
- 0.96" diagonal size, 32.7 x 23 x 4.9mm. Active Display Area: 20mm x 14mm.
- No back lighting with near 180° viewing angle.
- Easy 5 pin interface to any host device: VCC, TX, RX, GND, RESET.
- Serial TTL interface with auto-baud feature (300 to 256K baud).
- Powered by the 4D-Labs GOLDELOX-SGC processor (also available as separate OEM IC for volume users).
- On-board micro-SD memory card adaptor for storing of icons, images, animations, etc. Supports 64Mb to 2Gig micro-SD memory cards.
- Comprehensive set of built-in high-level graphics functions and algorithms that can draw lines, circles, textand much more.
- Display full color images, animations, icons and video clips.
- Supports all available Windows fonts and characters (imported as external fonts).
- Multiple switch/buttons feature on a single pin.
- 4.0V to 5.5V range operation (single supply).
- RoHS Compliant

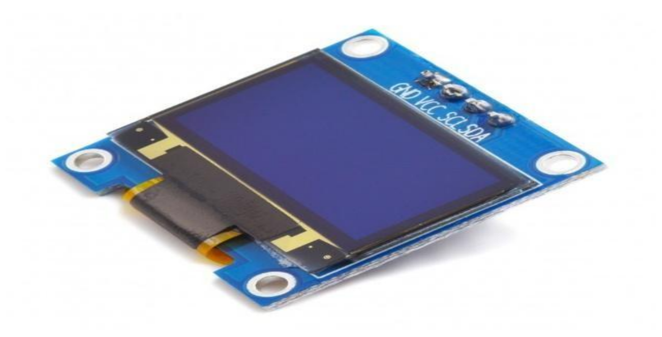


Fig 7: OLED Display

JUMPER WIRES

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools to make it easy to change a circuit as needed. Fairly simple. In fact, it doesn't get much more basic than jumper wires.

Though jumper wires come in a variety of colors, the colors don't mean anything. This means that a red jumper wire is technically the same as a black one. But the colors can be used to your advantage to differentiate between types of connections, such as ground or power.

Jumper wires typically come in three versions: male-to-male, male-to-female and female-to-female. The difference between each is in the end point of the wire. Male ends have a pin protruding and can plug into things, while female ends do not and are used to plug things into. Male-to-male jumper wires are the most common and what you likely will use most often. When connecting two ports on a breadboard, a male-to- interrupted.

Key features of the DS3231 include its I2C interface, which makes it compatible with microcontrollers like Arduino and Raspberry Pi. It also has an onboard battery backup, allowing it to maintain time even when power is off, making it ideal for clocks, data loggers and any project that requires reliable time-tracking. The DS3231 is known for its accuracy, with minimal drift over time, even under varying environmental conditions.

➢ Features

- 1 Hz output pin SQW.
- 32 KHz output pin 32K.
- Voltage Supply: 2.2 V ~ 5.5 V (for RTC).
- Time Format: HH: MM: SS (12/24 hr.).
- Date Format: YY-MM-DD-dd.
- DS 3231 based RTC with 2032 Battery Holder.

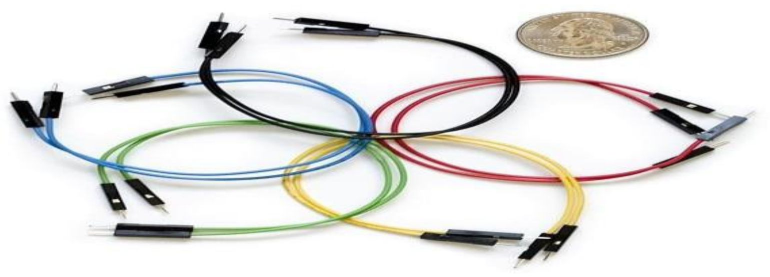


Fig 8: Jumper wires

SOFTWARE COMPONENTS

Android Studio:

Android Studio is the official integrated development environment (IDE) for building Android applications. In the context of a smart soil monitoring system android Studio plays a crucial role in developing a mobile application that allows users to interact with the system directly from their smartphones. This application can be designed to capture images of different soil types using the phone's camera and then analyze or display the soil's specifications.

Using Android Studio, developers can write code in languages like Java or Kotlin to create a user-friendly interface where users can take photos of soil samples. These images can be processed locally or uploaded to a server or cloud platform for further analysis using machine learning models. Once analyzed, the app can display important characteristics of the soil, such as texture, color, moisture level, or the presence of organic material.

Additionally, the app can be connected to a cloud database like Supabase to fetch or store soil specifications and sensor readings in real time. Android Studio supports various libraries and APIs that make it possible to integrate camera functionality, image processing, cloud communication and data visualization all within one application.

This allows farmers, researchers, or students to use the mobile app as a portable tool for soil analysis, getting instant feedback and maintaining records without needing a desktop system. Overall android Studio enables the creation of a powerful and efficient application that enhances the practicality and accessibility of the smart soil monitoring project.

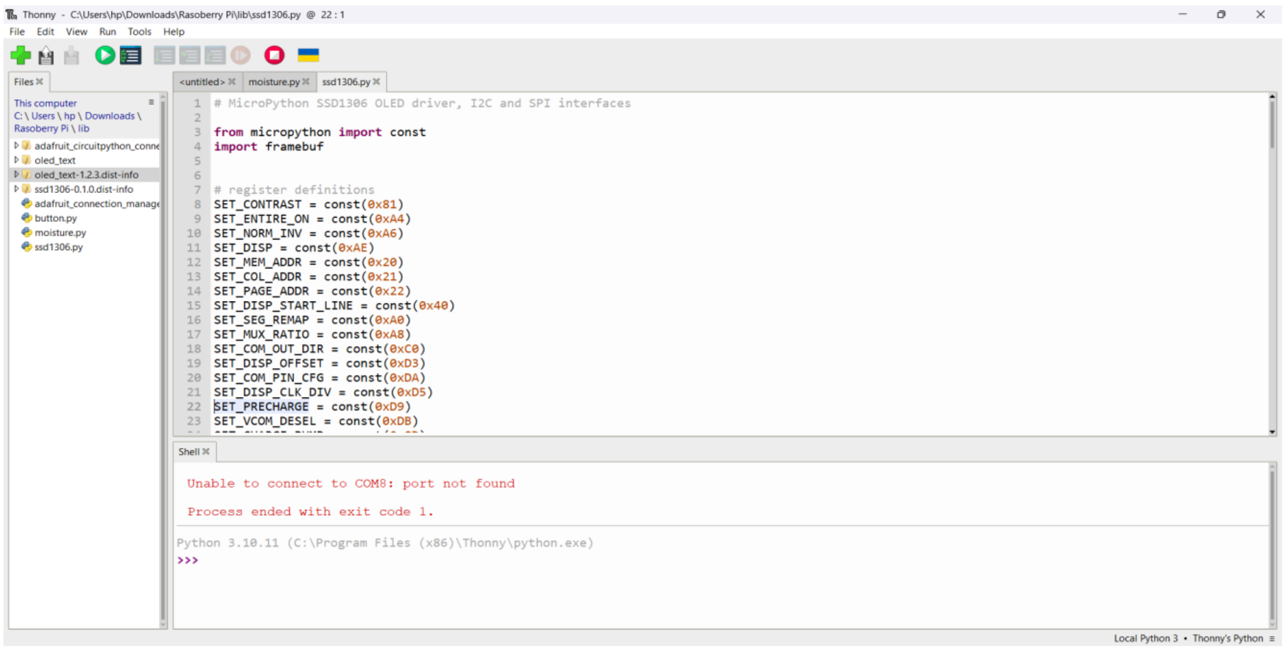
Thonny IDE:

Thonny IDE is a lightweight and beginner-friendly integrated development environment specifically designed for writing and running Python and MicroPython code. It is widely used for programming microcontrollers like the Raspberry Pi Pico. In a smart soil monitoring system, Thonny IDE is particularly useful because it allows developers to write, testand upload sensor-interfacing code directly to the microcontroller using MicroPython.

For this project, Thonny can be used to write code that reads data from sensors such as humidity, temperature (e.g., using DHT11 or DHT22)and NPK (Nitrogen, Phosphorus, Potassium) soil nutrient sensors. The IDE provides a simple interface where you can connect the Pico via USB, open a script and upload it with just a few clicks. It also features a built-in shell to see real-time outputs from the sensors, which helps in debugging and verifying if the sensors are working correctly.

The humidity and temperature sensor code typically involves importing a MicroPython DHT library and using the Pico's GPIO pins to trigger readings. For the NPK sensor, analog or UART communication is used depending on the sensor typeand Thonny allows real-time monitoring of the values through serial output.

Thonny's simplicity makes it ideal for both beginners and advanced users, providing a quick and efficient environment to develop, test and run the core logic of the smart soil monitoring system. It serves as the bridge between the physical sensors and the digital system, ensuring accurate and live data collection before that data is sent to the cloud or displayed in an app.


Thonny - C:\Users\hp\Downloads\Rasoberry Pi\lib\ssd1306.py @ 22 : 1
File Edit View Run Tools Help
Files
This computer
C: \ Users \ hp \ Downloads \ Rasoberry Pi \ lib
adafruit_circuitpython_conne
oled_text
oled_text-1.2.3.dist-info
ssd1306-0.1.0.dist-info
adafruit_connection_manage
button.py
moisture.py
ssd1306.py
<untitled>
moisture.py
ssd1306.py
Shell
Local Python 3 • Thonny's Python

```
# MicroPython SSD1306 OLED driver, I2C and SPI interfaces

from micropython import const
import framebuf


# register definitions
SET_CONTRAST = const(0x81)
SET_ENTIRE_ON = const(0xA4)
SET_NORM_INV = const(0xA6)
SET_DISP = const(0xAE)
SET_MEM_ADDR = const(0x20)
SET_COL_ADDR = const(0x21)
SET_PAGE_ADDR = const(0x22)
SET_DISP_START_LINE = const(0x40)
SET_SEG_REMAP = const(0xA0)
SET_MUX_RATIO = const(0xA8)
SET_COM_OUT_DIR = const(0xC0)
SET_DISP_OFFSET = const(0xD3)
SET_COM_PIN_CFG = const(0xDA)
SET_DISP_CLK_DIV = const(0xD5)
SET_PRECHARGE = const(0xD9)
SET_VCOM_DESEL = const(0xDB)
```

```
Unable to connect to COM8: port not found

Process ended with exit code 1.

Python 3.10.11 (C:\Program Files (x86)\Thonny\python.exe)
>>>
```

Fig 9: Thonny IDE

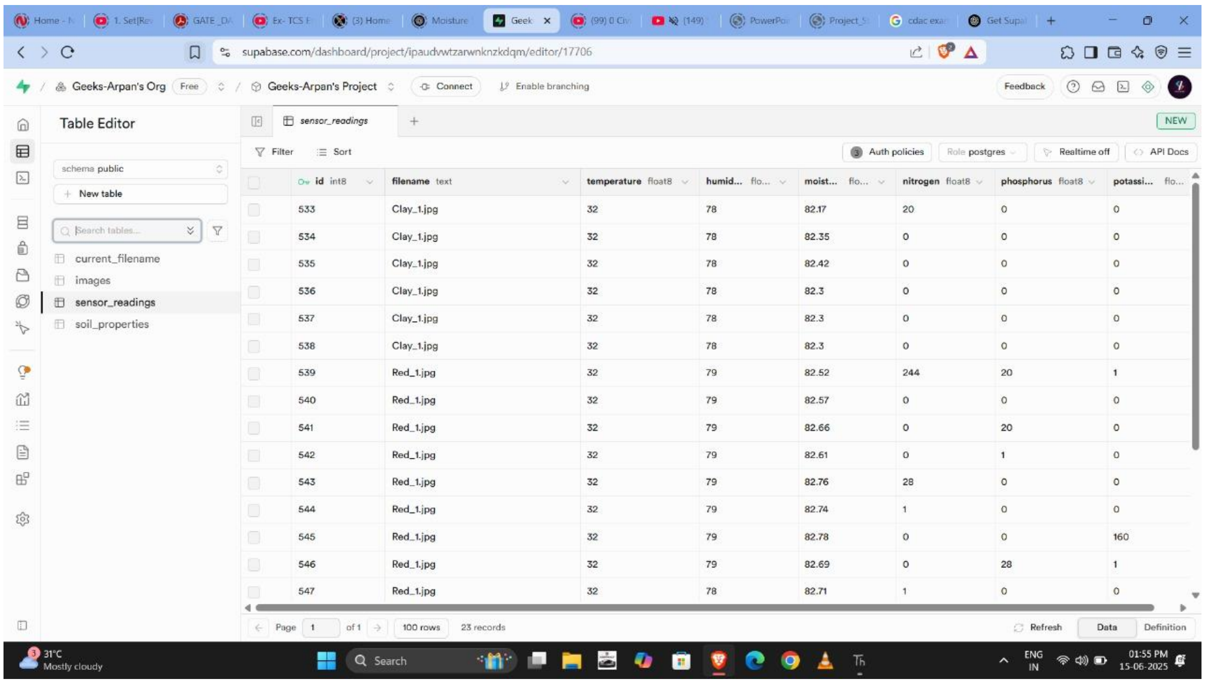
Supa base:

Supabase is an open-source backend-as-a-service platform that provides a real-time PostgreSQL database, authentication, storageand other backend features through a developer-friendly interface. In a smart soil monitoring system, Supabase can be used to store live sensor data efficiently and accessibly.

