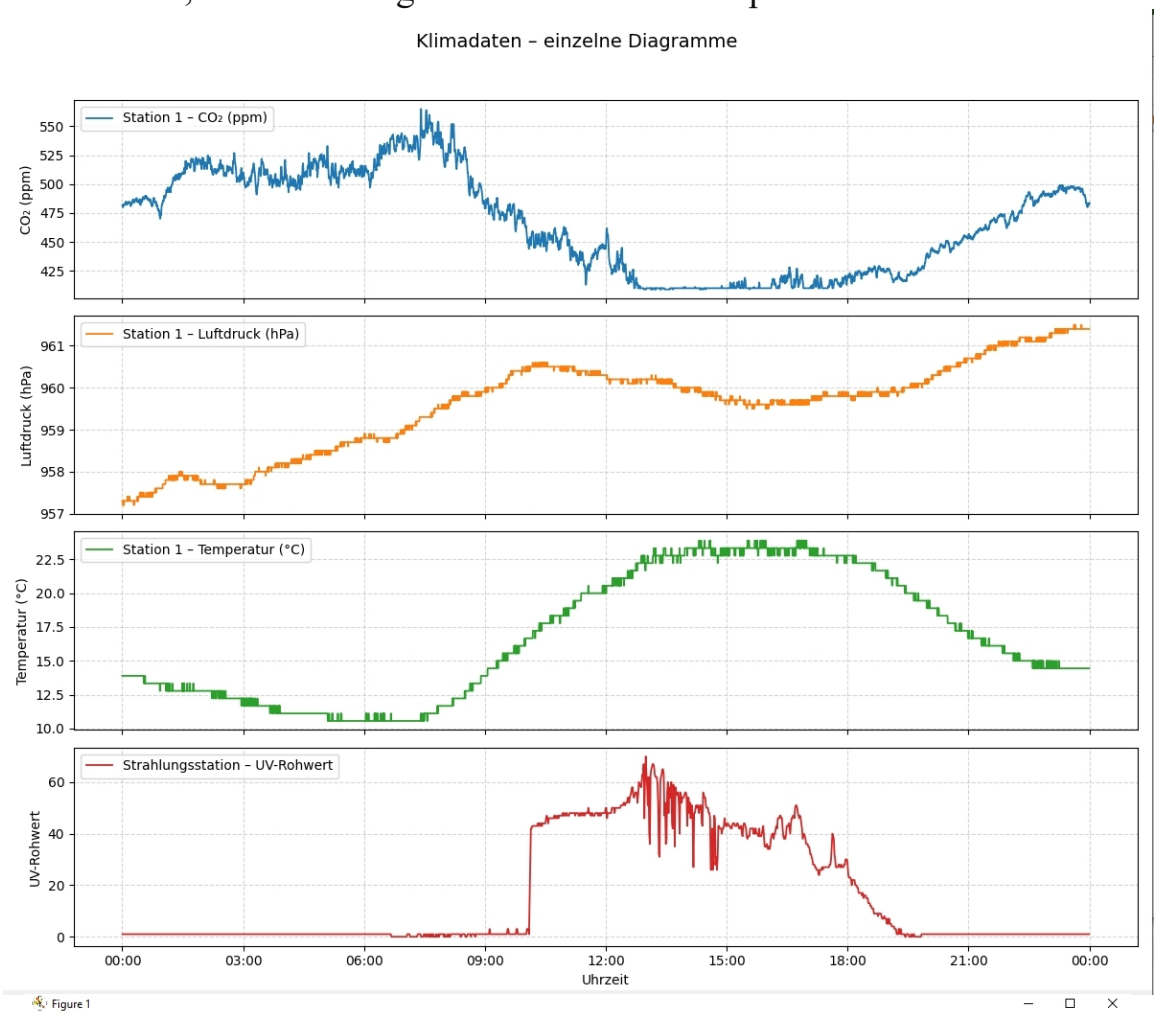
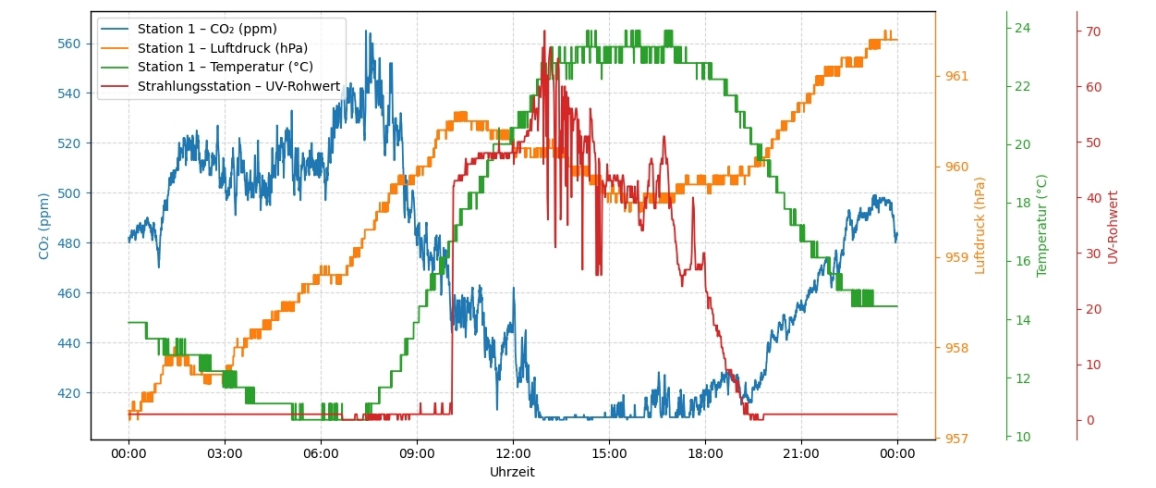
Interaktive Klimadatenanalyse und Visualisierung

Übersicht:

Ein interaktives, Python-basiertes Werkzeug zur Untersuchung von Umwelt- und Klimadaten im zeitlichen Verlauf. Wähle einen bestimmten Tag oder eine Kalenderwoche aus, stelle bis zu vier Messgrößen zusammen und visualisiere ihre Zusammenhänge mit Matplotlib. Vergleiche Messdaten verschiedener Stationen, um Muster, Trends und mögliche Wechselwirkungen zwischen Temperatur, CO_2, Feinstaub, UV-Strahlung und weiteren Umweltparametern zu erkennen.





