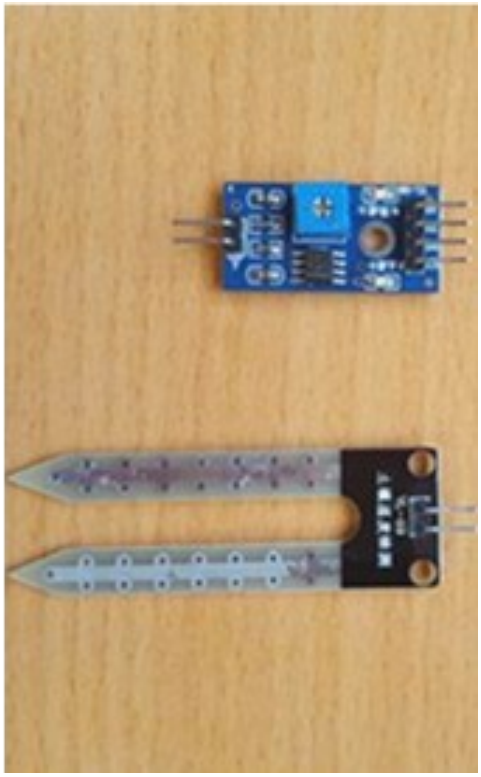
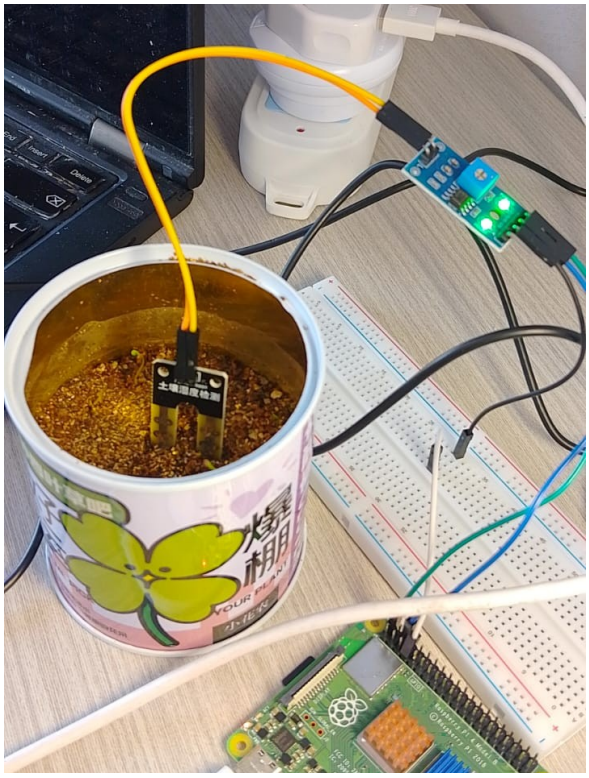
# Project #2

## Agile Raspberry Pi Plant Moisture Sensor with Email Notification





In this Project, you will use a moisture sensor and a Raspberry Pi to send you an email notification when your plant needs watering. Very useful if you're forgetful of your houseplants.

This Project is in Three  parts. Part 1 will focus on planning using Agile methods and methodology. The second phase will focus on the Raspberry Pi setup and implementation.. The third phase will deal with Email delivery and management.

### Introducing the Soil Moisture Sensor

The sensor board itself has both analogue and digital outputs. The Analogue output gives a variable voltage reading that allows you to estimate the moisture content of the soil. The digital output gives you a simple "on" or "off" when the soil moisture content is above a certain value. The value can be set or calibrated using an adjustable on board potentiometer. In this case, we just want to know either "Yes, the plant has enough water" or "No, the plant needs watering!", so we'll be using the digital out. We'll set this value later at which you want the sensor to "trigger" a notification.

**Soil Moisture Sensor Applications**

The Soil Moisture Sensor has many applications in agriculture, including

- Monitoring the moisture content of the soil to optimize irrigation schedules
- Monitoring drought conditions
- Predicting crop yields
- Monitoring the moisture content of the soil to optimize water resources
- Detecting leaks in irrigation systems

# Part 1 – The Agile Process

For a project focusing on Agile, consider the "Soil Sensor & Email Management" for the Raspberry Pi, using the Scrum framework to manage development. Your approach should produce the following documentation

**1. Summarize Project Idea:**

- **Project Description:**
- Raspberry Pi – Physical Programming – Soil Sensor. A project to design, build, and test a Raspberry Pi device with connected soil moisture sensor components.
- Apply agile principles to the project
- Complete a coherent and accurate video presentation of the project
- Submit a final Report