When sensors connected to a microcontroller like Raspberry Pi Pico collect data such as soil moisture, temperature, humidity, or pH levels, this data can be sent through a Wi-Fi module to a server or directly to the Supabase database using RESTful APIs or client libraries. Each data reading is structured and pushed to a specific table in Supabase, where it gets stored with time stamps, device identifiers, or location tags. Supabase automatically handles the database operations, making it easy to insert, read, or update live data.

One of the key benefits of using Supabase is its real-time feature. Whenever new sensor data is added, it can immediately trigger updates to connected dashboards or mobile apps. This allows farmers or users to monitor soil conditions in real time, receive alertsand make decisions remotely. Developers can also build simple web or mobile interfaces using Supabase's JavaScript client to display the data, control devices, or visualize trends over time.

Overall, Supabase simplifies the process of storing and managing live sensor data in a smart soil monitoring system, making it highly suitable for modern IoT-based agriculture projects.



id int8	filename text	temperature float8	humid... flo...	moist... flo...	nitrogen float8	phosphorus float8	potassi... flo...
533	Clay_1.jpg	32	78	82.17	20	0	0
534	Clay_1.jpg	32	78	82.35	0	0	0
535	Clay_1.jpg	32	78	82.42	0	0	0
536	Clay_1.jpg	32	78	82.3	0	0	0
537	Clay_1.jpg	32	78	82.3	0	0	0
538	Clay_1.jpg	32	78	82.3	0	0	0
539	Red_1.jpg	32	79	82.52	244	20	1
540	Red_1.jpg	32	79	82.57	0	0	0
541	Red_1.jpg	32	79	82.66	0	20	0
542	Red_1.jpg	32	79	82.61	0	1	0
543	Red_1.jpg	32	79	82.76	28	0	0
544	Red_1.jpg	32	79	82.74	1	0	0
545	Red_1.jpg	32	79	82.78	0	0	160
546	Red_1.jpg	32	79	82.69	0	28	1
547	Red_1.jpg	32	78	82.71	1	0	0

Fig 10: Supabase Interface

IMPLEMENTATION

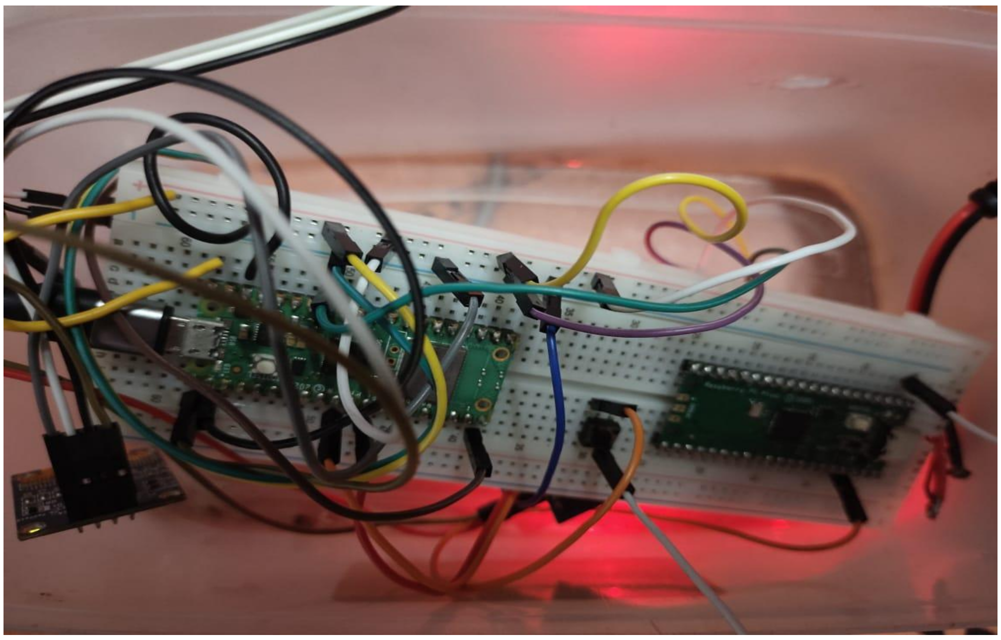


Fig 11a: Implementation of the model

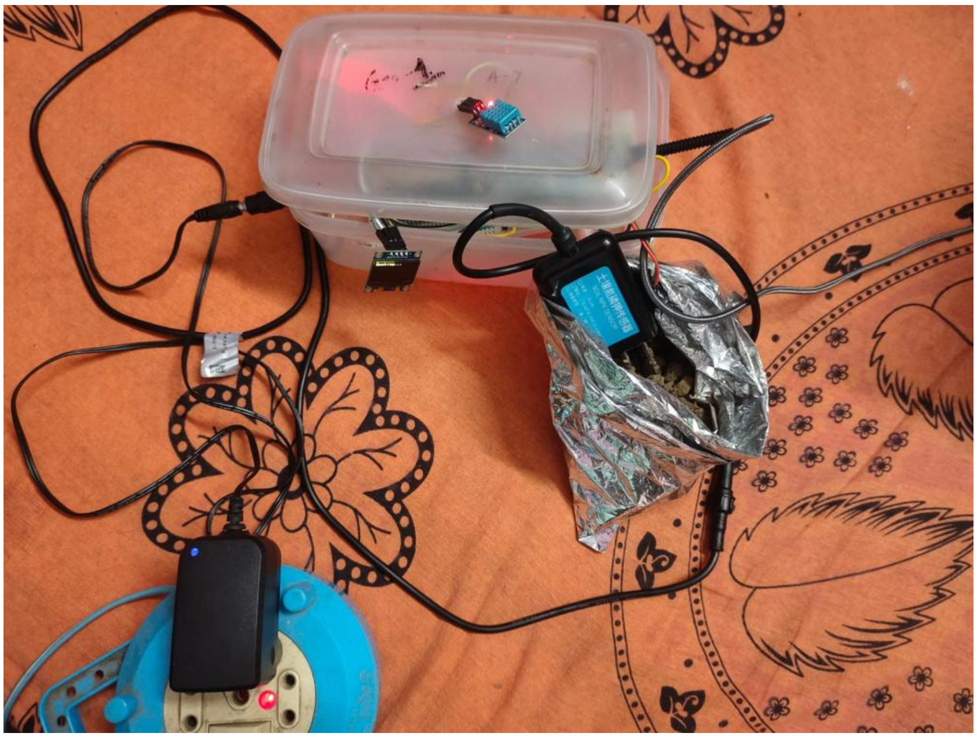
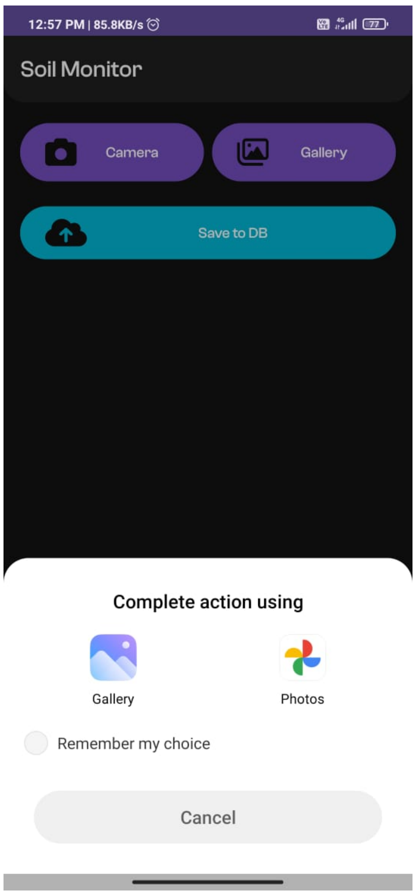


Fig 11b: Overall prototype of the model

APPLICATION INTERFACE



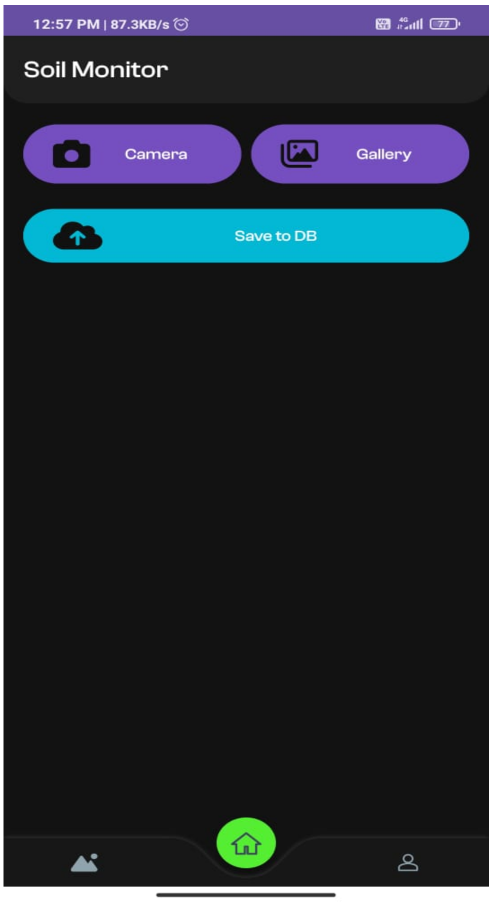


Fig 12: Soil Monitor App

The Soil Monitor App interface is simple and user-friendly. The main screen has three primary buttons:

Camera: To capture a real-time image of the soil using the device camera.

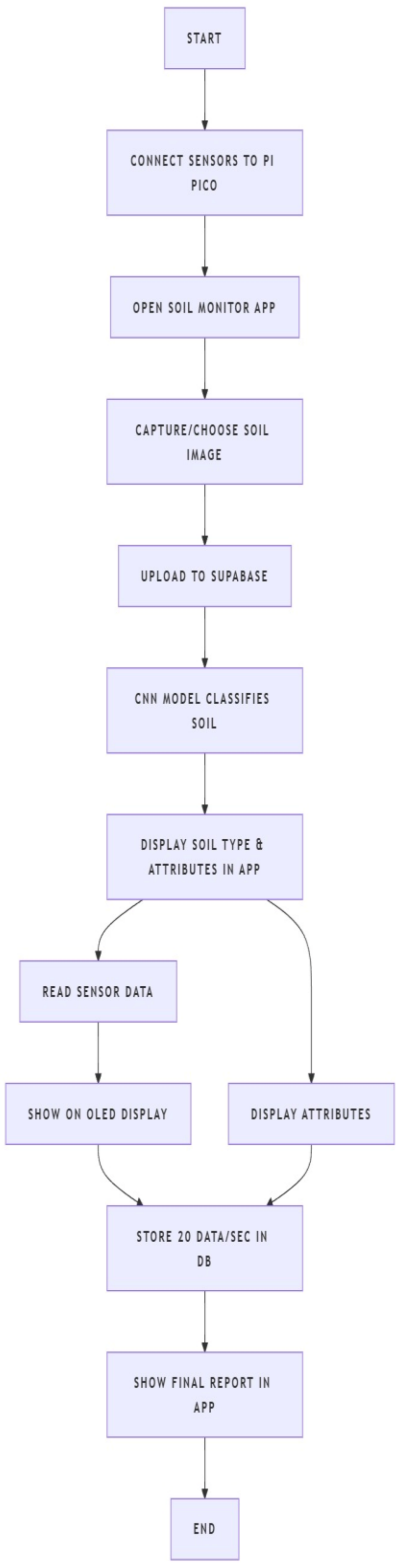
Gallery: To select an existing soil image from the phone's storage.

Save to DB: To upload the selected image to the Supabase database.