Projektbeschreibung:

Mit einer interaktiven Kalenderauswahl für **einzelne Tage** und **Kalenderwochen** sowie eine dynamisch erzeugte Sensorauswahl können bis zu **vier** Messgrößen verschiedener Stationen ausgewählt werden. Ihre zeitlichen Verläufe werden basierend auf der Tkinter Oberfläche mit Matplotlib grafisch dargestellt um mögliche Wechselwirkungen, Korrelationen und Zusammenhänge der Sensordaten zu untersuchen.

Dieses Programm, **Climate_Data_Analysis_DE.py** ist ebenfalls komplett kostenlos, wurde in der Programmiersprache **Python** mit Unterstützung von OpenAI ChatGPT entwickelt. Die Sensordaten werden von Raspberry-Pi-Zero-2-W-Stationen erfasst und auf einer SSD gespeichert. Die grafische Auswertung erfolgt auf einem Windows-PC, alternativ auch auf einen Raspberry PI.
Die Klimastationen benützen einen preisgünstigen Raspberry PI Zero W , allerdings würde dieser für die Auswertung der Sensordaten überfordert sein, also:

- Raspberry Pi Zero 2 W: Messdaten sammeln und auf der SSD speichern.
- Windows-PC: Daten über Samba nach `H:\data` kopieren und dort schnell auswerten.

Ausführliche Informationen zu meiner Klimastation(en) findet man auf meiner Homepage: **https://www.pdp11gy.com/**
https://pdp11gy.com/Klimastation.html
mit download links für die gesamten Software und Projekt Informationen:
Github: **https://github.com/pdp11gy/Environmental-and-Climate-Data-Station**

Voraussetzungen und Installation

Im Ordner **Diagrams/Diagram-Example.pdf** sind bereits die einzelnen Schritte beschrieben um die Voraussetzungen zu erfüllen. Hier nochmals die einzelnen Schritte:

Erforderlich ist die open source library **matplotlib**
Diese muss eventuell noch installiert werden mit
Windows-Eingabeaufforderung: Test ob python istalliert ist: **python –version**
Falls nicht, **python** und **Thonny**(=empfohlen) installieren.
Matplotlib installieren: **python -m pip install matplotlib**
Bei Thonny unter den Menü "Tools": manage Packages noch matplotlib eintragen

Raspberry Pi OS : sudo apt update und **sudo apt install python3-matplotlib**

Datenübertragung vom Raspi server nach windows.

dies kann mit der Windows-Eingabeaufforderung und **scp** erfolgen, z.B.
scp pi@192.168.178.xx:/mnt/ssd/data/data.json F:

Mein Favorit ist allerdings ein Windows Netzlaufwerk, da ein Raspberry PI Zero bei der Datenauswertung mit grafischer Darstellung schnell an seine Grenzen kommt.
Die **rechenintensive Auswertung läuft auf meinem Windows-PC**, während der Raspberry Pi weiterhin nur seine eigentliche Aufgabe erledigt – Sensoren auslesen und Daten speichern.
Dazu muss zuerst auf den Raspberry PI Server **samba** installiert werden mit **sudo apt update** und s**udo apt install samba .** Danach die Datei /etc/samba/smb.conf am Ende mit folgenden Eintrag ergänzen:

```
[Klimadaten]
   path = /mnt/ssd
   browseable = yes
   read only = yes
   guest ok = yes
```

read only = yes , weil :Raspberry Pi sammelt die Daten und Windows liest sie nur aus.
Danach Samba neu starten: **sudo systemctl restart smbd**
Unter Windows dann im Explorer das Netzwerk Laufwerk aktivieren.

Konfiguration

Die Konfiguration ist in der Datei **config.ini** hinterlegt und kann mit einen beliebigen Test Editor bei Bedarf angepasst werden.

Windows	**Raspberry PI**
[Paths]	[Paths]
data_folder = H:/**data**	data_folder = /mnt/ssd/data
data_filename = daten.json	data_filename = daten.json
output_folder = D:\result	output_folder = /mnt/ssd/result
output_filename = **klimadaten**.png	output_filename = klimadaten.png

Beim Programmstart wird immer diese Datei **config.ini** ausgelesen. Es sind die Grundeinstellungen und müssen vorhanden sein.

Wichtige Info:

Das **Climate_Data_Analysis** Programm greift immer auf den **/data** Ordner zu.
In diesem Ordner befinden sich die eigentlichen .json - Daten, sortiert nach Jahr,Woche und Tag.

Beispiel: Aufteilung(15-JUN-2026): /data/2026_KW24/2026-06-15 /daten.json

```
data/
└── 2026_KW24/
    └── 2026-06-15/
        └── daten.json
```

Im mitgelieferten Ordner /data sind für Referenz und Test Zwecke Beispiel Daten von meinen Wetterstationen für KW28 und KW37 beigefügt. Bei KW28 stehen nur die Daten von meiner Station-1 zur Verfügung. Bei KW37 sind die Daten von Station-1, Station-2 und der Strahlenstation enthalten. In der weiteren Beschreibung wird immer auf diese Dateien referenziert.
(Bei Bedarf kann ich alle meine bisherigen Daten zur Verfügung stellen)

output_filename:

Um ein Überschreiben der Klimadaten mit gleichen Datum zu verhindern prüft das das Programm vor jedem Speichern, ob der Dateiname schon existiert, und verwendet automatisch die nächste freie Versionsnummer. Auch nach einem Neustart funktioniert das, weil die vorhandenen Dateien auf der Festplatte geprüft werden. Beispiel: Bei wiederholten Tagesauswertungen entstehen automatisch:
V1-**klimadaten**_2026-10-16.png
V2-**klimadaten**_2026-10-16.png
V3-**klimadaten**_2026-10-16.png
Auch wie folgt implementiert bei der Wochenanalyse:
V1-**klimadaten**_KW41_2026.png
V1-**klimadaten**_KW41_2026_statistik.csv