**2. Document Agile Framework Workflow (Scrum/Sprint/Burndown):**

- Plan the project using a single Scrum schedule, divided into **ONE** two-week duration sprint. Including all the tasks you are required to complete.
- **User Stories**: Define project requirements for each task as user stories (e.g., "As a user, I want to be able to add a task so that I can remember it").
- Complete a **Burndown Table & Chart** to document and manage your Sprint from the beginning to the final actual implementation and submission. Show clearly ACTUAL versus ESTIMATED project progress (in units of Hours)

**3. Tools:**

- Project Management Software: Use **Excel** to manage Scrums, Sprints and Burndown Tables/Charts.
- Version Control: Use **Git** and **GitHub** for code management and collaboration.

**SUBMISSION NOTE**

- **In week 1** please submit your **User Stories, Burndown Table & Chart** for the Estimated Tasks and Durations in the file **Agile_Week1.docx**. Include a cover sheet with your personal details. See link on Moodle

- At the end of **week 2**, submit the final User Stories, Burndown Table & Chart (including Estimated **AND** Actual data) documentation for the completed project. Put all details in the final report document. See submission details at the end of this document

# Part 2 – Setting Up the Raspberry Pi and Sensors

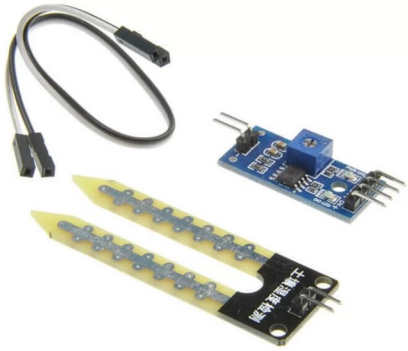
## *Introduction*

For this Project, you will work individually. You will each create your own Raspberry Pi / Components device. Create a new Remote Repository **Project2_2025**.

### What you will need:

- Raspberry Pi
- Soil Moisture Sensor
- A plant to use and experiment with the probes.

### Soil Moisture Sensor



A soil moisture sensor is a device used to measure the moisture in the soil. The sensor works by sending out an electromagnetic signal transmitted through the soil. The signal is then received by the sensor and the moisture content is measured.

There are many benefits to using a soil moisture sensor. One benefit is that it can help to save water. The sensor can be used to monitor the moisture content in the soil and then the irrigation system can be adjusted accordingly. This can help to reduce the amount of water that is used and can also help to reduce the amount of water that is wasted. Another benefit of using a soil moisture sensor is that it can help to improve the health of the plants.

### Soil Moisture Sensor PinOut

The soil Moisture sensor FC-28 has four pins

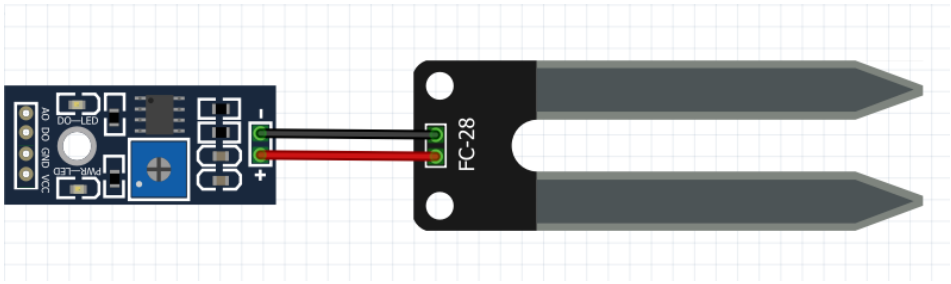
- VCC: For power
- A0: Analog output
- D0: Digital output
- GND: Ground