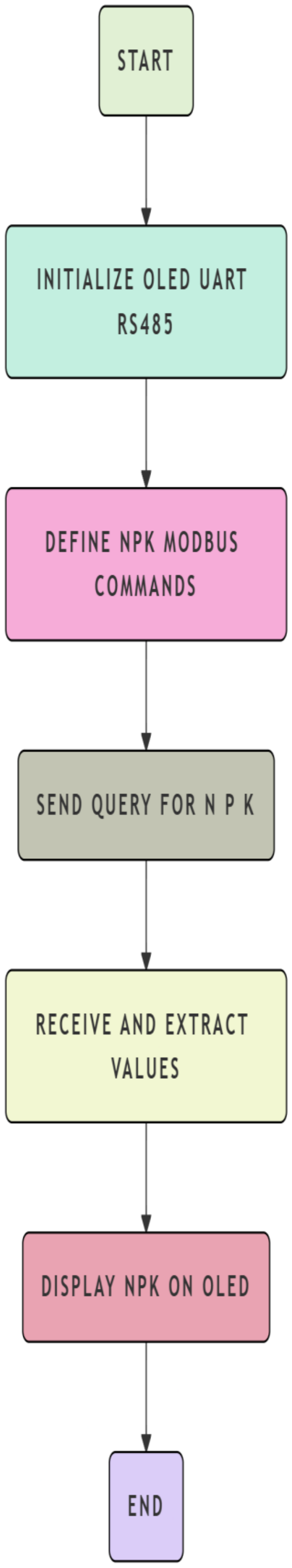
At the bottom, there is a navigation bar with icons for image view, home screen and user profile. When selecting an image from the gallery, users can choose between their default gallery app or Google Photos. This clean and modern layout makes it easy for users to collect and upload soil images for analysis.

ALOGRITHM

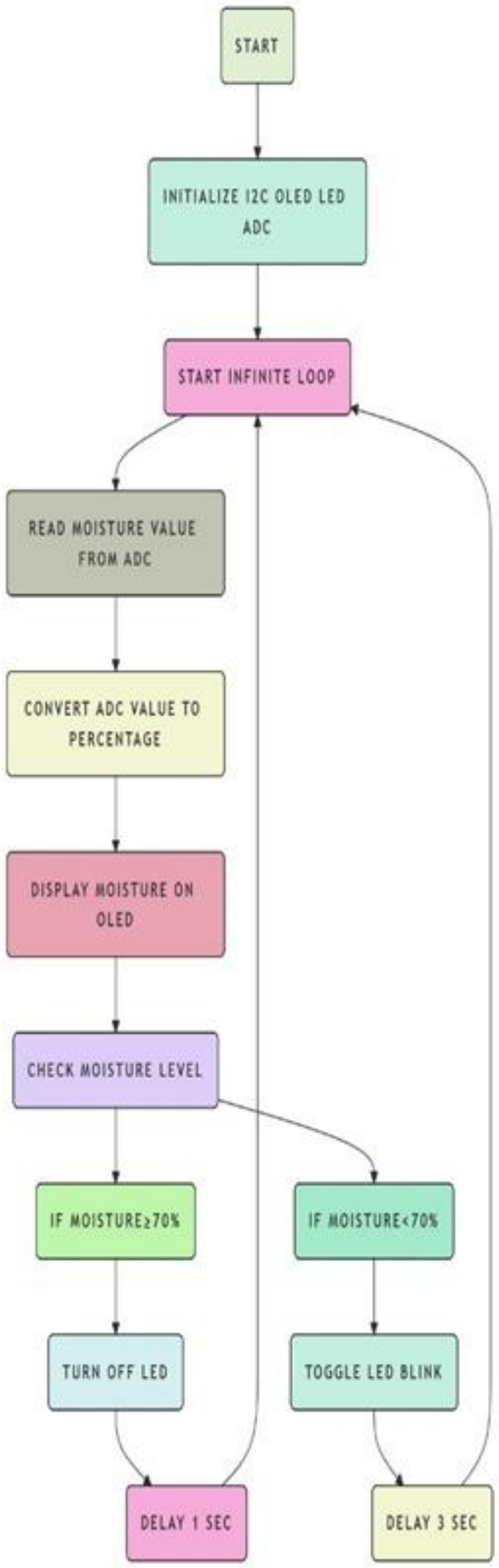
ALGORITHM OF THE PROJECT



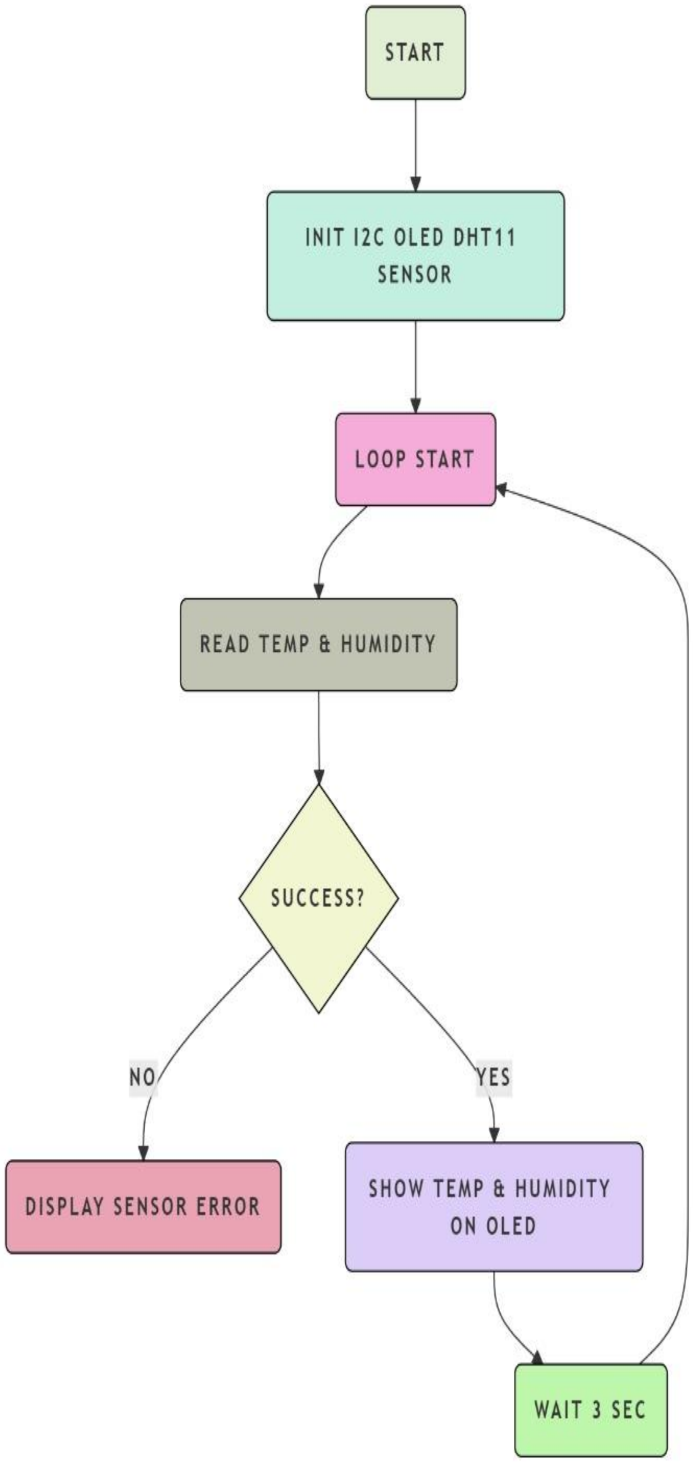
ALGORITHM FOR NPK SENSOR


START
INITIALIZE OLED UART RS485
DEFINE NPK MODBUS COMMANDS
SEND QUERY FOR N P K
RECEIVE AND EXTRACT VALUES
DISPLAY NPK ON OLED
END

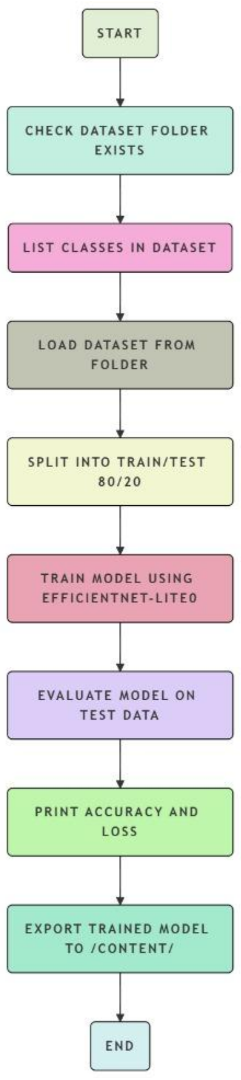
ALGORITHM FOR MOISTURE SENSOR



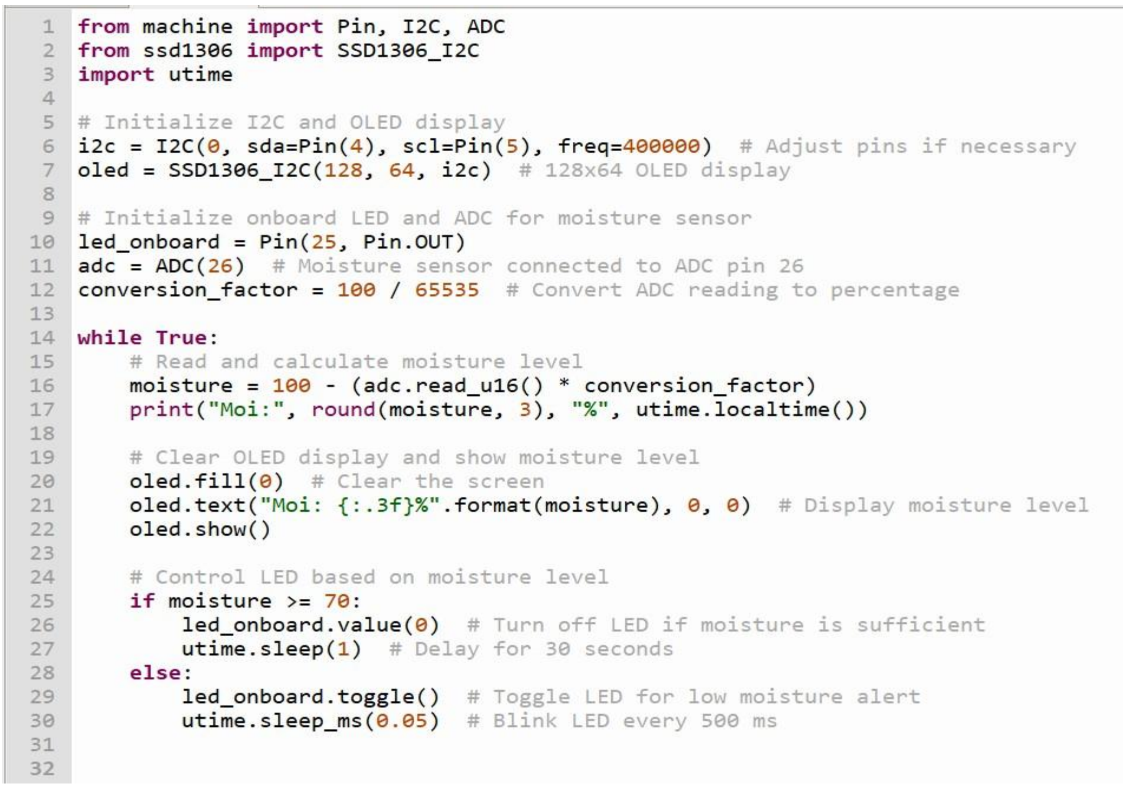
ALGORITHM FOR TEMPERATURE AND HUMIDITY SENSOR



ALGORITHM FOR IMAGE ANALYSIS

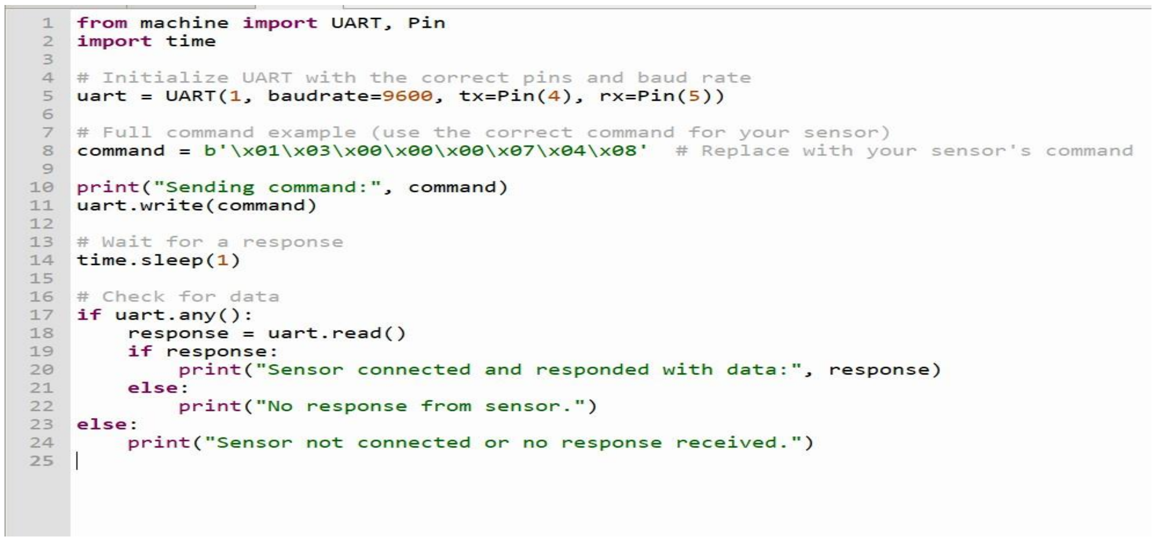

START
CHECK DATASET FOLDER EXISTS
LIST CLASSES IN DATASET
LOAD DATASET FROM FOLDER
SPLIT INTO TRAIN/TEST 80/20
TRAIN MODEL USING EFFICIENTNET-LITE0
EVALUATE MODEL ON TEST DATA
PRINT ACCURACY AND LOSS
EXPORT TRAINED MODEL TO /CONTENT/
END