Bedienung und Beispiele

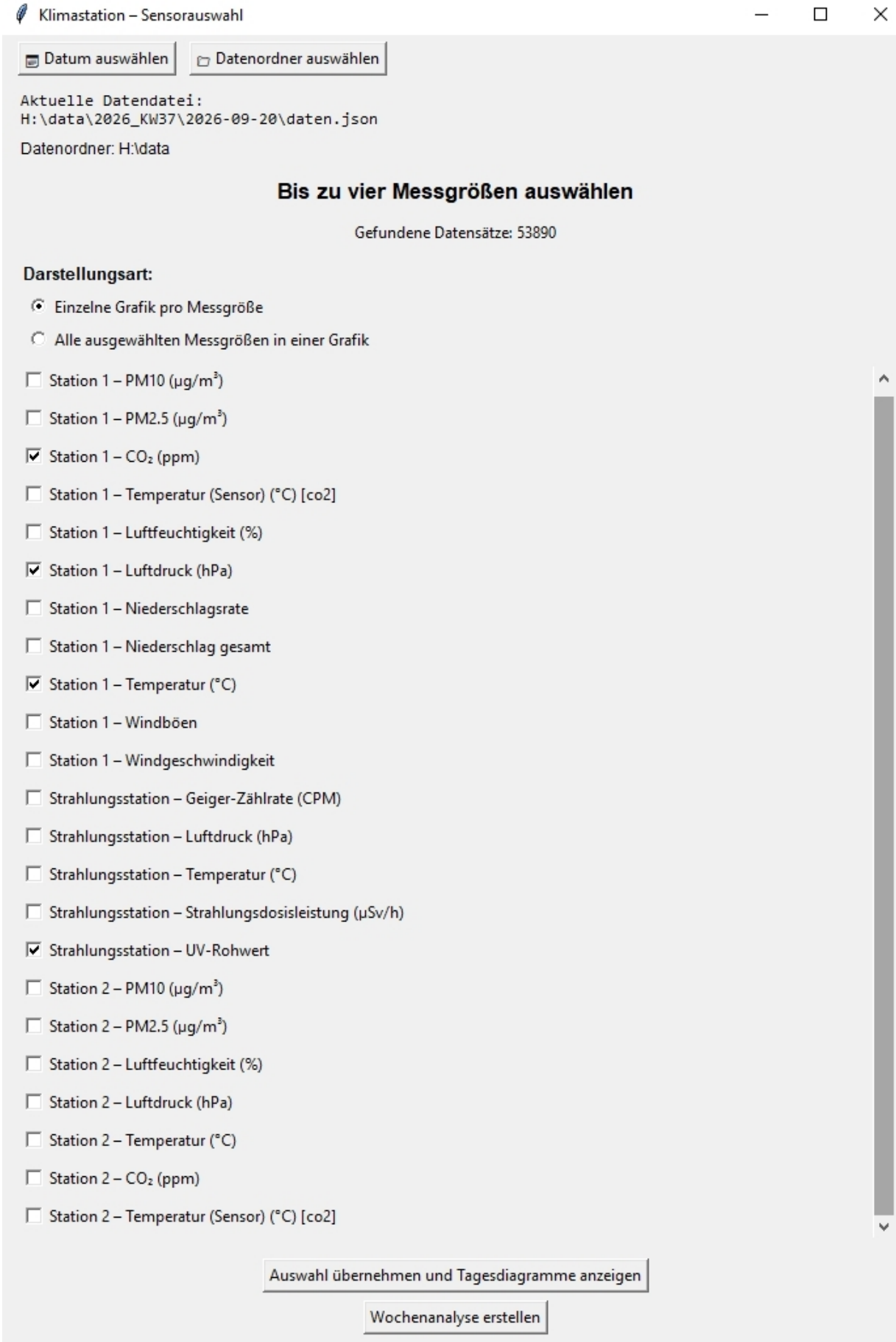
Das Programm kann auf verschiedene Art gestartet werden:

1) Windows-Eingabeaufforderung: **python3 Climate_Data_Analysis_DE.py**
2) Mit Thonny: Menu:"FILE": Open : **Climate_Data_Analysis_DE.py -->** start
3) Optional: Sehr komfortabel kann man das Programm mit dem Pyinstaller starten. Dieser muss mit folgenden Schritten installiert und konfiguriert werden:

```
 python -m pip install --upgrade pyinstaller
 python -m PyInstaller --clean --onefile --name Klimadaten_DE
Climate_Data_Analysis_DE.py
```

Damit wird des .exe File Klimadaten_DE.exe erzeugt, welches dann komfortabel direkt vom Windows Desktop , auch als ICON gestartet werden kann.

Das Programm meldet sich wie hier :



Oben links befindet sich der Button "**Datum auswählen**" . Bei Aktivierung erscheint folgendes Kalender Symbol:



Hiermit wird das gewünschete Datum ausgewählt und wird darunter wie folgt angezeigt: H:\data**2026_KW37\2026-09-20**\daten.json.

Der Button "**Datenordner auswählen**" rechts danben öfnnet eine Dialog Box um das Laufwerk festzulegen wo sich die Daten befinden, hier im Beispiel **H:\data**

Darunter kann die Darstellungsar festgelegt werden, also entweder "**Einzelne Grafik pro Messgröße**" oder "**Alle ausgewählten Messgrößen in einer Grafik**".

