

Documentation for Smart Waste Monitoring System

Table of Contents

1. Software Needed for Smart Waste Monitoring System	1
2. Necessary Libraries Required in Arduino IDE	1
3. Necessary Libraries Required in Visual Studio Code MQTT Publisher	2
4. Necessary Libraries Required in Visual Studio Code MQTT Subscriber	3
5. Connecting the wires from Arduino Uno Microcontroller to the sensors	5
6. Installation Steps for the Required Software	6
7. Instructions to connect the Arduino Microcontroller to PC, run the Python Scripts and InfluxDB Visualization Tool and Create the Relevant Dashboards	11

1) Software Needed for Smart Waste Monitoring System:

- 1) InfluxDB
- 2) Arduino IDE
- 3) Visual Studio Code
- 4) Python
- 5) MQTTX MQTT Client
- 6) Ubuntu LTS

2) Necessary libraries required in Arduino IDE:

1) Adafruit_VL53L1X

- a. Install via Arduino Library Manager (search for "Adafruit VL53L1X") or from GitHub:
https://github.com/adafruit/Adafruit_VL53L1X
- b. Also requires **Adafruit_BusIO** dependency library.

2) NeoGPS (NMEAGPS)

- a. Install via Arduino Library Manager (search for "NeoGPS") or from GitHub:
<https://github.com/SlashDevin/NeoGPS>

3) NeoSWSerial

- a. Install via Arduino Library Manager (search for "NeoSWSerial") or from GitHub:
<https://github.com/SlashDevin/NeoSWSerial>

4) GPSPORT.h

- a. This is typically part of the NeoGPS examples or a helper file. Ensure it's in your sketch folder or part of the NeoGPS library.

3) Necessary libraries required in Visual Studio Code

MQTT publisher:

- 1) **pip** – Package installer for Python used to install and manage Python libraries
 - a. Install: pip install pip
- 2) **python-dotenv** – For loading environment variables from .env files.
 - a. Install: pip install python-dotenv
- 3) **pyserial** – For serial communication (e.g., with Arduino).
 - a. Install: pip install pyserial
- 4) **json** – Built into Python (Parses MQTT payloads).
- 5) **ssl** – Built into Python (used for secure connections).
- 6) **socket** – Built into Python (used for network communication).
- 7) **time** – Built into Python (for delays and timestamps).
- 8) **threading** – Built into Python (for multithreading).
- 9) **os** – Built into Python (for environment variables and system operations).
- 10) **datetime** – Built into Python (for handling dates and times).
- 11) **tzlocal** – For getting the system's local timezone.
 - a. Install: pip install tzlocal
- 12) **paho-mqtt** – For MQTT communication (IoT messaging).
 - a. Install: pip install paho-mqtt
- 13) **geopy** – For geocoding (converting addresses to coordinates).
 - a. Install: pip install geopy
- 14) **smtplib** – Built into Python (for sending emails).
- 15) **email.mime** – Built into Python (for constructing email messages).

4) Necessary libraries required in Visual Studio Code

MQTT subscriber:

- 1) **pip** – Package installer for Python used to install and manage Python libraries
 - a. Install: pip install pip
- 2) **python-dotenv** - For loading environment variables from .env files.
 - a. Install: pip install python-dotenv
- 3) **json** - Built into Python (Parses MQTT payloads).
- 4) **ssl** - Built into Python (used for secure connections).
- 5) **socket** - Built into Python (used for network communication).
- 6) **csv** - Built into Python (Reads/writes CSV files (e.g., for data exports)).
- 7) **pytz** - Timezone handling
 - a. Install: pip install pytz
- 8) **os** - Built into Python (for environment variables and system operations).
- 9) **glob** - Built into Python (Finds files matching patterns
(e.g., glob.glob("logs/*.csv"))).
- 10) **re** - Built into Python (Parses strings with regex (e.g., extracting bin IDs from filenames))
- 11) **datetime** - Built into Python (for handling dates and times).
- 12) **tzlocal** - For getting the system's local timezone.
 - a. Install: pip install tzlocal
- 13) **paho-mqtt** - For MQTT communication (IoT messaging).
 - a. Install: pip install paho-mqtt
- 14) **influxdb_client** - Interacts with InfluxDB
 - a. Install: pip install influxdb-client
- 15) **smtplib** - Built into Python (for sending emails).
- 16) **email.mime** - Built into Python (for constructing email messages).
- 17) **tensorflow** - Deep learning framework
 - a. Install: pip install tensorflow
- 18) **numpy** - Numerical computing
 - a. Install: pip install numpy
- 19) **pandas** - Data manipulation
 - a. Install: pip install pandas
- 20) **matplotlib** - Data visualization

- a. Install: `pip install matplotlib`

21) scikit-learn - Machine learning metrics (e.g., MAE)

- a. Install: `pip install scikit-learn`

22) Prophet - Time series forecasting

- a. Install: `pip install prophet`

5) Connecting the wires from Arduino Uno Microcontroller to the sensors:

GPS NEO-M8 Modules:

- 1) Connect a male to female Jumper wire from VCC pin of the GPS module to the 5V pin of the Arduino Uno.
- 2) Connect a male to female Jumper wire from GND pin of the GPS module to the GND pin of the Arduino Uno.
- 3) Connect a male to female Jumper wire from RX pin of the GPS module to the D3 pin of the Arduino Uno.
- 4) Connect a male to female Jumper wire from TX pin of the GPS module to the D4 pin of the Arduino Uno.