PROGRAM



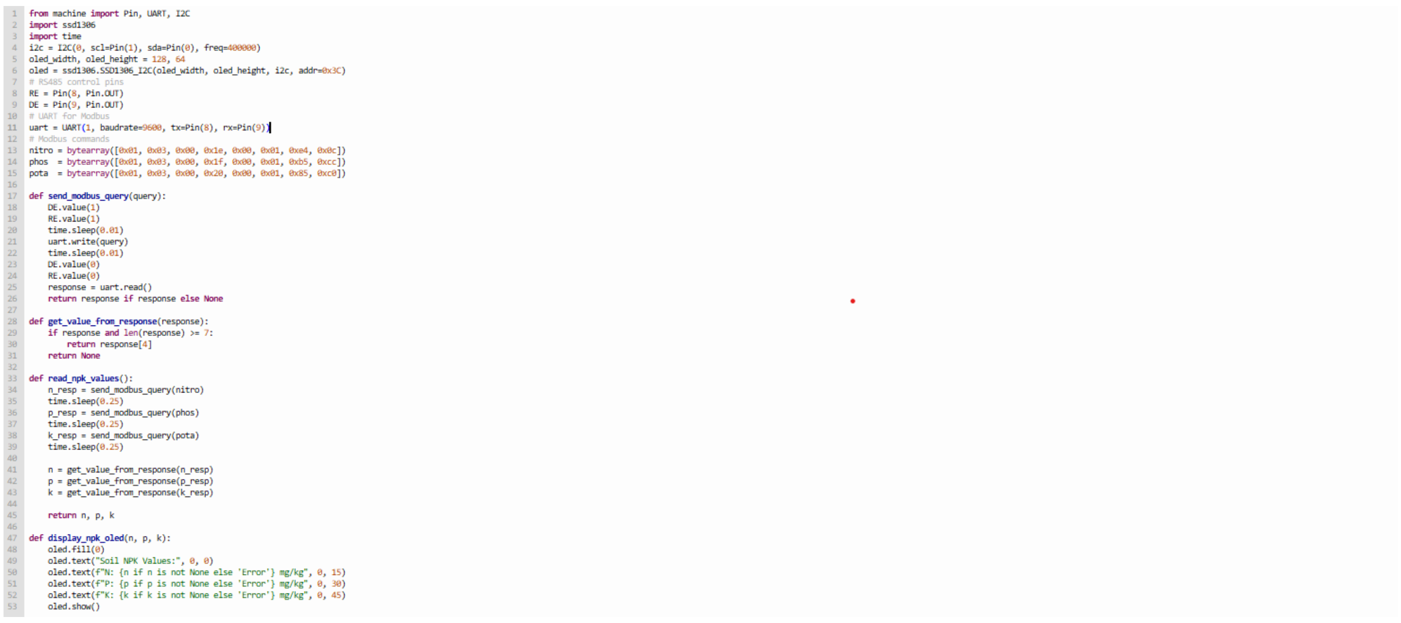
```python
from machine import Pin, I2C, ADC
from ssd1306 import SSD1306_I2C
import utime

# Initialize I2C and OLED display
i2c = I2C(0, sda=Pin(4), scl=Pin(5), freq=400000)  # Adjust pins if necessary
oled = SSD1306_I2C(128, 64, i2c)  # 128x64 OLED display

# Initialize onboard LED and ADC for moisture sensor
led_onboard = Pin(25, Pin.OUT)
adc = ADC(26)  # Moisture sensor connected to ADC pin 26
conversion_factor = 100 / 65535  # Convert ADC reading to percentage

while True:
    # Read and calculate moisture level
    moisture = 100 - (adc.read_u16() * conversion_factor)
    print("Moi:", round(moisture, 3), "%", utime.localtime())

    # Clear OLED display and show moisture level
    oled.fill(0)  # Clear the screen
    oled.text("Moi: {:.3f}%".format(moisture), 0, 0)  # Display moisture level
    oled.show()

    # Control LED based on moisture level
    if moisture >= 70:
        led_onboard.value(0)  # Turn off LED if moisture is sufficient
        utime.sleep(1)  # Delay for 30 seconds
    else:
        led_onboard.toggle()  # Toggle LED for low moisture alert
        utime.sleep_ms(0.05)  # Blink LED every 500 ms
```

Fig. 13a: Snapshots of moisture sensor code



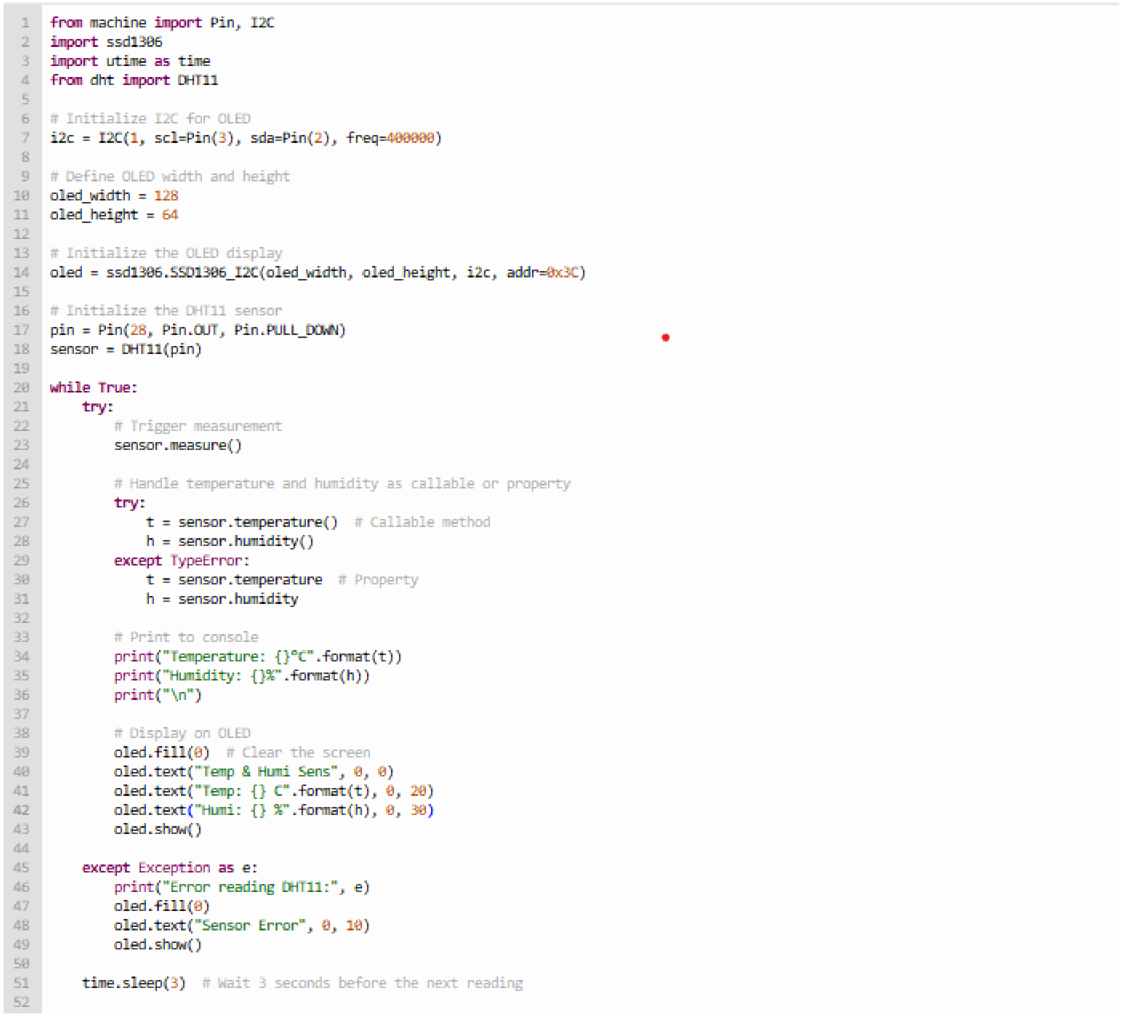
```python
from machine import UART, Pin
import time

# Initialize UART with the correct pins and baud rate
uart = UART(1, baudrate=9600, tx=Pin(4), rx=Pin(5))

# Full command example (use the correct command for your sensor)
command = b'\x01\x03\x00\x00\x00\x07\x04\x08'  # Replace with your sensor's command

print("Sending command:", command)
uart.write(command)

# Wait for a response
time.sleep(1)

# Check for data
if uart.any():
    response = uart.read()
    if response:
        print("Sensor connected and responded with data:", response)
    else:
        print("No response from sensor.")
else:
    print("Sensor not connected or no response received.")
```