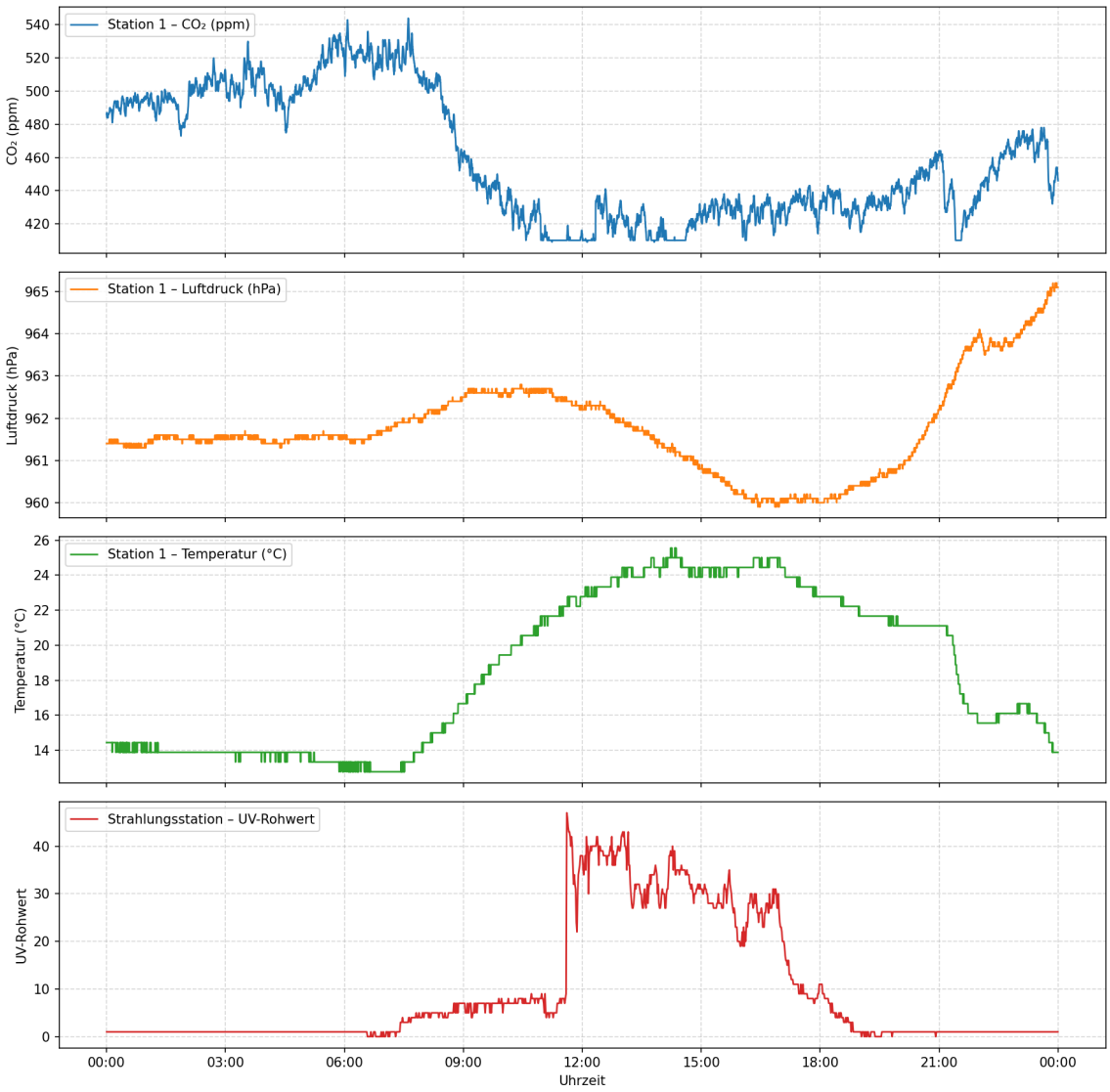
Darunter folgt die **Auswahl der Sensoren**, hier wie im obigen Beispiel mit 4 Sensoren: CO_2 (ppm), Luftdruck (hPa), Temperatur (°C), UV-Rohwert

Im ersten Beispiel wird zuerst der Button "**Auswahl übernehmen und Tagesdiagramme anzeigen**" aktiviert.

Es erscheint folgende Meldung: mit anschliessender Grafik:

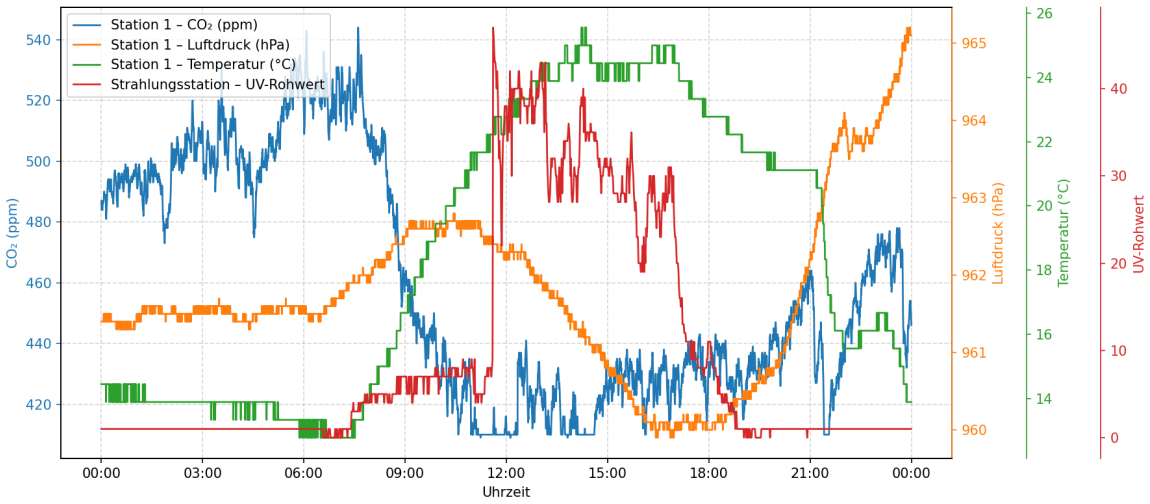
Ausgewählte Messgrößen:
Station 1 – CO_2 (ppm) -> ('station1', 'co2', 'co2')
Station 1 – Luftdruck (hPa) -> ('station1', 'ws3', 'pressure')
Station 1 – Temperatur (°C) -> ('station1', 'ws3', 'temperature')
Strahlungsstation – UV-Rohwert -> ('station1u2', 'geiger', 'uv_raw')
Datendatei: H:\data\2026_KW37\2026-09-20\daten.json
Diagramm gespeichert (neue Version): H:\result\V1-klimadaten_2026-09-20.png
Anzahl dargestellter Messgrößen: 4
Darstellungsart: einzeln

