

##### User Manual

Python App for updating & hosting Web Pages using DDC data

**Table of Contents**

[1. Introduction 3](#_Toc119079413)

[1.1 Overview 3](#_Toc119079414)

[1.2 Software Description 3](#_Toc119079415)

[1.3 Platform Requirements 3](#_Toc119079416)

[2. Getting Started 4](#_Toc119079417)

[2.1 Required Applications 4](#_Toc119079418)

[2.2 Directories & Contents 4](#_Toc119079419)

[3. Setting up 5](#_Toc119079420)

[3.1 Initial Setup 5](#_Toc119079421)

[3.1.1 Dependencies & main.pyw 5](#_Toc119079422)

[3.1.2 Excel Raw File 6](#_Toc119079423)

[3.1.3 HTML & soup.pyw Files 7](#_Toc119079424)

[4. Running the Program 9](#_Toc119079425)

[4.1 Procedures 9](#_Toc119079426)

[4.1.1 File Configuration 9](#_Toc119079427)

[4.1.2 Launching 10](#_Toc119079428)

# **1. Introduction**

## **1.1 Overview**

This app can read and extract point information from Direct Digital Controllers (DDC) and use those values to update web pages.

## **1.2 Software Description**

The software comprises four types of python files – main.pyw, soup.pyw, app.pyw, and wsgi.pyw.

The **main.pyw** file extracts point information from Direct Digital Controllers (DDC) and stores the values in an excel sheet.

The **soup.pyw** file updates specific values on the HTML web pages using the point information excel sheet generated from the DDC.

The **app.pyw** and **wsgi.pyw** files are used to create the hosting platform for the HTML web pages.

## **1.3 Platform Requirements**

* Modern Operating System:
  + Windows 7 or 10
  + Mac OS X 10.11 or higher, 64-bit
  + Linux: RHEL 6/7, 64-bit
* x86 64-bit CPU (Intel/AMD architecture)
* 4 GB RAM
* 5 GB free disk space

# **2. Getting Started**

## **2.1 Required Applications**

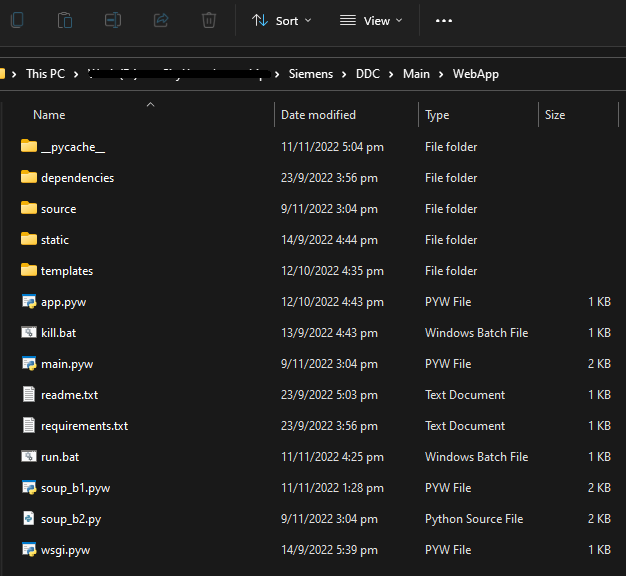
Before this program can be used, the following needs to be downloaded and installed:

* [Python](https://www.python.org/downloads/)
* [Anaconda](https://www.anaconda.com/products/distribution)
* [Visual Studio Code](https://code.visualstudio.com/download)

## **2.2 Directories & Contents**

There are several directories present in the program. The contents of each directory are listed below:

* **WebApp:** This is the main directory containing the python program files and all the other directories.
* **Dependencies:** contains all the dependencies that need to be installed for the program to run properly
* **Source:** contains the excel sheets – “raw” and “data”
  + Raw – this file is used to set the requirements of the data extraction process for the main.pyw file
  + Data – consists of the data extracted from the DDC using the main.pyw file
* **Static:** contains the CSS files, and media files for each web page
* **Templates:** contains the HTML file for each web page