### GPIO Python3 (update)

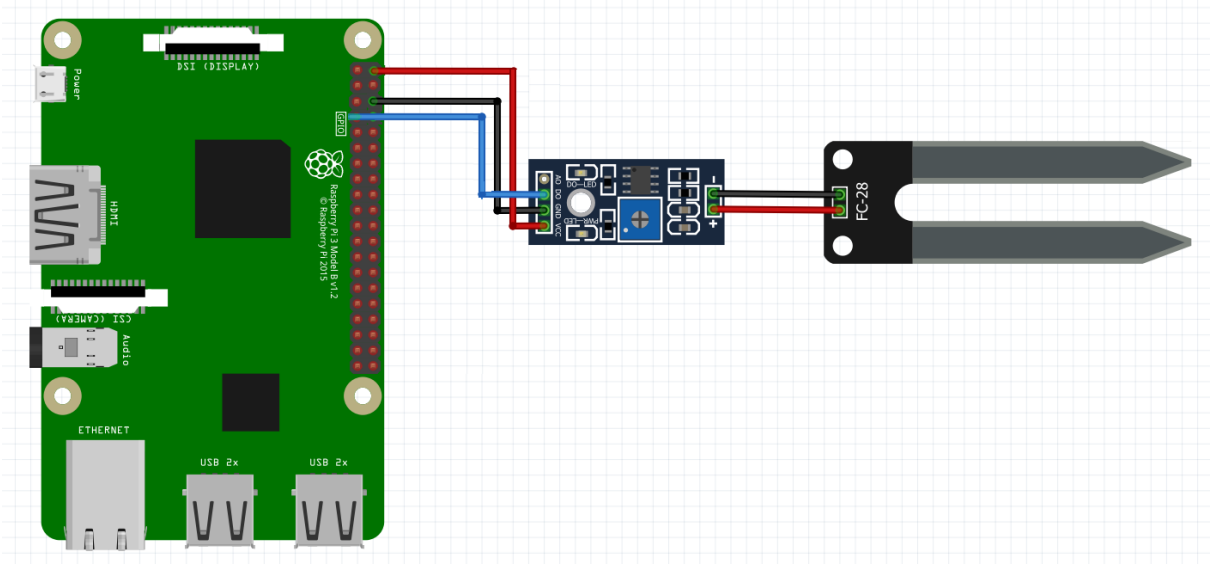
- sudo apt remove python3-rpi.gpio
- sudo apt update
- sudo apt install python3-rpi-lgpio

## 1. Connecting the Probe

- Let’s start by wiring up the probes to the sensor. Simply connect the two pins on the probe to the side of the sensor that only has 2 pins. It doesn’t matter which way round the wires go.




- **Wire the sensor to the Raspberry Pi.**
  - **VCC --> 5V**
  - **GND --> GND**
  - **D0 --> GPIO 04**



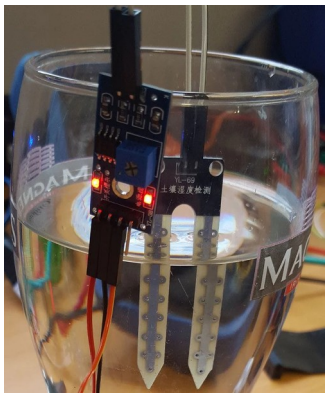

- **Turn on the Raspberry Pi**

  With everything now wired up, we can turn on the Raspberry Pi. Without writing any code we can test to see our moisture sensor working. When power is applied you should see the power light illuminate (with the 4 pins facing down, the power led is the one on the right

- **Perform quick Moisture Test**

  When the sensor detects moisture, a second led will illuminate (with the 4 pins facing down, the moisture detected led is on the left).

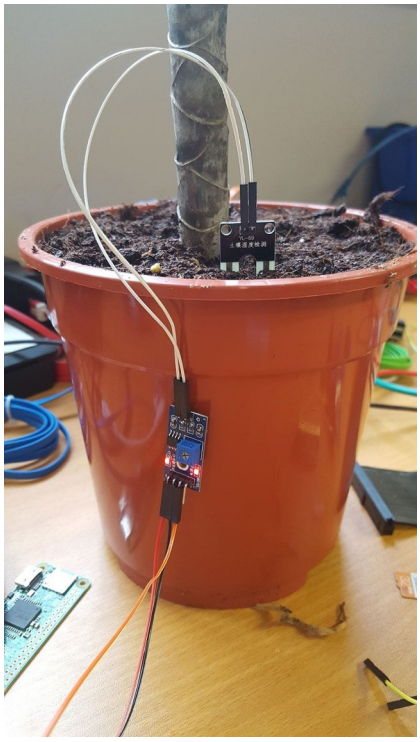
So as a quick test, get a glass of water (be very careful not to spill water!!) then place the probes into the water and see the detection led turn On



If the detection light doesn't illuminate, you can adjust the potentiometer on the sensor which allows you to change the detection threshold (this only applies to the digital output signal)

Now we can see our sensor working, it's time to calibrate it for your specific use.

In this project, we want to monitor the moisture levels of our plant pot. So we want to set the detection point at a level so that if it drops below we get notified that our plant pot is too dry and needs watering. Our plant (see example below), is a little on the dry side, but ok for now, if it gets any drier it'll need watering.