Klimadaten – einzelne Diagramme



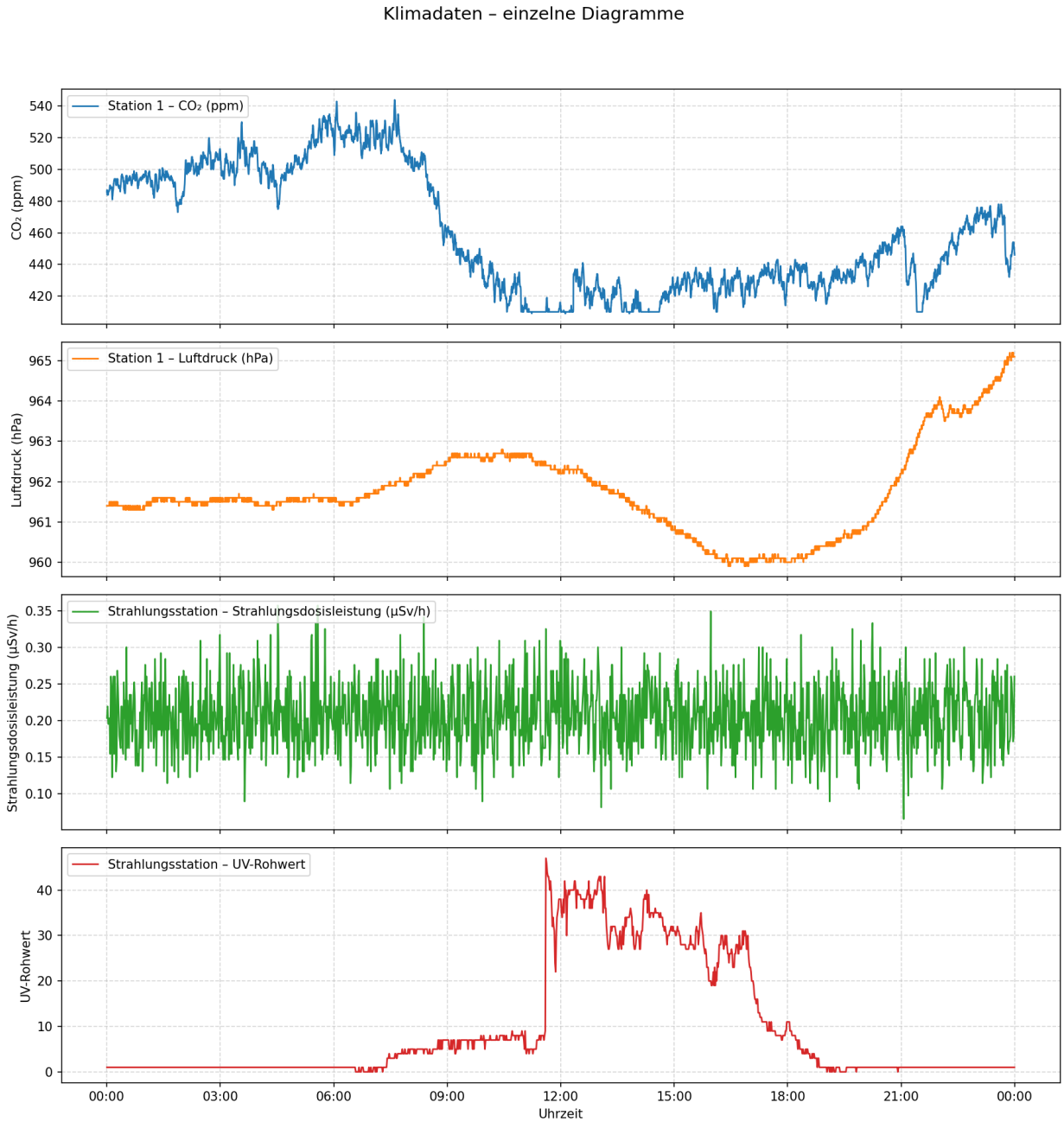
Nun die Darstellung: "**Alle ausgewählten Messgrößen in einer Grafik:**

Klimadaten – gemeinsames Diagramm



Mit der Strahlenstation ist eine Messung der Radioaktivität und UV-Strahlung im Zusammenhang mit Luftdruck und CO2 möglich . Es erscheint folgende Meldung mit Grafik:

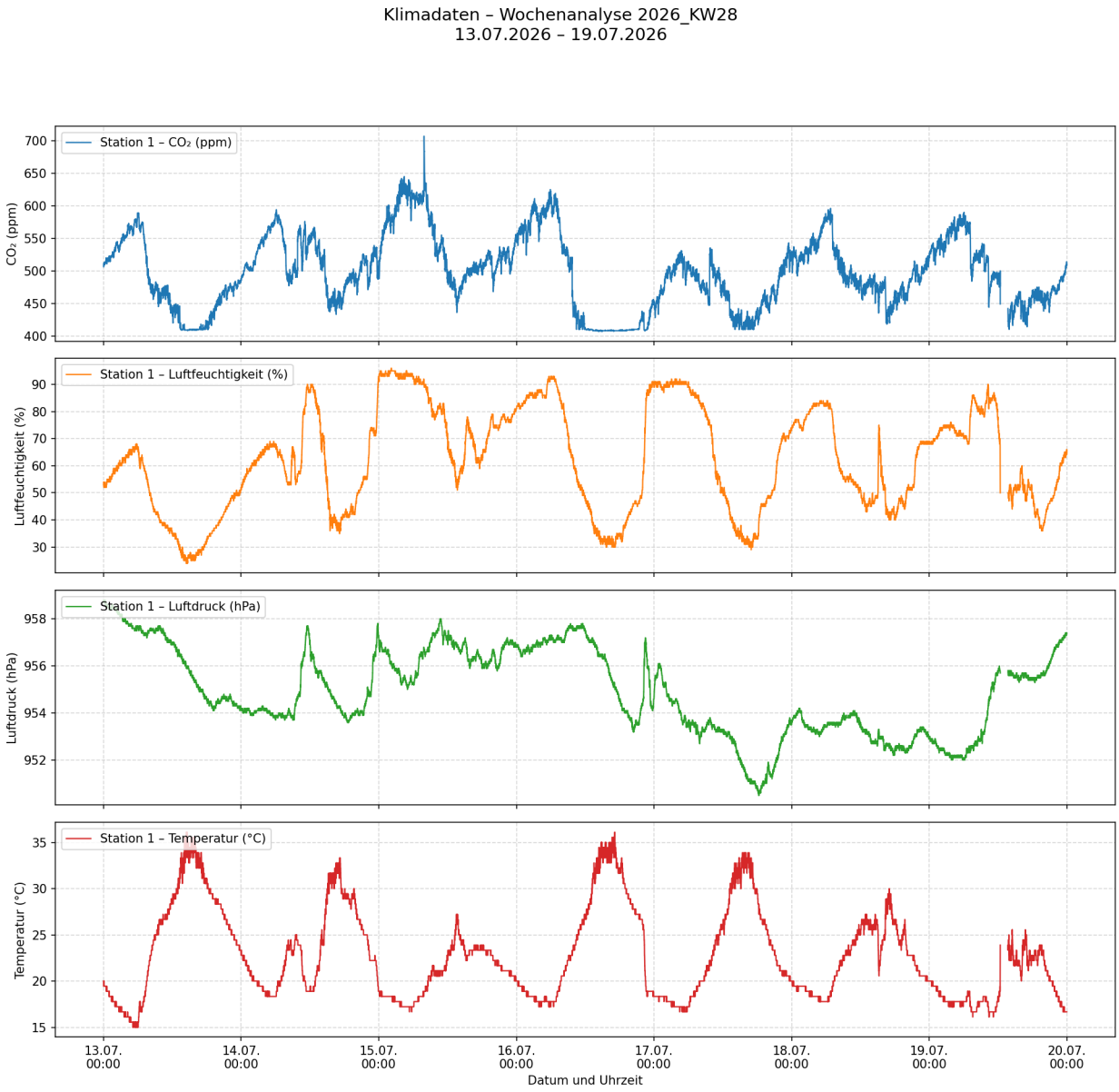
Ausgewählte Messgrößen:
Station 1 – CO_2 (ppm) -> ('station1', 'co2', 'co2')
Station 1 – Luftdruck (hPa) -> ('station1', 'ws3', 'pressure')
Strahlungsstation – Strahlungsdosisleistung (µSv/h) -> ('station1u2', 'geiger', 'usvh')
Strahlungsstation – UV-Rohwert -> ('station1u2', 'geiger', 'uv_raw')
Datendatei: H:\data\2026_KW37\2026-09-20\daten.json
Diagramm gespeichert (neue Version): H:\result\V1-klimadaten_2026-09-20.png
Anzahl dargestellter Messgrößen: 4
Darstellungsart: einzeln



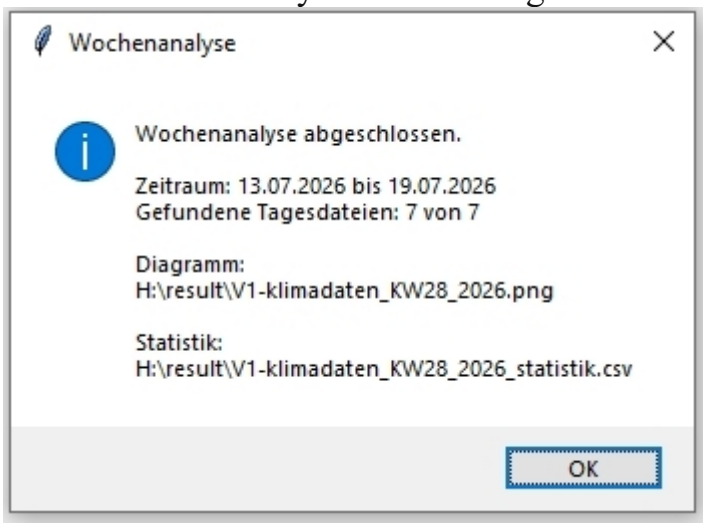
(Eine Wechselwirkung mit der Radioaktivität ist nicht ersichtlich)

Hier ein Beispiel mit einer Wochenanlyse. Dazu wählen wir ein Datum von einer Hitzeperiode in KW 28 aus., z.B. \2026_KW28\2026-07-13. Wird die Wochenalyse gestartet kommt folgende Meldung mit Grafik:

Ausgewählte Messgrößen:
Station 1 – CO_2 (ppm) -> ('station1', 'co2', 'co2')
Station 1 – Luftfeuchtigkeit (%) -> ('station1', 'ws3', 'humidity')
Station 1 – Luftdruck (hPa) -> ('station1', 'ws3', 'pressure')
Station 1 – Temperatur (°C) -> ('station1', 'ws3', 'temperature')
Datendatei: H:\data\2026_KW28\2026-07-13\daten.json
Diagramm gespeichert (neue Version): H:\result\V1-klimadaten_2026-07-13.png
Anzahl dargestellter Messgrößen: 4
Darstellungsart: einzeln
Wochenanalyse: 2026-07-13 bis 2026-07-19
Geladene Tage: 7 von 7
Wochen-Diagramm gespeichert (neue Version): H:\result\V1-klimadaten_KW28_2026.png
Wochenstatistik gespeichert (neue Version): H:\result\V1-klimadaten_KW28_2026_statistik.csv