Fig. 13b : Snapshots of OLED Display



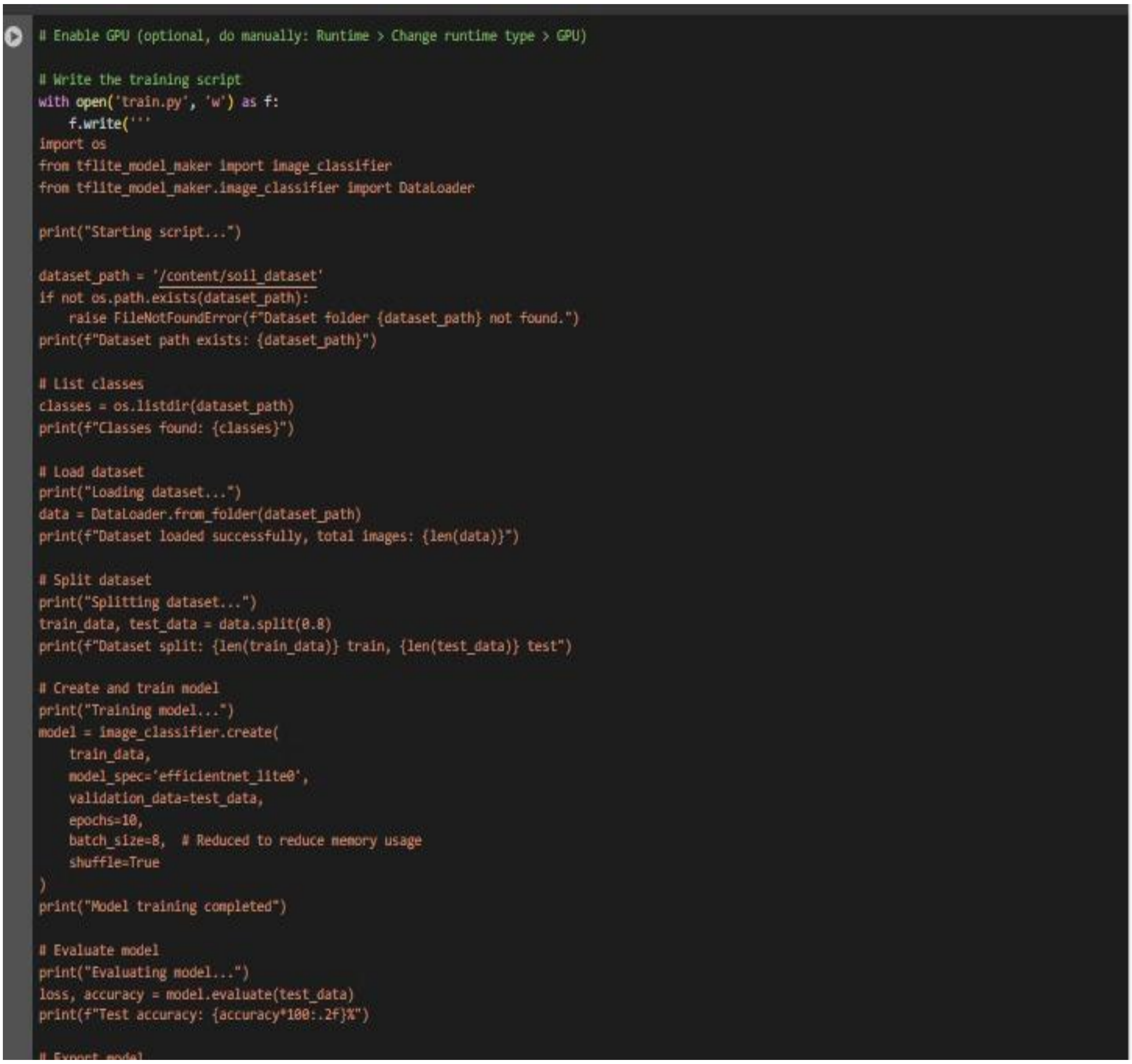
```python
from machine import Pin, UART, I2C
import ssd1306
import time
i2c = I2C(0, scl=Pin(1), sda=Pin(0), freq=400000)
oled_width, oled_height = 128, 64
oled = ssd1306.SSD1306_I2C(oled_width, oled_height, i2c, addr=0x3C)
# RS485 control pins
RE = Pin(8, Pin.OUT)
DE = Pin(9, Pin.OUT)
# UART for Modbus
uart = UART(1, baudrate=9600, tx=Pin(8), rx=Pin(9))
# Modbus commands
nitro = bytearray([0x01, 0x03, 0x00, 0x1e, 0x00, 0x01, 0xe4, 0x0c])
phos  = bytearray([0x01, 0x03, 0x00, 0x1f, 0x00, 0x01, 0xb5, 0xcc])
pota  = bytearray([0x01, 0x03, 0x00, 0x20, 0x00, 0x01, 0x85, 0xc0])

def send_modbus_query(query):
    DE.value(1)
    RE.value(1)
    time.sleep(0.01)
    uart.write(query)
    time.sleep(0.01)
    DE.value(0)
    RE.value(0)
    response = uart.read()
    return response if response else None

def get_value_from_response(response):
    if response and len(response) >= 7:
        return response[4]
    return None

def read_npk_values():
    n_resp = send_modbus_query(nitro)
    time.sleep(0.25)
    p_resp = send_modbus_query(phos)
    time.sleep(0.25)
    k_resp = send_modbus_query(pota)
    time.sleep(0.25)

    n = get_value_from_response(n_resp)
    p = get_value_from_response(p_resp)
    k = get_value_from_response(k_resp)

    return n, p, k

def display_npk_oled(n, p, k):
    oled.fill(0)
    oled.text("Soil NPK Values:", 0, 0)
    oled.text(f"N: {n if n is not None else 'Error'} mg/kg", 0, 15)
    oled.text(f"P: {p if p is not None else 'Error'} mg/kg", 0, 30)
    oled.text(f"K: {k if k is not None else 'Error'} mg/kg", 0, 45)
    oled.show()
```

Fig. 13c : Snapshots of NPK sensor code



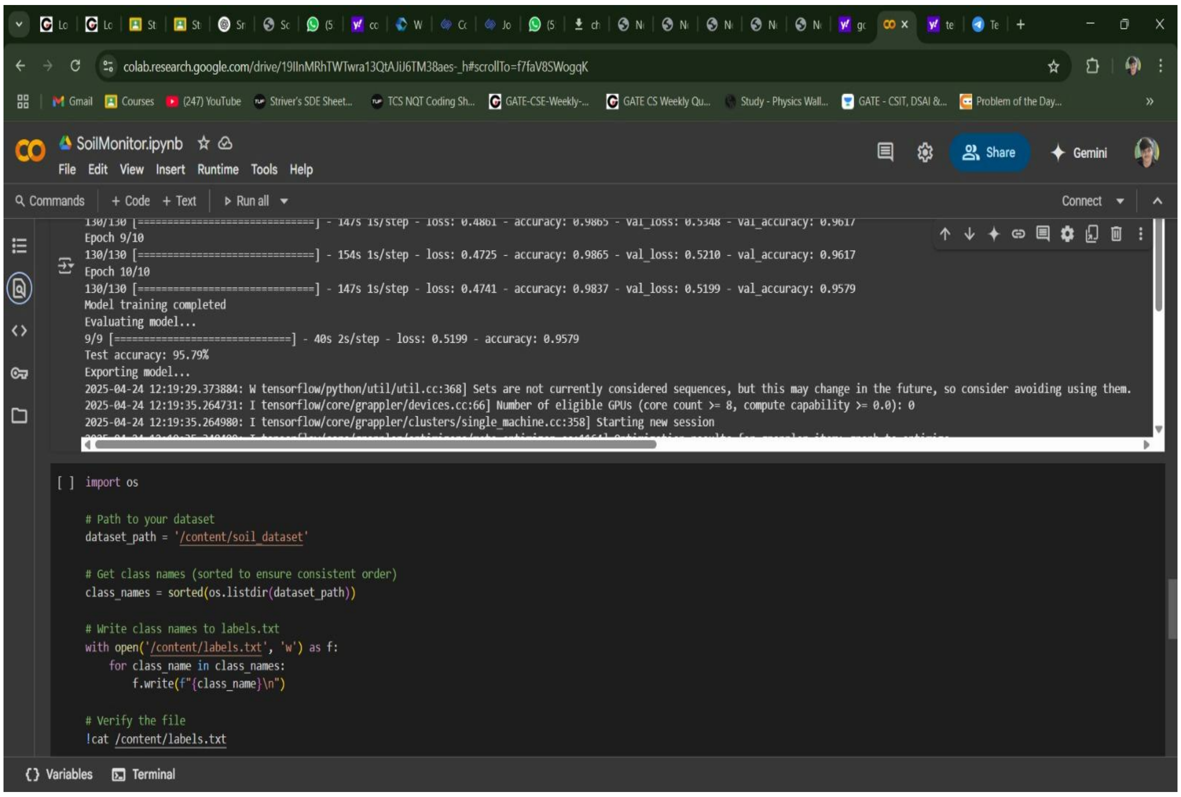
```python
from machine import Pin, I2C
import ssd1306
import utime as time
from dht import DHT11

# Initialize I2C for OLED
i2c = I2C(1, scl=Pin(3), sda=Pin(2), freq=400000)

# Define OLED width and height
oled_width = 128
oled_height = 64

# Initialize the OLED display
oled = ssd1306.SSD1306_I2C(oled_width, oled_height, i2c, addr=0x3C)

# Initialize the DHT11 sensor
pin = Pin(28, Pin.OUT, Pin.PULL_DOWN)
sensor = DHT11(pin)

while True:
    try:
        # Trigger measurement
        sensor.measure()

        # Handle temperature and humidity as callable or property
        try:
            t = sensor.temperature()  # Callable method
            h = sensor.humidity()
        except TypeError:
            t = sensor.temperature  # Property
            h = sensor.humidity

        # Print to console
        print("Temperature: {}°C".format(t))
        print("Humidity: {}%".format(h))
        print("\n")

        # Display on OLED
        oled.fill(0)  # Clear the screen
        oled.text("Temp & Humi Sens", 0, 0)
        oled.text("Temp: {} C".format(t), 0, 20)
        oled.text("Humi: {} %".format(h), 0, 30)
        oled.show()

    except Exception as e:
        print("Error reading DHT11:", e)
        oled.fill(0)
        oled.text("Sensor Error", 0, 10)
        oled.show()

    time.sleep(3)  # Wait 3 seconds before the next reading
```

Fig. 13d : Snapshots of Temp & Humidity sensor code



```
# Enable GPU (optional, do manually: Runtime > Change runtime type > GPU)

# Write the training script
with open('train.py', 'w') as f:
    f.write('''
import os
from tflite_model_maker import image_classifier
from tflite_model_maker.image_classifier import DataLoader

print("Starting script...")

dataset_path = '/content/soil_dataset'
if not os.path.exists(dataset_path):
    raise FileNotFoundError(f"Dataset folder {dataset_path} not found.")
print(f"Dataset path exists: {dataset_path}")

# List classes
classes = os.listdir(dataset_path)
print(f"Classes found: {classes}")

# Load dataset
print("Loading dataset...")
data = DataLoader.from_folder(dataset_path)
print(f"Dataset loaded successfully, total images: {len(data)}")

# Split dataset
print("Splitting dataset...")
train_data, test_data = data.split(0.8)
print(f"Dataset split: {len(train_data)} train, {len(test_data)} test")

# Create and train model
print("Training model...")
model = image_classifier.create(
    train_data,
    model_spec='efficientnet_lite0',
    validation_data=test_data,
    epochs=10,
    batch_size=8,  # Reduced to reduce memory usage
    shuffle=True
)
print("Model training completed")

# Evaluate model
print("Evaluating model...")
loss, accuracy = model.evaluate(test_data)
print(f"Test accuracy: {accuracy*100:.2f}%")

# Export model
```

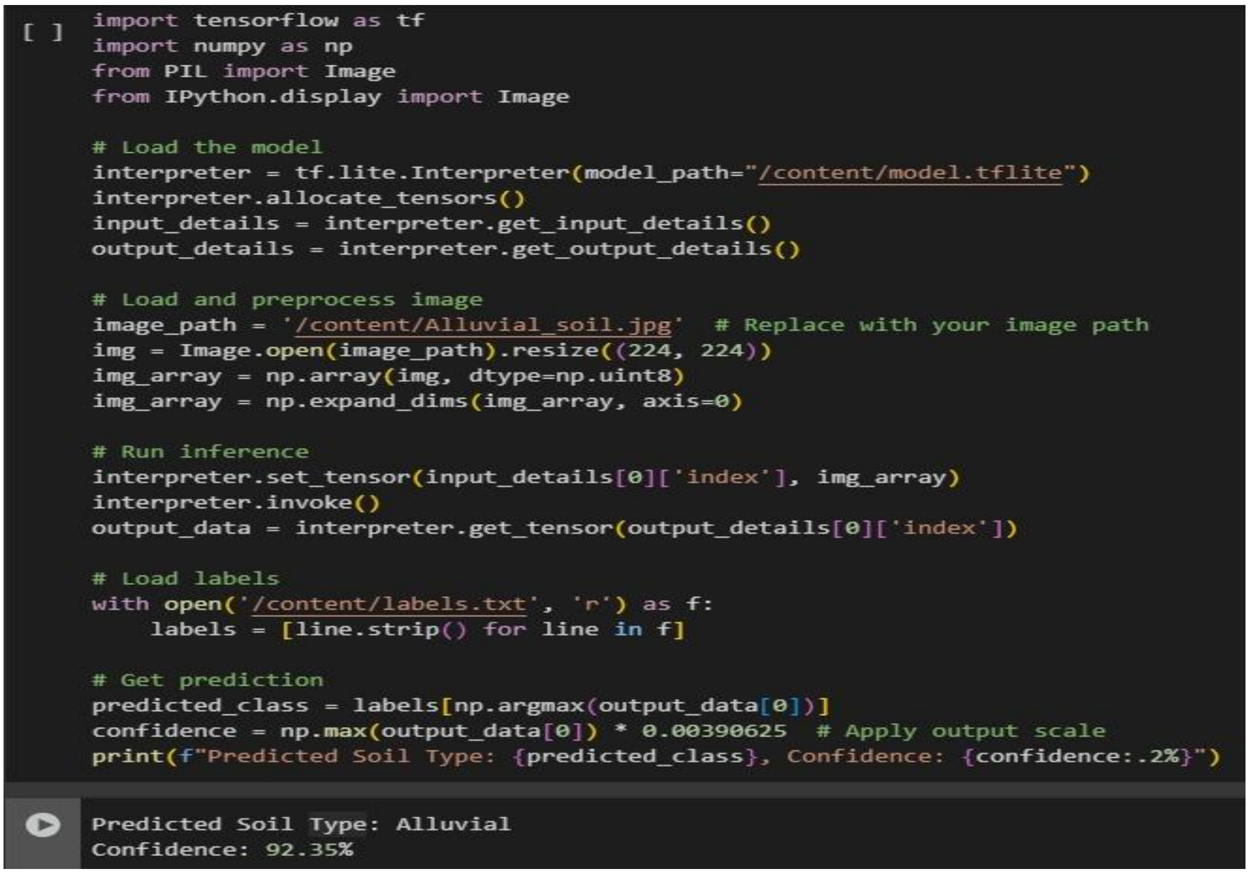
Fig 13e :Snapshot of Image analysis code


colab.research.google.com/drive/19IlnMRhTWTwra13QtAJiJ6TM38aes-_h#scrollTo=f7faV8SWogqK
SoilMonitor.ipynb
File Edit View Insert Runtime Tools Help
Commands + Code + Text Run all
Share Gemini Connect
Variables Terminal

```
130/130 [==============================] - 147s 1s/step - loss: 0.4861 - accuracy: 0.9865 - val_loss: 0.5348 - val_accuracy: 0.9617
Epoch 9/10
130/130 [==============================] - 154s 1s/step - loss: 0.4725 - accuracy: 0.9865 - val_loss: 0.5210 - val_accuracy: 0.9617
Epoch 10/10
130/130 [==============================] - 147s 1s/step - loss: 0.4741 - accuracy: 0.9837 - val_loss: 0.5199 - val_accuracy: 0.9579
Model training completed
Evaluating model...
9/9 [==============================] - 40s 2s/step - loss: 0.5199 - accuracy: 0.9579
Test accuracy: 95.79%
Exporting model...
2025-04-24 12:19:29.373884: W tensorflow/python/util/util.cc:368] Sets are not currently considered sequences, but this may change in the future, so consider avoiding using them.
2025-04-24 12:19:35.264731: I tensorflow/core/grappler/devices.cc:66] Number of eligible GPUs (core count >= 8, compute capability >= 0.0): 0
2025-04-24 12:19:35.264980: I tensorflow/core/grappler/clusters/single_machine.cc:358] Starting new session
```

```python
import os

# Path to your dataset
dataset_path = '/content/soil_dataset'

# Get class names (sorted to ensure consistent order)
class_names = sorted(os.listdir(dataset_path))

# Write class names to labels.txt
with open('/content/labels.txt', 'w') as f:
    for class_name in class_names:
        f.write(f"{class_name}\n")

# Verify the file
!cat /content/labels.txt
```