****

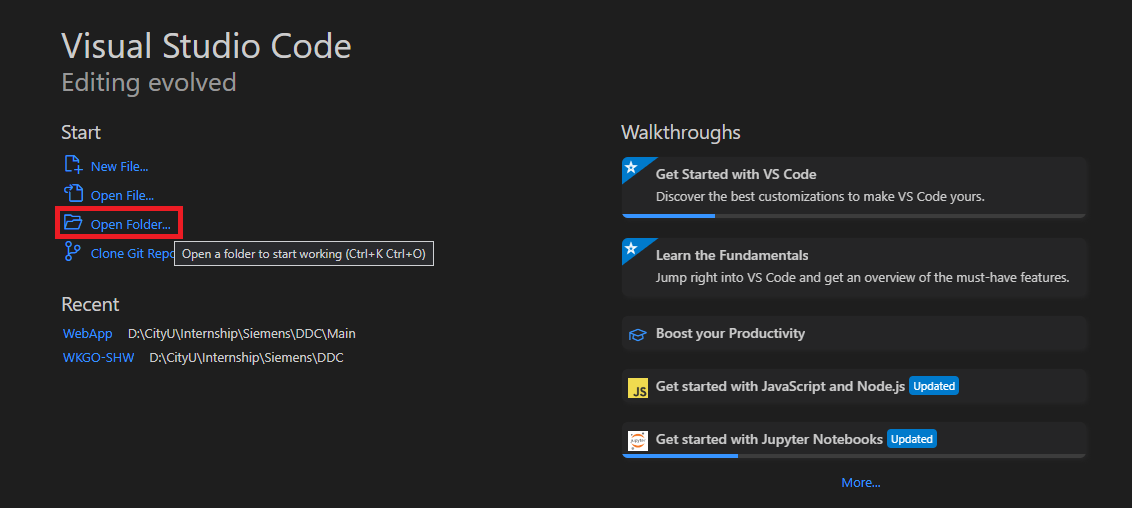
# **3. Setting up**

## **3.1 Initial Setup**

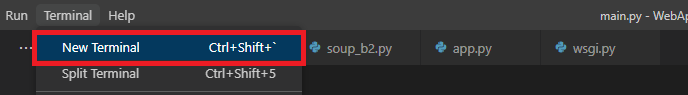
To run the program, the following steps need to be completed:

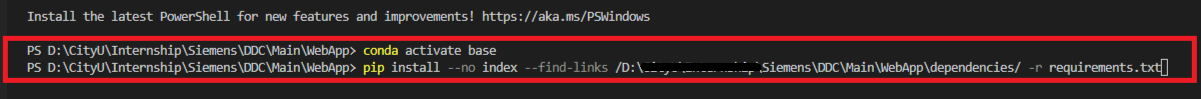
### **3.1.1 Dependencies & main.pyw**

* Open Anaconda Navigator, then launch VS Code and open the folder “WebApp”



* Open a new terminal, then write the following code "pip install --no-index --find-links /**path to dependencies directory**/ -r requirements.txt"





* + The “**path to dependencies directory**” is specific to the device. For example:



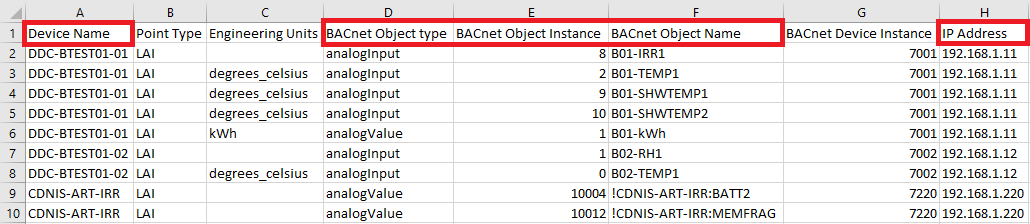
* In VS Code, open the **main.pyw** file from the WebApp folder
  + Check the LAN IP Address of the host device and update **line 7** in **main.pyw**: bacnet = BAC0.connect(ip=' ')
  + **Line 8** refers to the time interval of data extraction from the DDC. It is measured in milliseconds (ms) and shall be updated according to the user’s preference.



### **3.1.2 Excel Raw File**

The **raw.csv** file contains the device and point list. To extract more point information from different DDCs, the raw.csv file needs to be updated accordingly.

