1. Feladat - Adatgyűjtés és Adatelőkészítés

1. Válasszon egy nyilvánosan elérhető API-t vagy Open Datasetet. Például:
    - Időjárás API: OpenWeatherMap, AccuWeather
    - Tőzsdei árfolyam API: Exchangeratesapi.io
    - Kaggle, data.gov

2. Python segítségével gyűjtsön be adatokat az API-ból legalább 3 különböző időpontból vagy paraméterrel (pl. különböző városok időjárása, különböző termékek árfolyama stb.).

3. Mentse el a nyers adatokat Parquet vagy CSV formátumban.

4. Töltse be a mentett nyers adatokat Python (pandas) segítségével.

5. Végezzen el az adatokon legalább 3-4 olyan tisztítási és transzformációs lépést. Például:
    - Hiányzó értékek kezelése
    - Dátum formátum konvertálása
    - Redundáns oszlopok eltávolítása, új oszlopok létrehozása

6. Az előkészített adatokat mentse el egy tiszta Parquet vagy CSV fájlba

7. Készítsen egy egyszerű Tableau vagy Spotfire dashboardot (trial verzióval), amely az előkészített, tisztított adatokat jeleníti meg.
    - Használja a 6. pontban kimentett táblát adatforrásként
    - Készítsen legalább 2-3 különböző vizualizációt
    - A dashboard legyen interaktív

8. Készítsen egy rövid (max. 1 oldalas) leírást a feldolgozás lépéseiről, az adatforrásról és a tisztítási döntésekről

Alapkövetelmény:

- A Python script legyen moduláris
- Írjon rövid docstringeket vagy használjon type hint-et a főbb függvényekhez
- Használja a GIT-et a verziókövetésre (a megoldásokat egy GIT repository-ban készítse el, amelyet osszon meg velünk)

Opcionális:

- A Python script tartalmazzon alap logging beállítást. Legalább három log üzenet jelenjen meg a folyamat kulcslépéseinél (pl. adatlekérés, fájlmentés, adatátalakítás kezdete/vége)
- Legalább 1–2 rövid unit tesztet készítsen (pl. Pytest-tel), amelyek a fő függvények (pl. adat-transzformáció, tisztítás) működését ellenőrzik.

2. Feladat - PySpark

1. Generáljon egy nagy méretű (100 ezer – 1 millió soros) adatállományt, amely szenzoradatokat imitál. Az alábbi oszlopokat hozza létre:
    - Sensor_ID (pl.: ‘a323frt’, ‘df21fs1’…)
    - Date (pl.: 2025.10.20, 2025.10.09)
    - Location (pl.: Hungary, Germany…)
    - Parameter (Offset, Noise, Temperatur ,…)
    - Value (numerikus értékek)
    - Status (kategorikus értékek: Good, Bad)

2. Írjon egy PySpark scriptet, amely betölti a generált adatokat.
    - Végezzen el legalább 3 komplex transzformációt. Például:
        - Window függvények használata
        - GroupBy + aggregáció több mezőre
        - Pivot/unpivot
        - String vagy dátum manipuláció

3. Mentse el a feldolgozott adatokat (Parquet) formátumban. Úgy alakítsa ki a táblát, hogy az adattípusok illeszkedjenek a PySpark DataFrame mezőihez.

4. Válasszon egy egyszerű adatbázist (pl. SQLite, PostgreSQL Docker konténer), vagy amivel könnyedén tud dolgozni:
    - Hozzon létre egy táblát az adatbázisban, amelybe az előzőleg PySpark-al feldolgozott adatokat be tudja tölteni (a legenerált adatokat)
    - Írjon egy Python scriptet, amely PySpark-ból kiolvassa a feldolgozott adatokat és betölti az adatbázisba
    - Írjon legalább 3 SQL lekérdezést az adatbázishoz, amelyekkel ellenőrzi az adatok integritását és a feldolgozás helyességét

Alapkövetelmény:

- A Python script legyen moduláris
- Írjon rövid docstringeket vagy használjon type hint-et a főbb függvényekhez
- Használja a GIT-et a verziókövetésre (a megoldásokat egy GIT repository-ban készítse el, amelyet osszon meg velünk)

3. Feladat – Frontend és Backend

Készítsen egy egyszerű webes felületet, amely megjeleníti a második feladatban feldolgozott szenzoradatokból származó aggregált információkat.

1. Készítsen egy egyszerű Python alapú REST API-t (pl. Flask vagy FastAPI segítségével), amely a második feladatban betöltött adatbázisból tud adatokat szolgáltatni.
    - Az API-nak legalább 2-3 végpontja legyen:
        i. Egy végpont, ami visszaadja a „Sensor_ID”-k listáját.
        ii. Egy végpont, ami egy adott „Sensor_ID”-hez tartozó összes feldolgozott adatot visszaadja (vagy egy összefoglaló statisztikát, pl. átlagos érték).
    - Használjon pydantic modellt (FastAPI esetén) vagy marshmallow-t (Flask esetén) az adatok validálására.
    - Legyen alap hiba és státuszkód kezelés

2. Készítsen egy egyszerű HTML oldal, amely a CSS-t és Javascriptet használ
    - A Javascript segítségével hívja meg a backend API-t, és jelenítse meg az adatokat a weboldalon. Például:
        i. Egy legördülő lista a „Sensor_ID”-ről és a kiválasztás után jelenjen meg az adott szenzor adatai
        ii. Egy táblázat, amelyben az adatok rendezhetőek vagy szűrhetőek

3. Készítsen egy README.md fájlt, amely tartalmazza
    - A pipeline fő lépéseit (adatgyűjtés → feldolgozás → betöltés → API → frontend)
    - A futtatás lépéseit (Docker / manuálisan)