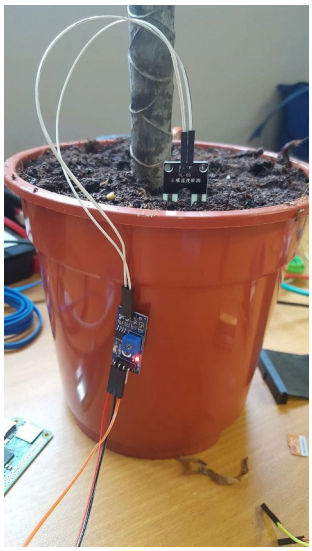
## 2. Create a test Python Script (SoilSensor.py)

- Create a new Python Script and enter the following code.
- **Run, Test, Commit and PUSH**

```
import RPi.GPIO as GPIO
import time

#GPIO SETUP
channel = 4
GPIO.setmode(GPIO.BCM)
GPIO.setup(channel, GPIO.IN)

def callback(channel):
        if GPIO.input(channel):
                print ("Water Detected!")
        else:
                print ("Water Detected!")

GPIO.add_event_detect(channel, GPIO.BOTH, bouncetime=300)  # let us know when the pin goes HIGH or LOW
GPIO.add_event_callback(channel, callback)  # assign function to GPIO PIN, Run function on change

# infinite loop
while True:
        time.sleep(0)
```

## 3. Adjust the potentiometer

Adjust the potentiometer to a point where the detection LED just illuminates. As soon as you reach the detection point, stop turning the potentiometer. That way, when then moisture levels reduce just a small amount the detection LED will go out.



# Part 3 - Sending an Email from Raspberry Pi

## 4. Test the Email / Raspberry Pi features

**1. Raspberry Pi: Send an Email using Python (SMTP Server)**

In this guide, you'll learn how to send an email from your Raspberry Pi using a Python Script and SMTP servers. The example we'll show can also be run on any other machine that runs Python.

**2. Introducing SMTP Servers**

SMTP means Simple Mail Transfer Protocol and it is an internet standard for email transmission. You can easily send emails using a Python Script on your Raspberry Pi using the smtplib library. This library defines an SMTP client session object that can be used to send mails. Learn more here.

**3. SMTP Server Settings**

To send emails using a Python script on your Raspberry Pi, you need a sender email and you'll need to know your email SMTP server settings. Below you'll find the settings for the some popular email providers.

**4. Outlook SMTP Server Settings**

- For **Outlook** accounts, these are the SMTP Server settings:
    - SMTP Server: smtp.office365.com
    - SMTP Username: Complete Outlook email address
    - SMTP Password: Your Outlook password
    - SMTP Port: 587
    - SMTP TLS/SSL Required: Yes

- To configure **QQ Mail**'s SMTP settings for accessing it with an email client, you'll need the following:
    - SMTP Server: smtp.qq.com
    - Username: Your full QQ Mail address (e.g., your-email@qq.com)
    - Password: Your QQ Mail password
    - SMTP Port: 465 or 587
    - SMTP TLS/SSL Required: Yes

**NOTE**: If you're using another email provider, you need to search for its SMTP Server settings—you'll easily find them with a quick internet search.

## 5. Creating an App Password

### Sender Email (New Account)

It is recommend to create a new email account to send the emails to your main personal email address. Do not use your main personal email to send emails via a Python script. If something goes wrong in your code or if by mistake you make too many requests, you can be banned or have your account temporarily disabled.

### Create a Sender Email Account

Create a new email account for sending emails with

### Create an App Password

You need to create an app password so that new devices can send emails using your Email account. An App Password is a 16-digit passcode that gives a less secure app or device permission to access your Account.

An app password can only be used with accounts that have 2-step verification turned on.

### Open your Email Account.

In the navigation panel, select Security.
Under “Signing in” select 2-Step Verification > Get started.
Follow the on-screen steps.
After enabling 2-step verification, you can create an app password.

1. Open your Email Account.
2. In the navigation panel, select Security.
3. Under “Signing in” select App Passwords
4. In the Select app field, choose mail. For the device, select Other and give it a name, for example Raspberry Pi. Then, click on Generate. It will pop-up a window with a password that you’ll use on the Python script to send emails. Save that password (even though it says you won’t need to remember it) because you’ll need it later.

Now, you should have an app password that you can use on your Python script to send emails.