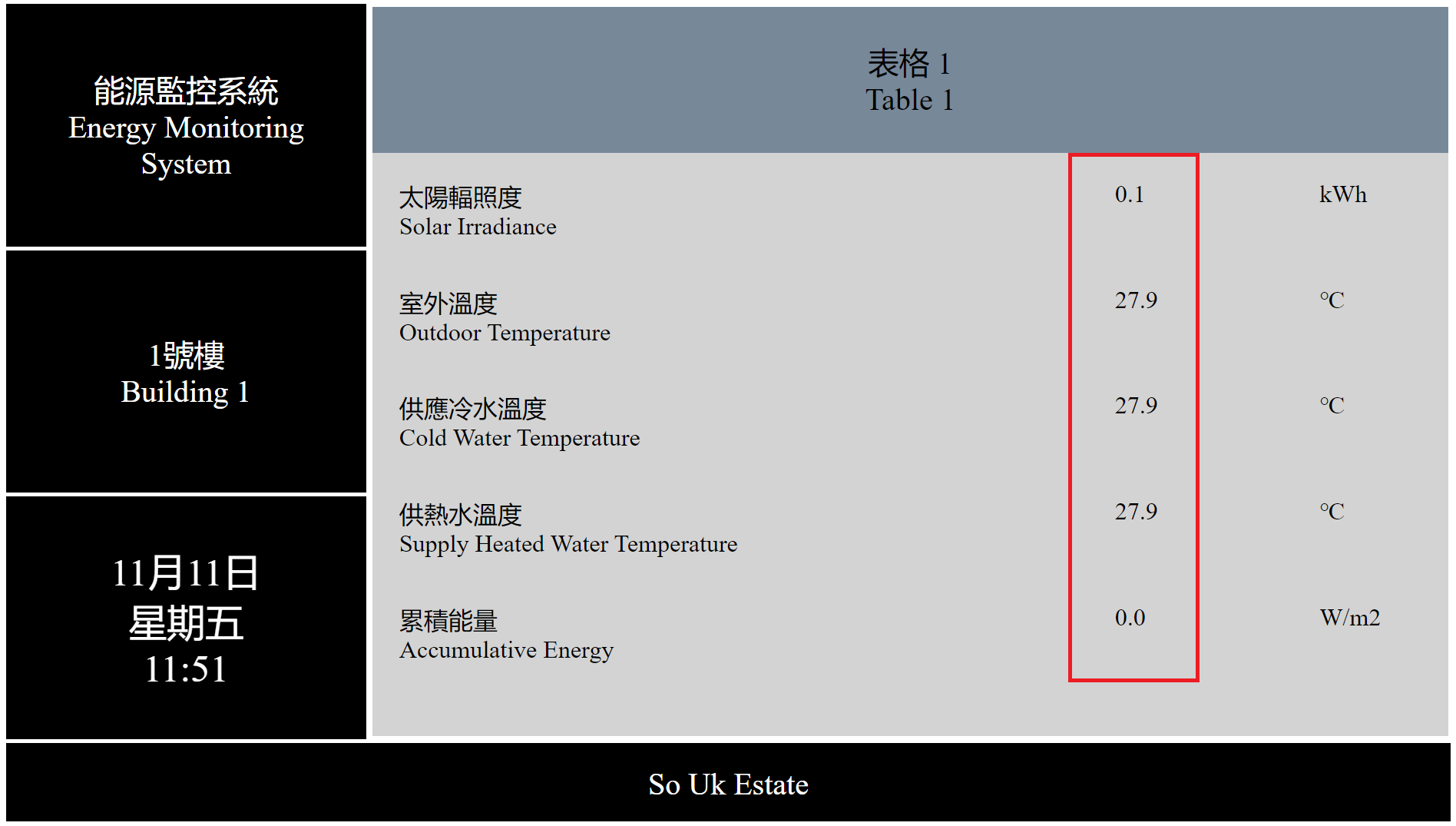
* Add the **Device Name** to the first column
* Input the corresponding Object type. If the type is **Analog Input**, **Analog Output**, or **Analog Value**, it has to be written as **analogInput, analogOutput,** and **analogValue**,respectively.
* Finally, update the **Object Instance**, **Object Name**, and **IP Address**.



### **3.1.3 HTML & soup.pyw Files**

To ensure that the table values on the web pages are correct, some elements must be cross-checked between the HTML and soup.pyw files.

The **soup.pyw** file uses data from the **data.csv** file.