Fig 13 f: Snapshot of Test Accuracy

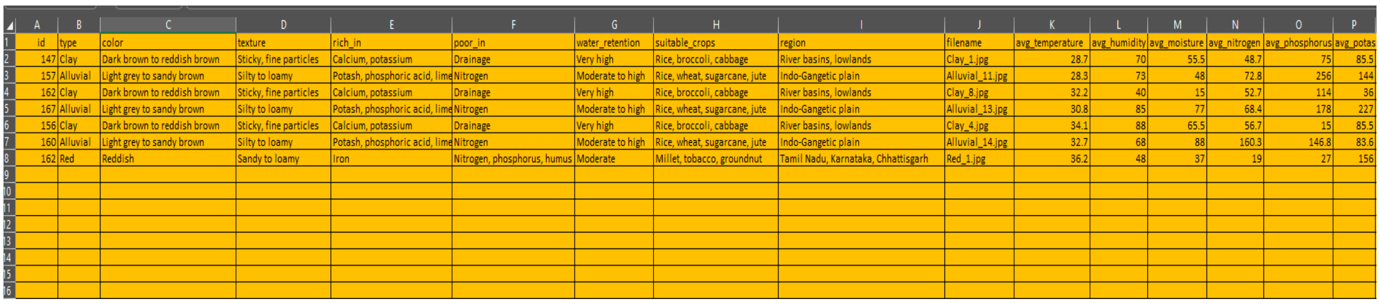


```python
import tensorflow as tf
import numpy as np
from PIL import Image
from IPython.display import Image

# Load the model
interpreter = tf.lite.Interpreter(model_path="/content/model.tflite")
interpreter.allocate_tensors()
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()

# Load and preprocess image
image_path = '/content/Alluvial_soil.jpg'  # Replace with your image path
img = Image.open(image_path).resize((224, 224))
img_array = np.array(img, dtype=np.uint8)
img_array = np.expand_dims(img_array, axis=0)

# Run inference
interpreter.set_tensor(input_details[0]['index'], img_array)
interpreter.invoke()
output_data = interpreter.get_tensor(output_details[0]['index'])

# Load labels
with open('/content/labels.txt', 'r') as f:
    labels = [line.strip() for line in f]

# Get prediction
predicted_class = labels[np.argmax(output_data[0])]
confidence = np.max(output_data[0]) * 0.00390625  # Apply output scale
print(f"Predicted Soil Type: {predicted_class}, Confidence: {confidence:.2%}")
```

```
Predicted Soil Type: Alluvial
Confidence: 92.35%
```

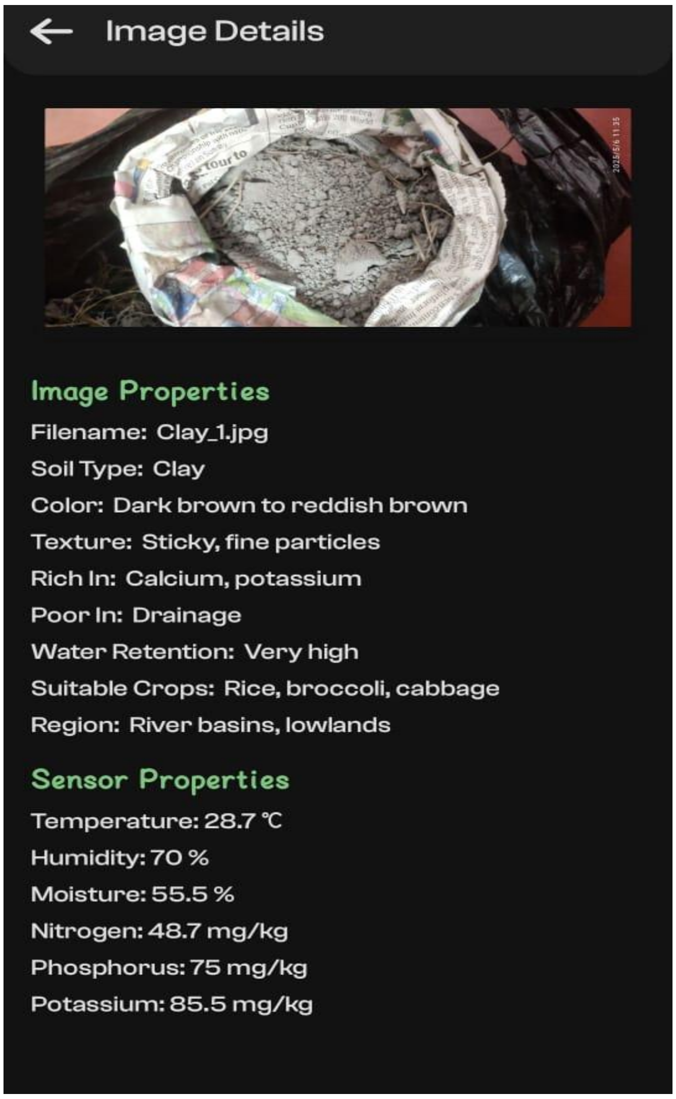
Fig 13g: Output for Image Analysis

RESULT



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	id	type	color	texture	rich_in	poor_in	water_retention	suitable_crops	region	filename	avg_temperature	avg_humidity	avg_moisture	avg_nitrogen	avg_phosphorus	avg_potas
2	147	Clay	Dark brown to reddish brown	Sticky, fine particles	Calcium, potassium	Drainage	Very high	Rice, broccoli, cabbage	River basins, lowlands	Clay_1.jpg	28.7	70	55.5	48.7	75	85.5
3	157	Alluvial	Light grey to sandy brown	Silty to loamy	Potash, phosphoric acid, lime	Nitrogen	Moderate to high	Rice, wheat, sugarcane, jute	Indo-Gangetic plain	Alluvial_11.jpg	28.3	73	48	72.8	256	144
4	162	Clay	Dark brown to reddish brown	Sticky, fine particles	Calcium, potassium	Drainage	Very high	Rice, broccoli, cabbage	River basins, lowlands	Clay_8.jpg	32.2	40	15	52.7	114	36
5	167	Alluvial	Light grey to sandy brown	Silty to loamy	Potash, phosphoric acid, lime	Nitrogen	Moderate to high	Rice, wheat, sugarcane, jute	Indo-Gangetic plain	Alluvial_13.jpg	30.8	85	77	68.4	178	227
6	156	Clay	Dark brown to reddish brown	Sticky, fine particles	Calcium, potassium	Drainage	Very high	Rice, broccoli, cabbage	River basins, lowlands	Clay_4.jpg	34.1	88	65.5	56.7	15	85.5
7	160	Alluvial	Light grey to sandy brown	Silty to loamy	Potash, phosphoric acid, lime	Nitrogen	Moderate to high	Rice, wheat, sugarcane, jute	Indo-Gangetic plain	Alluvial_14.jpg	32.7	68	88	160.3	146.8	83.6
8	162	Red	Reddish	Sandy to loamy	Iron	Nitrogen, phosphorus, humus	Moderate	Millet, tobacco, groundnut	Tamil Nadu, Karnataka, Chhattisgarh	Red_1.jpg	36.2	48	37	19	27	156
9																
10																
11																
12																
13																
14																
15																
16																

Fig. 14: Snippet of the database



← Image Details

Image Properties

Filename: Clay_1.jpg

Soil Type: Clay

Color: Dark brown to reddish brown

Texture: Sticky, fine particles

Rich In: Calcium, potassium

Poor In: Drainage

Water Retention: Very high

Suitable Crops: Rice, broccoli, cabbage

Region: River basins, lowlands

Sensor Properties

Temperature: 28.7 ℃

Humidity: 70 %

Moisture: 55.5 %

Nitrogen: 48.7 mg/kg

Phosphorus: 75 mg/kg

Potassium: 85.5 mg/kg

Fig. 15 : Sample result received in the app after result analysis

A box plot for soil features such as temperature, humidity, moisture, nitrogen, phosphorus and potassium is useful to visualize the distribution of these variables across the dataset. It shows important statistics like the median, quartiles and range, which help in understanding how the data is spread. Additionally, box plots help in identifying outliers, which are values that deviate significantly from the rest of the data. Detecting outliers is important because they might indicate unusual measurements or errors that could affect further analysis. By comparing the variability of different soil attributes through their box plots, one can gain insights into which features have more stability and which ones are more variable. This understanding is valuable when preparing the data for modeling, as it can inform decisions about data cleaning, transformations, or feature selection. Overall, box plots serve as an effective summary tool during exploratory data analysis, providing a clear and concise visual representation of the soil data characteristics.

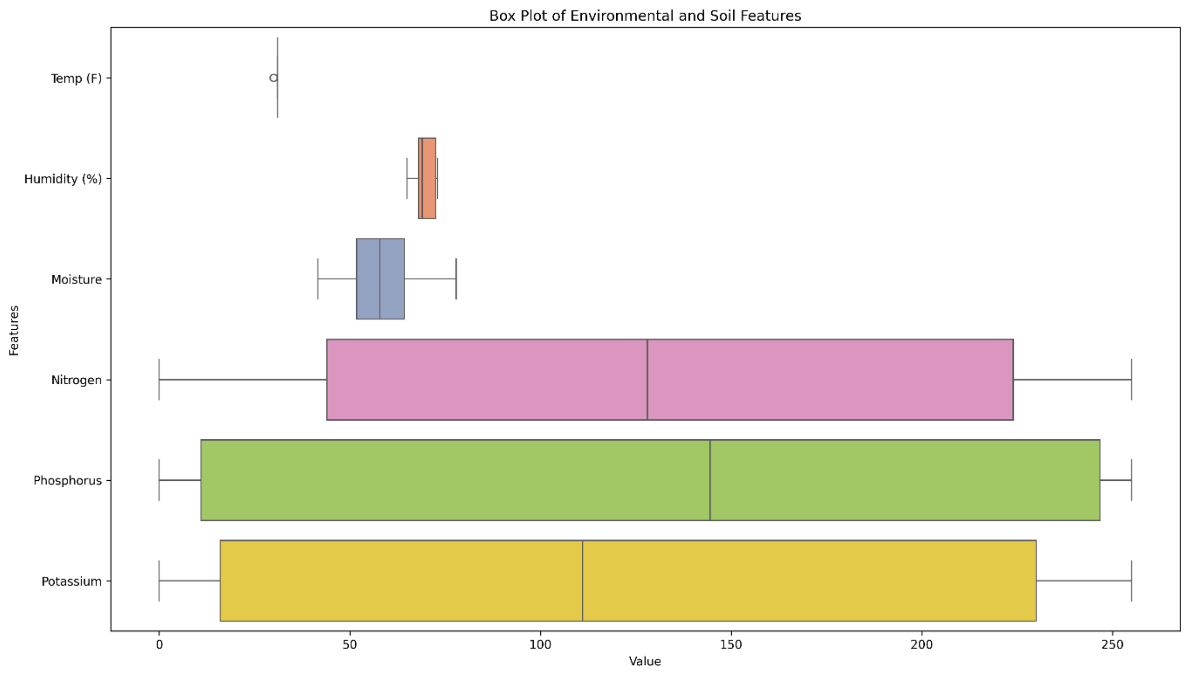


Fig. 16 : Box Plot of the soil features

The box plot here visually compares the distribution of various environmental and soil parameters such as temperature, humidity, moisture, nitrogen, phosphorus and potassium. Temperature and humidity show low variability, with temperature displaying a slight outlier, indicating that these environmental conditions are relatively stable—possibly reflecting a controlled or consistent setting like a greenhouse. Moisture exhibits moderate variability and a slightly right-skewed distribution, suggesting differences in irrigation or soil type across samples. In contrast, the soil nutrients—nitrogen, phosphorus and potassium—demonstrate a very wide range, spanning from 0 to approximately 255. This broad spread indicates significant variability in soil fertility and could imply the need for localized soil management practices. Additionally, the 0–255 range for these nutrients suggests that the data might be collected using 8-bit sensors, which could introduce quantization limits. Overall, the plot highlights stable environmental conditions but highly variable soil nutrient levels, which could be crucial for making precise agricultural decisions.