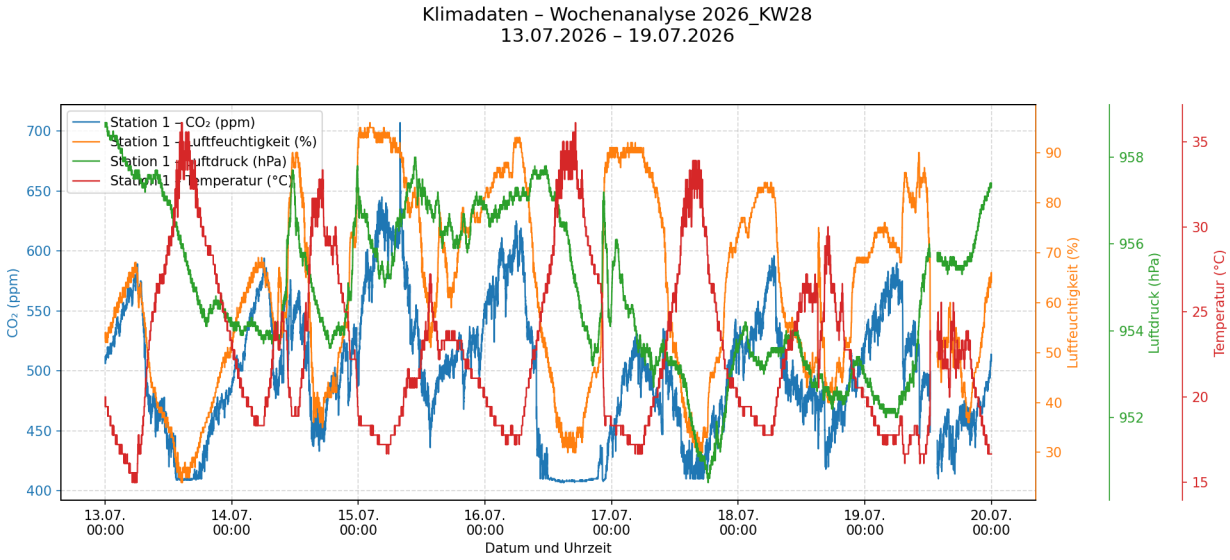
Nach Abschluß der Analys erscheint folgende Meldung:



Zusätzlich erstellt das Programm eine CSV-Datei mit Anzahl der Messwerte, Minimum, Mittelwert und Maximum für jede ausgewählte Messgröße.
Die Statistik Datei steht im .csv Format zur Verfügung mit folgenden Inhalt:

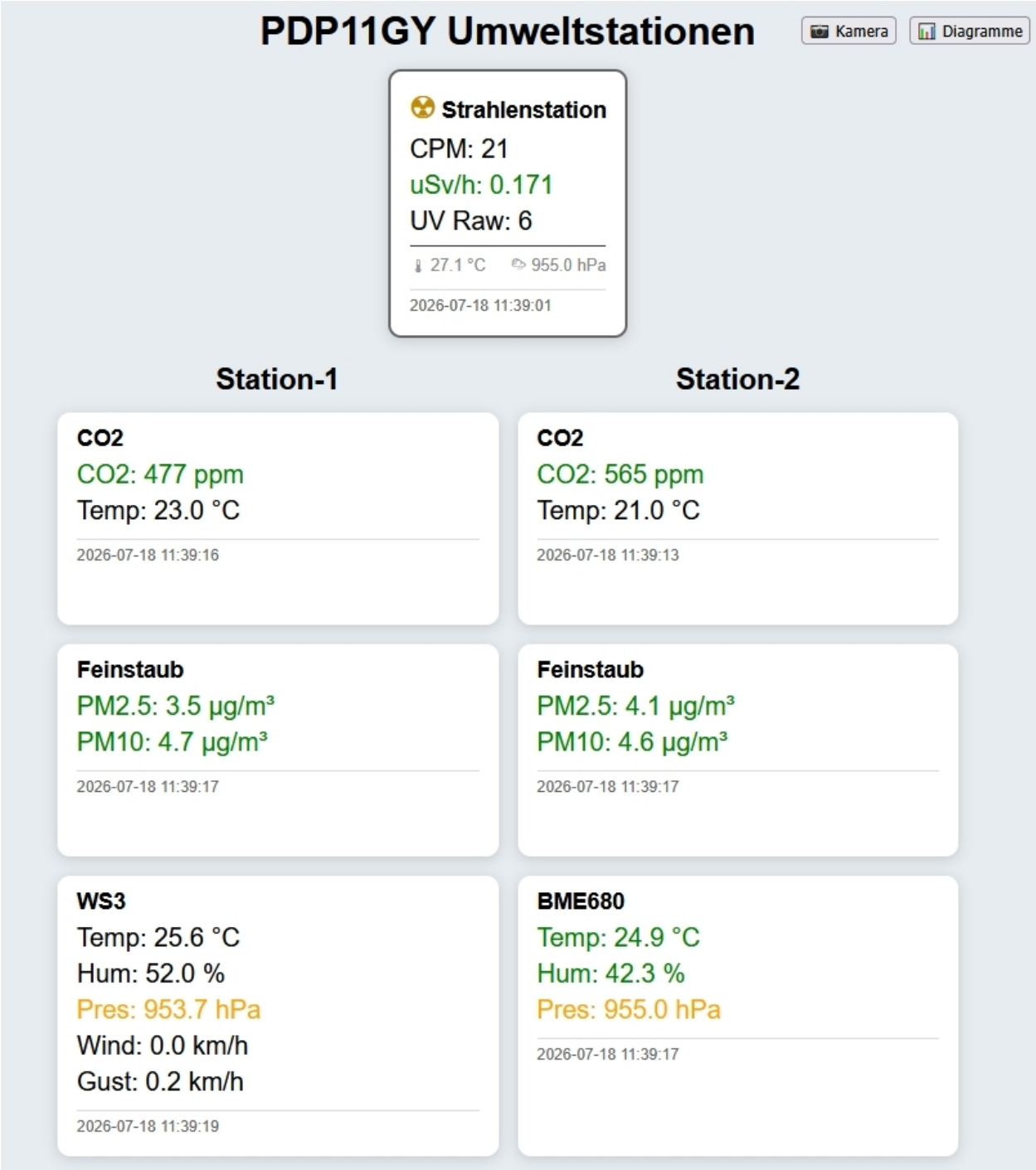
Station;Sensor;Messgröße;Einheit;Anzahl Messwerte;Minimum;Mittelwert;Maximum
Station 1;co2;CO₂;ppm;54373;407,000;497,818;707,000
Station 1;ws3;Luftfeuchtigkeit;%;46420;24,000;62,082;96,000
Station 1;ws3;Luftdruck;hPa;46420;950,500;954,904;958,800
Station 1;ws3;Temperatur;°C;46420;15,000;22,835;36,111

Hier noch die Analyse in einer Grafik:



Datenformat und Architektur

Zum besseren Verständnis hier ein Überblick über den Aufbau der Klimastation:
Detailierte Information: **https://pdp11gy.com/Klimastation.html**



Der Aufbau besteht derzeit aus **3** Stationen: **Station-1**, **Station-2** und eine spezielle **Strahlenstation**. Die sehr preisgünstige Hardware Architektur basiert auf Raspberry PI PICO W für die Übertragung der Sensordaten und jeweils ein Raspberry PI Zero W als Server System.