If you’re using another email provider, check how to create an app password. You should be able to find the instructions with a quick Internet search “your_email_provider + create app password”.

# The Email-Sending Script

Create a new Python file called **send_email.py** and enter the following code. You need to insert your sender email details and the recipient email.

```
import smtplib
from email.message import EmailMessage

#Set the sender email and password and recipient emaic
from_email_addr ="REPLACE_WITH_THE_SENDER_EMAIL"
from_email_pass ="REPLACE_WITH_THE_SENDER_EMAIL_APP_PASSWORD"
to_email_addr ="REPLACE_WITH_THE_RECIPIENT_EMAIL"

# Create a message object
msg = EmailMessage()

# Set the email body
body ="Hello from Raspberry Pi"
msg.set_content(body)

# Set sender and recipient
msg['From'] = from_email_addr
msg['To'] = to_email_addr

# Set your email subject
msg['Subject'] = 'TEST EMAIL'

# Connecting to server and sending email
# Edit the following line with your provider's SMTP server details
server = smtplib.SMTP('smtp.gmail.com', 587)

# Comment out the next line if your email provider doesn't use TLS
server.starttls()
# Login to the SMTP server
server.login(from_email_addr, from_email_pass)

# Send the message
server.send_message(msg)

print('Email sent')

#Disconnect from the Server
server.quit()
```

- **Run, Test, Commit and PUSH**

## How the Code Works

Continue reading to learn how the code works and what changes you need to make to the script to make it work for you. You start by importing the libraries you need for SMTP and email-related functions: `smtplib` and the `EmailMessage` class from the `email.message` module.

```
import smtplib

from email.message import EmailMessage
```

Next, you create variables for the email address to send from, that email's app password, and an email address to send to. We suggest you create a second email to send the notifications to your everyday email because you will be giving less secure apps access to the account you send from.

```
#Set the sender email and password and recipient emaiç

from_email_addr ="REPLACE_WITH_THE_SENDER_EMAIL"

from_email_pass ="REPLACE_WITH_THE_SENDER_EMAIL_APP_PASSWORD"

to_email_addr ="REPLACE_WITH_THE_RECIPIENT_EMAIL"
```

Create an `EmailMessage()` object called `msg` that will handle the email message properties.

```
# Create a message object

msg = EmailMessage()
```

Set the email body on the following lines. You can change it to whatever text you want.

```
# Set the email body

body ="Hello from Raspberry Pi"

msg.set_content(body)
```

Then, we set the sender and recipient in the email message properties.

```
msg['From'] = from_email_addr

msg['To'] = to_email_addr
```

The following line sets the email subject, you can change it to whatever you want.

```
msg['Subject'] = 'TEST EMAIL'
```

Then, you establish communication with an SMTP server. Pass the provider's SMTP server address as a string as the first argument to `smtplib.SMTP()`, and the port as an int as the second argument.

```
server = smtplib.SMTP('smtp.gmail.com', 587)
```

In this script, we're using a Gmail SMTP server and port. If you use another email provider, make sure to change those values.

The `server.starttls()` function is necessary for email providers that use TLS to encrypt messages (which are practically all email providers). If your email provider doesn't use TLS, you can remove or comment out that line.

```
server.starttls()
```

## 5. Combine Soil Sensor and Email source code ( SoilSensorEmail.py)

- Combine the **SoilSensor.py** and **send_email.py** scripts to create a new Python Script **SoilSensorEmail.py.** This script will record 4 daily potentiometer readings. For each reading, the Raspberry Pi will send you an email, reporting the plant status (ie. "Water NOT needed" / "Please water your plant"). Chose appropriate daily times. ***Keep records of your email history for a 3 day period. Include these in your report.***
- **Run, Test, Commit and PUSH**

## 6. Video Demo

- Create a 5 minute video demo / presentation of your project. Begin with a direct to camera introduction describing the purpose of the project. Then show clearly how the the Pi and connected components are set up. Finally demonstrate the Project running, including the interaction with the physical components, as well as the screen output from the python program

## *7. Submission*

- Project #2 Report. PDF format. Use the standard report template. Including Cover Page and Table of Contents and Video URL.
    - Part 1: Document The Agile Process
    - Part 2: A report on the implementation of the Soil Sensor Project. Including
        - Introduction
        - Description of implementation: **Tasks completed (1-7)**. Use screenshots, images of the final working Raspberry Pi & connected components to support your report.
        - Completed source code
        - Git command history..
- Video Presentation: Upload Presentation video to BiliBili account. Include the URL Link to Presentation Video in your report