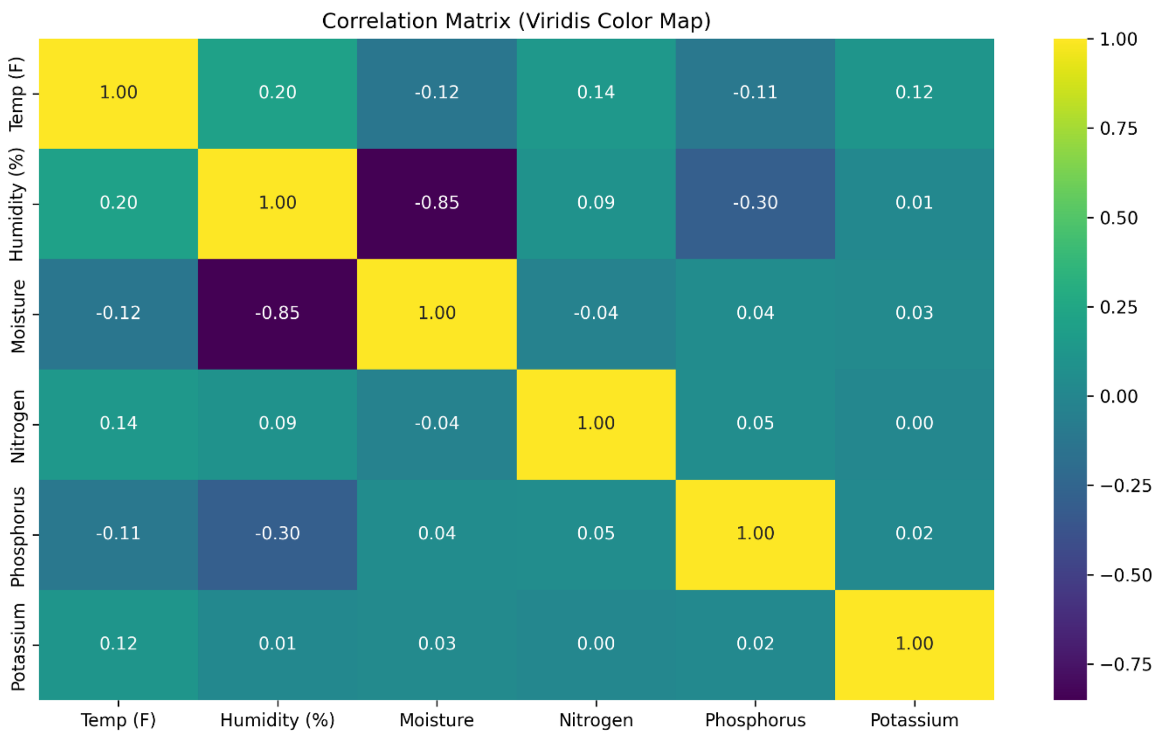


Fig. 17 : Correlation plot with the soil features

The correlation matrix using the Viridis color map provides insights into the linear relationships among various environmental and soil features such as temperature, humidity, moisture, nitrogen, phosphorus and potassium. One of the most significant observations is the strong negative correlation between humidity and moisture (−0.85), suggesting that as humidity increases, soil moisture tends to decrease—an inverse relationship that may be influenced by factors such as evapotranspiration or irrigation cycles. Temperature shows weak positive correlations with all other features, indicating minimal linear association. Interestingly, the macronutrients—nitrogen, phosphorus and potassium—show negligible correlations with one another and with environmental variables (mostly ranging between −0.04 and 0.14), suggesting that their concentrations in the soil are largely independent and possibly influenced by localized fertilization or soil heterogeneity. Overall, the matrix highlights one strong environmental interaction (humidity vs. moisture) while confirming that soil nutrient distributions are relatively uncorrelated, reinforcing the need for targeted soil treatment in agricultural practices.

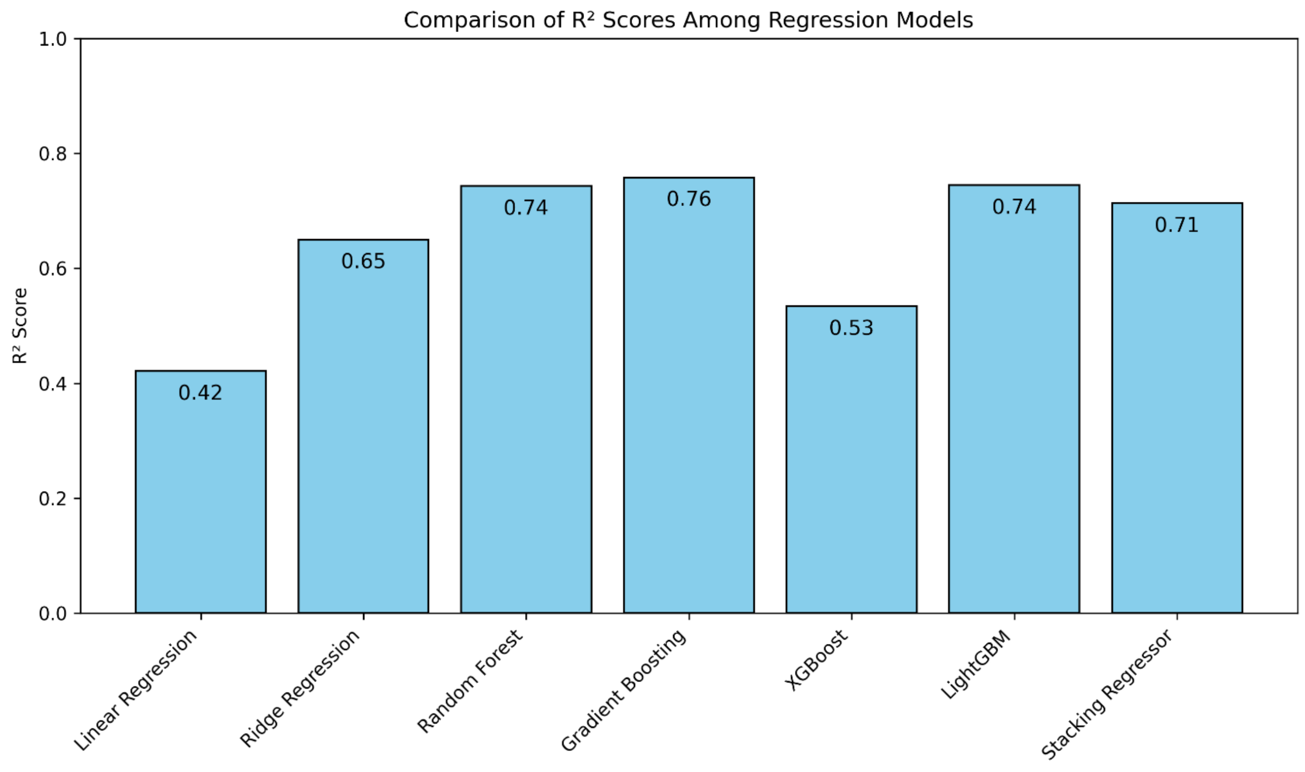


Fig. 18 : Comparison of correlation values among different regression models

The bar chart titled "Comparison of R^2 Scores Among Regression Models" presents the predictive performance of several regression algorithms based on their R^2 scores. Gradient Boosting achieves the highest score of 0.76, indicating it explains the largest proportion of variance in the target variable among the models tested. Random Forest and LightGBM both follow closely with scores of 0.74, showing similarly strong performance, particularly in capturing complex, non-linear relationships. The Stacking Regressor, which combines multiple models, also performs well with a score of 0.71. Ridge Regression demonstrates moderate predictive power with a score of 0.65. In contrast, XGBoost, despite being a powerful boosting algorithm, only achieves an R^2 score of 0.53, suggesting it may not be optimally tuned or as well-suited to this particular dataset. Linear Regression performs the worst with a score of 0.42, highlighting the limitations of using simple linear assumptions for modeling. These results collectively indicate that ensemble and gradient boosting models are generally more effective for regression tasks involving complex or high-dimensional data.

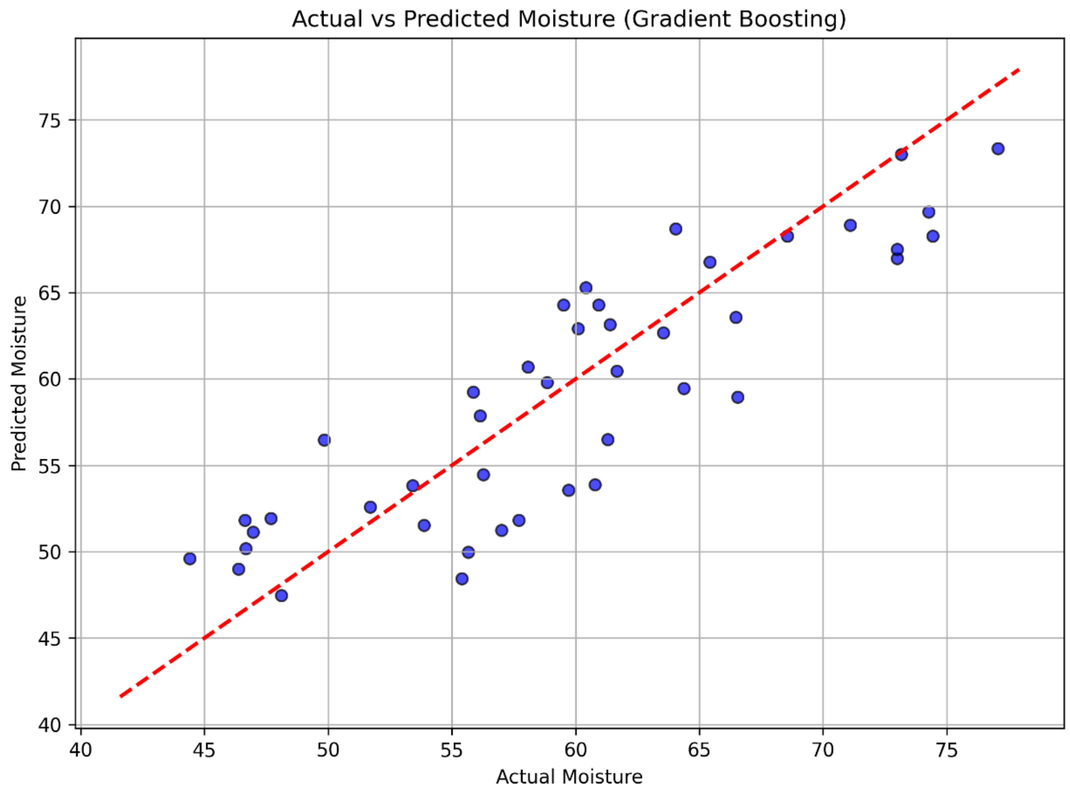


Fig. 19 : Actual vs Predicted Moisture of soil using Gradient Boosting regression model

The scatter plot titled "Actual vs Predicted Moisture (Gradient Boosting)" illustrates the performance of the Gradient Boosting regression model in predicting soil moisture levels. Each point on the plot represents a prediction made by the model, where the x-axis shows the actual moisture values and the y-axis shows the corresponding predicted values. The red dashed line represents the ideal scenario where predicted values perfectly match the actual values. Most points are closely clustered around this diagonal line, indicating that the model's predictions are generally accurate and consistent with the actual measurements. However, there are a few deviations, particularly in the mid-range and higher moisture values, where the predictions slightly under or overestimate the actual values. Overall, the plot supports the conclusion that the Gradient Boosting model performs reliably in predicting soil moisture, aligning with its high R^2 score observed in the earlier comparison of regression models.

