IHK-Dokumentation

1 Projektauftrag und Zielsetzung

1.1 Projekthintergrund

Im Rahmen des Berufsschulunterrichts der Klasse FA 11.1 wurde die Entwicklung einer Wetter-App initiiert. Ziel war es, eine praxisnahe Anwendung zu erstellen, die aktuelle Wetterdaten mittels der OpenMeteo API abruft und visualisiert.

1.2 Projektziel

Entwurf und Implementierung einer plattformunabhängigen GUI zur Anzeige von Temperatur-, Wind- und Niederschlagsdaten
Nutzung von Python 3.13 und moderner Bibliotheken (Streamlit, Plotly)
Sicherstellung hoher Wartbarkeit und Erweiterbarkeit

1.3 Abgrenzung

Kein Hardware-Anschluss (Arduino)
Keine Cloud-Deployments
Betriebssystem: Linux (Pop!_OS, Ubuntu 22.04 LTS, Debian) und Windows

2 Projektmanagement

2.1 Zeitplan und Aufwand

Phase	Geplante Dauer	Tatsächliche Dauer
Anforderungsanalyse	2 Stunden	1 Stunde
Design und Fachkonzept	2 Stunden	3 Stunden
Implementierung Basis	6 Stunden	30 Stunden
Test und Dokumentation	2 Stunden	5 Stunden
Gesamt	12 Stunden	39 Stunden

2.2 Rollen und Organisation

Projektleitung, Architekturdesign, Implementierung und Test wurden von einer Person (Projektverantwortlicher) übernommen.

3 Lasten- und Pflichtenheft

3.1 Lastenheft

Auswahl von Land, Stadt und Postleitzahl
Anzeige von Tages- und Stundenwerten
Export der Wetterdaten als CSV

3.2 Pflichtenheft

Nutzung der OpenMeteo API zur Datenbeschaffung
Georeferenzierung via Geopy/Nominatim
Cache-Strategie und Retry-Logik bei API-Fehlern

4 Fachkonzept

4.1 Anwendungsfälle (Use Cases)

Ort eingeben – Nutzer gibt Land, Stadt, PLZ ein
Zeitraum wählen – Auswahl: tages- oder stundenweise
Daten abrufen – Klick auf „Go“
Anzeige – Auswahl zwischen Summary View und Detailed Analysis
CSV-Export – Download der angezeigten Daten

4.2 Klassendiagramm

Location
GeoLocationClient
OpenMeteoService (Implementierung von WeatherClientInterface)
WeatherData

5 Technisches Konzept

5.1 Architekturübersicht

UI-Schicht: Streamlit Webinterface
Service-Schicht: OpenMeteoService
Client-Schicht: GeoLocationClient, OpenMeteoClient
Domain: Modelle (Location, Coordinates, WeatherData)

5.2 API-Integration

5.2.1 Geopy/Nominatim (geopy_api.py)

```
class GeoLocationClient:
        def __init__(self):
        self.geo_client = Nominatim(user_agent=agent_name)
        def get_coordinates(self, location: Location) -> Coordinates:
        address = ", ".join([location.city, location.postal_code, location.country])
        response = self.geo_client.geocode(address)
        if not response:
        raise ValueError(f"Location not found: {address}")
        return Coordinates(latitude=response.latitude, longitude=response.longitude)
```

5.2.2 OpenMeteo API (open_meteo_api.py)

```
class OpenMeteoClient:
        def __init__(self, cache: CacheStrategy):
        self.client = openmeteo_requests.Client(session=cache.retry_session)
        def get_weather(self, url, params: dict):
        return self.client.weather_api(url, params)[0]
```

5.2.3 Service-Logik (open_meteo_service.py)

Parameteraufbau (_build_parameter)
Antwortverarbeitung (_handle_response, _hourly_data_handling, _daily_data_handling)
Konvertierung in pandas.DataFrame

6 Implementierung

6.1 Bibliotheken

numpy, pandas, plotly, streamlit
requests-cache, retry-requests
geopy, openmeteo-requests

6.2 Installation

ZIP-Datei entpacken
pip install -r requirements.txt
PYTHONPATH=src streamlit run src/application/ui/webapp_ui.py

7 Bedienungsanleitung (Integriert)

7.1 Eingabefelder (linke Seite)

Country, Postal Code, City, Time Interval (Days/Hours), Time Span, Go-Button, Download CSV

7.2 Anzeige (rechte Seite)

Summary View / Day: Tageswerte (Temperatur, gefühlte Temperatur, Wind)
Detailed Analysis / Day: Tabellenansicht (Temperatur, Niederschlag, Wind, Sicht, Wolken, Böen)
Detailed Analysis / Hour: Stündliche Daten für alle Parameter

7.3 FAQ

Standardwerte in settings.py ändern
Diagrammzomm: Auswahl mit Maus, Doppelklick zurück
Download-Button erscheint nach Datenabruf
PNG-Export via Kamerasymbol im Plot

8 Test und Qualitätssicherung

8.1 Funktionstests

Abruf unterschiedlicher Orte und Zeiträume
Vergleich gegen OpenMeteo-Weboberfläche

8.2 Kompatibilität

Browser: Firefox, Edge, Chrome unter Linux und Windows

9 Projektreflexion

Ursprüngliches Arduino-Hardwarekonzept verworfen
Wechsel von Tkinter zu Streamlit für schnellere Umsetzung
Erweiterung der Anforderungen und Mehraufwand (30h statt 10h)