The steps are as follows:

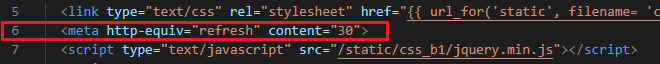
* The **id** on the HTML files must be the same as the **Object Name** on the data.csv file.
  + Please make sure to update the **index.html** file, **not** the **new.html** file

|  |  |
| --- | --- |
|  |  |

* The **updating interval** (ms) for the HTML file can be found on the **soup.pyw** file on **line 12**.



* The **refresh time** (ms) for the HTML file can be found on **line 6** of the **index.html** file.
  + Make sure to keep the **refresh time** closer to the **updating interval** on the **soup.pyw** file



* The intervals (ms) of the sliders for the **footer**, **top left section**, **media**, and **tables** are 10000, 10000, 900000, and 225000.

# **4. Running the Program**

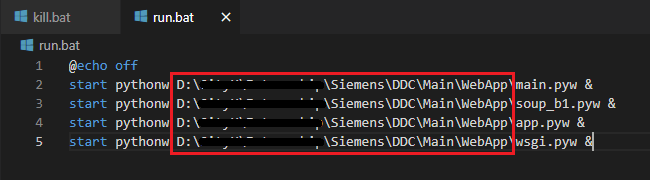
## **4.1 Procedures**

The procedures for running the program are listed below.

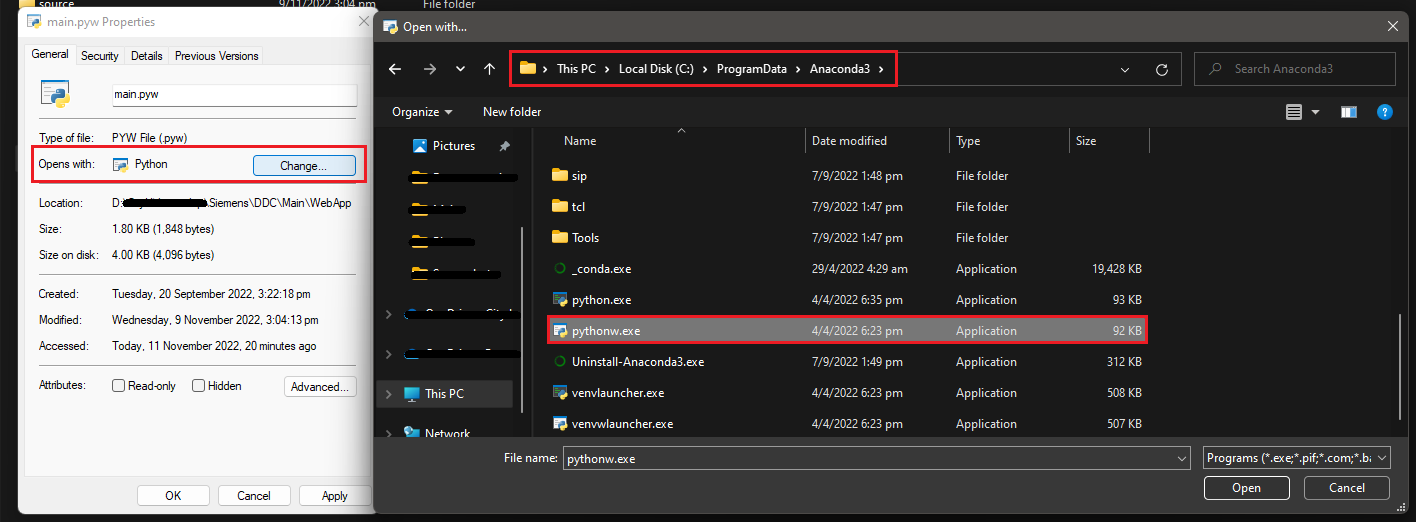
### **4.1.1 File Configuration**

The following steps are to be followed to configure the python program files:

* Open the **run.bat** file in VS Code and check if the path to the **.pyw** files are correct.



* In the folder containing the **.pyw** files, right click on each file and select properties
  + In properties, change **“Opens with”** to **pythonw.exe** from “C:\Program Data\Anaconda3”



### **4.1.2 Launching**

The following steps are to be followed to launch the WebApp:

* Launch **CMD.exe** from Anaconda Navigator
  + In CMD.exe, write **“cd”** then the **path** to **WebApp directory**
  + Then write **“run.bat”** to launch the python program
  + To close the program, write **“kill.bat”**