The Principal Component Analysis (PCA) scatter plot illustrates the clustering of soil features using KMeans after dimensionality reduction through Principal Component Analysis. Each data point is plotted using two principal components, which capture the most significant variance in the original multivariate dataset. The use of distinct markers and colors differentiates three identified clusters: Cluster 0 (circles), Cluster 1 (squares)and Cluster 2 (diamonds). Most points are concentrated around the central region, indicating some overlap, although the clusters are largely well separated along the principal components. Cluster 1 shows a more distinct spread in the upper region, while Cluster 0 and Cluster 2 share more proximity, though they maintain distinguishable patterns. A few outliers appear on the far left of the plot, possibly indicating unique samples or noise. This visualization confirms that the PCA transformation, combined with KMeans clustering, effectively separates the data into meaningful groups, helping reveal underlying structure within the soil features.

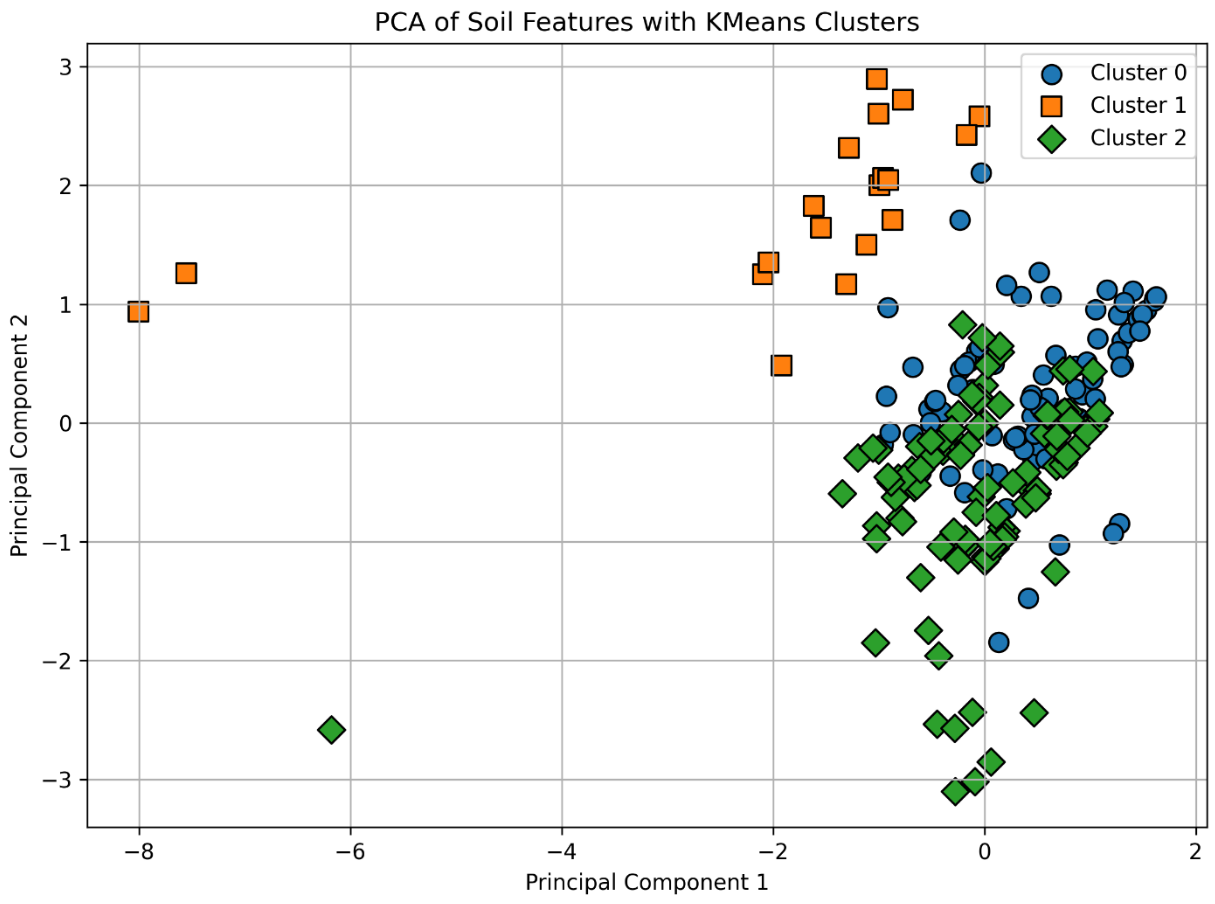


Fig. 20 : PCA Plot with soil features

APPLICATIONS

The **Applications** of the Smart Soil Health Measuring Project are wide-ranging and impactful across several sectors, from agriculture to environmental management. Here are some key areas where this technology can make a difference:

1. **Precision Agriculture**: By providing real-time data on soil moisture, temperature, pH and nutrient levels, this project enables farmers to optimize irrigation and fertilization practices. The result is reduced waste, lower input costs and improved crop yields, allowing farmers to use resources precisely where they are needed.
2. **Sustainable Land Management**: The sensor network provides insights into soil health over time, allowing land managers to implement sustainable practices that maintain soil quality. This helps prevent soil degradation, reduce erosion and improve biodiversity in farmlands, supporting long-term land productivity.
3. **Drought and Climate Adaptation**: In regions facing water scarcity or irregular rainfall, the system's soil moisture monitoring capability can support water.
   conservation efforts. By guiding more efficient water use, it helps farmers adapt to changing climate conditions and reduces the risk of crop failures due to drought.
4. **Research and Development**: For agricultural researchers, the data collected by the project can be used to study soil behavior under various environmental conditions, contributing valuable information for developing more resilient crops. This data can also support breeding programs aimed at enhancing crop adaptability to soil and climate variability.
5. **Community and Cooperative Farming**: Small-scale and community farmers can benefit by pooling resources to implement this low-cost, open-source technology. The project can aid in cooperative efforts, making it easier for smallholders to monitor and improve soil health collectively, which could lead to better productivity and resource-sharing in rural areas.
6. **Environmental Monitoring and Restoration**: Beyond agriculture, the technology can be used in environmental restoration projects to monitor soil health in reforestation areas, wetlands and grasslands. This supports the success of ecological restoration efforts by tracking soil recovery and ensuring suitable conditions for vegetation regrowth.

The Smart Soil Health Measuring Project has the potential to positively impact agriculture, conservation and sustainable land management practices. By providing easy access to critical soil data, it empowers a wide range of stakeholders to make informed decisions that can lead to healthier soils, more resilient crops and a more sustainable environment.

CONCLUSIONS

Smart Soil Health Measuring Project represents a transformative step toward sustainable and data-driven agriculture. By leveraging a network of affordable, IoT-enabled sensors to monitor key soil parameters, the project empowers farmers, researchers and environmentalists with real-time insights into soil health. This approach not only optimizes resource usage—such as water and fertilizers—but also supports the development of more resilient and productive farming practices. With promising prospects, including the integration of advanced sensors, machine learning and satellite technology, the project holds the potential to significantly impact global food security and environmental sustainability. As agriculture faces the pressing challenges of climate change and population growth, innovative solutions like this will be essential in ensuring the health and productivity of our soils for future generations.

Looking ahead, the project has a wide scope for growth, including advanced sensors, predictive analytics and satellite integration, which could further transform agriculture on a global scale. In an era where climate change and population growth demand smarter solutions, this project provides a sustainable pathway to healthier soils and improved food security for future generations.

FUTURE SCOPE

The **Future Scope** of the Smart Soil Health Measuring Project is promising, with significant potential to expand and evolve as agricultural technology advances. By continuously refining sensor accuracy, data analytics and IoT integration, the project could become an indispensable tool for precision agriculture and sustainable land management. Here are several directions in which this project could grow:

1. **Enhanced Sensor Capabilities**: Future iterations could integrate more advanced sensors to detect additional parameters like soil organic carbon, heavy metal presence and soil compaction levels. This would provide a more detailed understanding of soil health, supporting even more precise and efficient land management.
2. **Machine Learning Integration**: The incorporation of more machine learning algorithms could enable predictive modelling based on historical and real-time data. By recognizing patterns, the system could forecast soil health trends, predict droughts, or warn of nutrient deficiencies, enabling proactive decision- making to enhance crop resilience and yield.
3. **Satellite and Drone Integration**: Integrating satellite imagery and drone technology with on-the-ground sensors would allow for remote soil monitoring on a broader scale. This combination could provide a real-time view of soil health across large areas, benefiting large-scale agricultural operations and enabling better resource allocation.
4. **Automated Irrigation Systems**: By connecting with automated irrigation systems, the soil health monitoring network could autonomously adjust water delivery based on current soil moisture levels, creating a fully integrated system that optimizes water use and improves crop hydration.
5. **Community-Based and Open-Source Networks**: Expanding open-source platforms and creating community-based sensor networks could facilitate collaborative efforts among farmers, researchersand environmentalists worldwide. This could drive shared innovation and broader application, particularly in resource-limited regions.

These future developments could revolutionize the agricultural sector by promoting greater food security, minimizing resource wasteand advancing sustainable farming practices on a global scale.

REFERENCES

[1] Hans Müller, Friedrich Weber, “Sensor Integration Strategy for a Smart Soil Monitoring Robot AgriBot-Sense”, IEEE Transactions on Soil Mechanics, May,2014

[2] Ivan Petrov, Nikolai Sokolov, Alexei Ivanov, Dmitri Antonov, Sergei Pavlov, Viktor Volkovand Yuri Smirnov, “Development of Autonomous Soil Analysis Robot SoilScan-1 – Simplification of Control Using Adaptive Probe Mechanism”, Proc. Global Forum on Smart Robotics (GFSRM)., pp. 10-10-2016.

[3] https://www.raspberrypi.com/documentation/microcontrollers/pico-series

[4] https://www.elprocus.com/major-electronic-components

[1] Hans Müller, Friedrich Weber, “Sensor Integration Strategy for a Smart Soil Monitoring Robot AgriBot-Sense”, IEEE Transactions on Soil Mechanics, May,2014

[2] Ivan Petrov, Nikolai Sokolov, Alexei Ivanov, Dmitri Antonov, Sergei Pavlov, Viktor Volkovand Yuri Smirnov, “Development of Autonomous Soil Analysis Robot SoilScan-1 – Simplification of Control Using Adaptive Probe Mechanism”, Proc. Global Forum on Smart Robotics (GFSRM)., pp. 10-10-2016.

[3] https://www.raspberrypi.com/documentation/microcontrollers/pico-series

[4] https://www.elprocus.com/major-electronic-components

[5] Manikandan, S., & Ramesh, S. (2019). Soil nutrients monitoring and analyzing system using Internet of Things. International Journal of Engineering and Advanced Technology (IJEAT), 8(6), 1418–1421. https://doi.org/10.35940/ijeat.F1302.088619

[6] Sivakumar, V. G., Baskar, V. V., Vadivel, M., Vimal, S. P., & Murugan, S. (2023). IoT based site-specific nutrient management system for soil health monitoring. Proceedings of the 2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS), 1265–1270. https://doi.org/10.1109/ICSSAS56758.2023.10125443

[7] Sivakumar, V. G., Baskar, V. V., Vadivel, M., Vimal, S. P., & Murugan, S. (2023). IoT and GIS integration for real-time monitoring of soil health and nutrient status. Proceedings of the 2023 International Conference on Self Sustainable Artificial Intelligence Systems (ICSSAS), 1265–1270. https://doi.org/10.1109/ICSSAS56758.2023.10331694

[8] Postolache, O., Pereira, J. M. D., Viegas, V., & Girao, P. M. (2023). IoT-based systems for soil nutrients assessment in horticulture. Sensors, 23(1), 403. https://doi.org/10.3390/s23010403

[9] Khedr, E. H., & Al-Khayri, J. M. (2024). Enhancing soil nutritional status in smart farming: The role of IoT-based management for meeting plant requirements. Journal of the Science of Food and Agriculture, 104(2), 887–4325. https://doi.org/10.1002/jsfa.12345

[10] https://www.kaggle.com/datasets/rahuljaiswalonkaggle/soil-fertility-dataset

[11] Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), 1–58. https://doi.org/10.1145/1541880.1541882

Aggarwal, C. C. (2017). Outlier analysis (2nd ed.). Springer. https://doi.org/10.1007/978-3-319-47578-3

[12] Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008). Isolation forest. Proceedings of the 8th IEEE International Conference on Data Mining (ICDM'08), 413–422. https://doi.org/10.1145/1401890.1401945

[13] Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: A review and recent developments. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 374(2065), 20150202. https://doi.org/10.1098/rsta.2015.0202

-------- --------