Time-Of-Flight VL53L1X Sensors:

- 1) Connect a male to female Jumper wire from VIN pin of the ToF sensor to the 3.3V pin of the Arduino Uno.
- 2) Connect a male to female Jumper wire from GND pin of the ToF sensor to the GND pin of the Arduino Uno.
- 3) Connect a male to female Jumper wire from SCL pin of the ToF sensor to the A5 pin of the Arduino Uno.
- 4) Connect a male to female Jumper wire from SDA pin of the ToF sensor to the A4 pin of the Arduino Uno.
- 5) Connect a male to female Jumper wire from GPIO1 pin of the ToF sensor to the D1 pin of the Arduino Uno.
- 6) Connect a male to female Jumper wire from XSHUT pin of the ToF sensor to the D2 pin of the Arduino Uno.

6) Installation Steps for the required Software:

1) InfluxDB (On MQTT Subscriber End)

Steps:

Open the Ubuntu LTS and run the following commands:

Ubuntu and Debian

Add the InfluxData key to verify downloads and add the repository

```
curl --silent --location -O \
https://repos.influxdata.com/influxdata-archive.key
echo
"943666881a1b8d9b849b74caebf02d3465d6beb716510d86a39f6c8e8dac
7515 influxdata-archive.key" \
| sha256sum --check - && cat influxdata-archive.key \
| gpg --dearmor \
| sudo tee /etc/apt/trusted.gpg.d/influxdata-archive.gpg > /dev/null \
&& echo 'deb [signed-by=/etc/apt/trusted.gpg.d/influxdata-archive.gpg]
https://repos.influxdata.com/debian stable main' \
| sudo tee /etc/apt/sources.list.d/influxdata.list
```

Update the package list to ensure you install the latest available version

```
sudo apt-get update && sudo apt-get install influxdb2
```

Start and enable the service

Unmask the service (in case it was masked/prevented from starting previously)

```
sudo systemctl unmask influxdb
```

Reload systemd manager configuration to detect new services

```
sudo systemctl daemon-reload
```

Start the InfluxDB service (this will run it immediately)

```
sudo systemctl start influxdb
```

(Recommended addition) Enable InfluxDB to start automatically on boot

```
sudo systemctl enable influxdb
```

(Recommended addition) Verify the service is running properly

```
sudo systemctl status influxdb
```


Next Steps:

- 1) Open <http://localhost:8086> in a browser.
- 2) **Set Username, Password and Confirm Password** that it is **easy to remember**.
- 3) **Set an initial organization name, initial bucket name and generate an API token**, and **note the organization name, bucket name and token** by **saving** them into a **.txt file**.

2) Arduino IDE (On MQTT Publisher End)

- 1) **Download** the latest Arduino IDE from the official site:
<https://www.arduino.cc/en/software>
 - i. Choose Windows Win 10 and newer (or Windows ZIP file for portable install).
- 2) **Run the installer** (.exe file) and follow the prompts.
 - i. Check "Install USB drivers" (important for Arduino boards).
- 3) **Launch Arduino IDE** after installation.

Install Board Support (e.g., Arduino AVR Boards)

1. Open **Arduino IDE** → **File** → **Preferences**.
2. In **Additional Boards Manager URLs**, add:

https://m5stack.oss-cn-shenzhen.aliyuncs.com/resource/arduino/package_m5stack_index.json

3. Go to **Tools** → **Board** → **Boards Manager**, search for **Arduino AVR Boards**, and install.
4. Select your board:
 - i. **Tools** → **Board** → **Arduino AVR Boards** → **"Arduino Uno"**

3) Visual Studio Code

- 1) **Download** VS Code for Windows:
<https://code.visualstudio.com/download>
- 2) **Run the installer** (.exe file) and follow the prompts.
- 3) **Launch VS Code** after installation.
- 4) Install Extensions
 - i. Open Extensions (Ctrl+Shift+X) and install:
 1. Python (for using Python scripts).
 2. Jupyter (for using Jupyter Notebook for running time series machine learning models).

4) Python

Step 1: Download Python

- i. Go to the official Python website:
<https://www.python.org/downloads/>

- ii. Click "**Download Python 3.x.x**" (latest stable version).

Step 2: Run the Installer

- 1) Open the downloaded .exe file (e.g., python-3.11.5-amd64.exe).
- 2) **Check these boxes during installation:**
- 3) "**Add Python to PATH**" (critical for command-line usage).
- 4) "**Install launcher for all users**" (recommended).
- 5) Click "**Install Now**" (default settings are fine).

Step 3: Verify Installation

- 1) Open Command Prompt (Win + R -> type cmd -> Enter)
- 2) Run: python --version
- 3) Expected Output: Python 3.x.x (Depending on the Python Version you've installed).