Die Übertragung der Sensordaten erfolgt im ~15 Sekunden Rytmus zum Serversystem und wird pro Tag im daten.json Format gesichert mit folgenden Frams:

{"temp": 13, "status": 0, "station": "station1", "sensor": "co2", "co2": 484, "timestamp": "2026-09-20 00:02:20"}
{"usvh": 0.195, "station": "station1u2", "cpm": 24, "uv_raw": 1, "temperature": 23.4, "pressure": 964, "sensor": "geiger", "timestamp": "2026-09-20 00:02:22"}
{"wind_speed": 0.0, "station": "station1", "sensor": "ws3", "pressure": 961.4, "wind_gust": 0.0, "rain_rate": 0, "rain_total": 0, "humidity": 76, "temperature": 14.444445, "timestamp": "2026-09-20 00:02:23"}
{"sensor": "Feinstaub", "pm25": 2.2, "pm10": 4.4, "station": "station2", "timestamp": "2026-09-20 00:02:23"}
{"sensor": "Feinstaub", "pm25": 2.2, "pm10": 4.6, "station": "station1", "timestamp": "2026-09-20 00:02:24"}
{"humidity": 38.7, "pressure": 963.1, "station": "station2", "sensor": "bme680", "temperature": 22.5, "timestamp": "2026-09-20 00:02:25"}
{"temp": 18, "status": 0, "station": "station2", "sensor": "co2", "co2": 565, "timestamp": "2026-09-20 00:02:29"}

Die Abbilding in der Sofware sieht dann so aus:

```
STATION_NAMEN = {
    "station1": "Station 1",
    "station2": "Station 2",
    "station1u2": "Strahlungsstation",
}

FELD_INFO = {
    "co2": ("CO₂", "ppm"),
    "pm25": ("PM2.5", "µg/m³"),
    "pm10": ("PM10", "µg/m³"),
    "temperature": ("Temperatur", "°C"),
    "temp": ("Temperatur (Sensor)", "°C"),
    "humidity": ("Luftfeuchtigkeit", "%"),
    "pressure": ("Luftdruck", "hPa"),
    "wind_speed": ("Windgeschwindigkeit", ""),
    "wind_gust": ("Windböen", ""),
    "rain_rate": ("Niederschlagsrate", ""),
    "rain_total": ("Niederschlag gesamt", ""),
    "usvh": ("Strahlungsdosisleistung", "µSv/h"),
    "cpm": ("Geiger-Zählrate", "CPM"),
    "uv_raw": ("UV-Rohwert", ""),
}
```

Die Architektur ist sehr flexibel gestaltet und kann somit beliebig erweitert oder angepasst werden.

Mitwirkende und Zusammenarbeit

Projektidee, Hardware und Datenerfassung

Das Projekt *Environmental and Climate Data Station* wurde von Reinhard Heuberger konzipiert und realisiert. Die Projektidee, die Auswahl und Integration der Sensorhardware, die Programmierung der Datenerfassung sowie der Aufbau und Betrieb der Klimastationen liegen in seiner Verantwortung.

Ziel ist es, mit kostengünstiger Hardware langfristig Umwelt- und Klimadaten zu sammeln und diese für interessierte Anwender verständlich auszuwerten.

Unterstützung durch OpenAI ChatGPT

Bei der Entwicklung der Python-basierten Auswertungssoftware wurde OpenAI ChatGPT als KI-gestütztes Hilfsmittel eingesetzt. Die Zusammenarbeit umfasste insbesondere die Programmstruktur, die grafische Benutzeroberfläche mit Tkinter, die Verarbeitung der JSON-Messdaten, die Darstellung mit Matplotlib sowie die Entwicklung und Fehlersuche bei der Kalender- und Ordnerauswahl.

Weitere Unterstützung erfolgte bei der Konfiguration über `config.ini`, der automatischen Vergabe von Versionsnummern für gespeicherte Diagramme und Statistiken sowie bei der Implementierung der wöchentlichen Datenauswertung.

Gemeinsame Entwicklung und Verantwortung

Die Auswertungssoftware entstand in einer schrittweisen Zusammenarbeit: Reinhard Heuberger brachte die Anforderungen aus dem praktischen Betrieb seiner Klimastationen ein, prüfte die vorgeschlagenen Änderungen in seiner eigenen Umgebung und gab Rückmeldungen zu Funktion und Bedienbarkeit. ChatGPT unterstützte daraufhin bei der Programmierung, Überarbeitung und Fehlerbehebung.

Die endgültige Auswahl, Integration und praktische Prüfung der Software erfolgte durch Reinhard Heuberger. Die Unterstützung durch ChatGPT wird hier ausdrücklich genannt, um den Einsatz KI-gestützter Werkzeuge bei der Softwareentwicklung transparent zu machen.

Projekt und Datenerfassung: Reinhard Heuberger
KI-gestützte Entwicklungsunterstützung: OpenAI ChatGPT

Ausblick

Die Entwicklung der Auswertungssoftware ist ein fortlaufender Prozess. Erfahrungen aus dem praktischen Betrieb, neue Anforderungen und Rückmeldungen von Anwendern können in zukünftige Verbesserungen einfließen. Ziel ist es, die Software verständlich, zuverlässig und für weitere Interessierte nutzbar zu gestalten.

Das Projekt zeigt, wie kostengünstige Sensorhardware, eigene praktische Erfahrungen und KI-gestützte Programmierunterstützung zusammenwirken können. Die Weiterentwicklung erfolgt schrittweise durch Reinhard Heuberger, unterstützt durch OpenAI ChatGPT bei Programmierung, Analyse und Fehlerbehebung.

Für Anregungen und Hinweise:
Mail an **info@pdp11gy.com**