5) MQTTX MQTT Client

Download & Install

1. **Download** MQTTX for Windows based on your system architecture like 32-bit, 64-bit or AMD:
<https://mqttx.app/downloads>
2. **Run the installer** (.exe file) and follow the prompts.
3. **Launch MQTTX** after installation.

Test MQTT Connection

1. Open MQTTX and click "**New Connection**".
2. Enter:
 - **Name:** Real-Time Bin Monitoring
 - **Host:** broker.emqx.io
 - **Port:** 8883
3. Click "**Connect**".
4. Click "**New Subscription**" and input "**sensor/data**" as the topic and select QoS as **1** → If successful, you can receive real-time data waste level monitoring information.

6) Ubuntu LTS

Step 1: Enable WSL (Windows Subsystem for Linux)

Before installing Ubuntu, you must enable WSL:

1) Open PowerShell as Administrator

- Press Win + X → Select "Windows Terminal (Admin)" or "PowerShell (Admin)".

2) Run the following command:

- `wsl --install`
- This enables WSL and installs the latest Ubuntu by default.

3) If WSL is already installed, just enable it:

`dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart`

`dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart`

4) Restart your computer when prompted.

5) Press Win + R, type `optionalfeatures`, and hit Enter.

6) Check and ensure that both the checkboxes for Windows Subsystem for Linux and Virtual Machine Platform are checked.

7) Restart your computer when prompted.

Step 2: Install Ubuntu LTS from Microsoft Store

1) Open Microsoft Store

- Press Win + S → Search for "**Microsoft Store**" → **Open** it.

2) Search for "Ubuntu"

- Choose "**Ubuntu << Version >> LTS**" (e.g., Ubuntu 22.04 LTS).

3) Click "Install"

- Wait for the download (~300MB) and installation.

4) Launch Ubuntu

- Open Start Menu → Search for "Ubuntu" → Click to launch.

5) Set Up a Username & Password

- When prompted, enter a new UNIX username and password (does not need to match Windows credentials).

Step 3: Update Ubuntu (Recommended)

Run these commands in the Ubuntu terminal:

`sudo apt update && sudo apt upgrade -y`

Step 4 (Optional): Set WSL 2 as Default

WSL 2 is faster and more efficient. To set it as default:

`wsl --set-default-version 2`

Troubleshooting


1. "WSL --install" Fails?

- **Manually install WSL:**
`wsl --install -d Ubuntu`
- **If issues persist, check:**
 - Virtualization is enabled in BIOS (Intel VT-x / AMD-V).

2. Ubuntu Not Launching?

- **Reset WSL:**
`wsl --shutdown`
`wsl -t Ubuntu`
`wsl --unregister Ubuntu`
- **Reinstall Ubuntu from the Microsoft Store.**

7) Instructions to connect the Arduino microcontroller to PC, run the Python scripts and launch the InfluxDB visualization tool:

- 1) **Download** all the necessary files from GitHub (<https://github.com/RoyTeong/Smart-Waste-Monitoring-Capstone-Project>) based on the instructions listed in the README file.
- 2) **Launch** Arduino IDE on MQTT Publisher's PC.
- 3) **Connect** the Arduino Microcontroller via USB into the MQTT publisher's PC.
- 4) If USB for the Arduino Uno is not recognized, try to unplug and plug in again multiple attempts until connection can be established.
- 5) **Select** the Arduino Uno Board and a respective **serial port**, i.e. COM3
- 6) **Open** Serial Monitor under Tools -> Serial Monitor and set the baud rate to **115200**.
- 7) **Click** on the **Upload** button with the  symbol at the top left-hand corner of the window.
- 8) **Download** the .env file from GitHub and **save** to the local PC on **both MQTT Publisher and Subscriber (same directory as the Python Script)**
- 9) **Open** a new terminal and **type** `python mqtt_publish.py` on **1 PC** and `python mqtt_subscribe.py` on **another PC**, then **press Enter**. (**Ensure Directory shown on the terminal is the same as the Python Script that was saved to.**)
- 10) **Launch Ubuntu <<version>> LTS** on **MQTT Subscriber's PC**.
- 11) **Type** `sudo systemctl start influxdb` and **Press Enter**.
- 12) **Input** the password that you've set during installation of Ubuntu LTS and **Press Enter**.
- 13) **Navigate** to <http://localhost:8086> on a web browser.
- 14) **Input** your username and password that you've set during installation of InfluxDB, and **press Enter**.
- 15) **Navigate** to Dashboards, click on create Dashboard and select new dashboard. Create a title for the Dashboard, i.e. Bin Level Monitoring <<month>> <<year>>, replace month and year with the actual month and

year, and add the necessary cells as stated with the necessary visualization types and flux query in the waste_monitoring.txt and click on the tick button at the top right hand corner of the window, and repeat this at the beginning of each month.

- 16) **Download and Open** the Predictive_Analytics_Future Waste Levels Jupyter Notebook file and click on **Run All** at the top of the navigation bar of VS Code to perform predictive analytics to predict and forecast future waste levels based on historical data collected.