

Weekly Project

Mini Data Engineering Pipeline – API & SQL Focus

מטרת הפרויקט

בפרויקט זה תבנו מערכת מבוססת שירותים (Service-Based Architecture) שמדמה Data Pipeline בעולם האמיתי.

הדגש הוא על:

- תכנון נכון של שירותים
 - זרימת דאטה ברורה
 - שימוש מושכל ב-pandas וב-SQL
 - תרגום שאלות עסקיות פשוטות לנקודות קצה
-

מערכת – Tactical Weather Intelligence

יחידת **רקיע** פועלת כיחידת תכנון ותצפית מבצעית, האחראית על הבנת המרחב האווירי והסביבתי שבו מתקיימת הפעילות.

כשם שהרקיע משקף את תנאי הסביבה הרחבים – רוחות, עננות ושינויים אקלימיים – כך תפקידה של היחידה הוא לספק מבט עליון, יציב וברור על תנאי השטח.

היחידה אינה עוסקת בלחימה ישירה, אלא ב-**איסוף מודיעין סביבתי** המסייע בקבלת החלטות טקטיות: תזמון תנועה, הערכת סיכונים סביבתיים והתאמת פעילות לתנאי מזג האוויר בפועל. גם שינויים קטנים ברוח או בטמפרטורה עשויים להשפיע על דיוק, יציבות ויכולת תמרון של אמצעים מבצעיים.

המערכת שאתם מפתחים משמשת כ-**תשתית נתונים מרכזית** עבור יחידת רקיע: היא מרכזת נתוני מזג אוויר גולמיים, מנקה ומארגנת אותם, שומרת אותם בצורה מסודרת במסד נתונים, ומאפשרת שאלות בסיסיות המספקות תמונת מצב סביבתית אמינה ונגישה.

מבט כללי על תהליך העבודה עם נתונים

לפני שנצלול לפרטים הטכניים, חשוב להבין את **שלבי היסוד בעבודה עם נתונים**, כפי שהם מתבצעים כמעט בכל מערכת Data Engineering בעולם האמיתי.

- **Data Ingestion** – זהו שלב האיסוף וקליטה. כאן אנחנו מביאים נתונים ממקור חיצוני אל תוך המערכת שלנו, כמו שהם. הנתונים עדיין גולמיים, לא מסודרים, ולא בהכרח מתאימים לשימוש.
- **Data Cleaning / Transformation** – זהו שלב הסידור וההתאמה. בשלב זה אנו מתקנים בעיות בסיסיות בנתונים: פורמטים, טיפוסים, ערכים חסרים או כפולים. המטרה אינה לנתח את הנתונים, אלא להפוך אותם ל-**נקיים, עקביים וברורים**.
- **Data Preparation** – זהו שלב ההכנה לאחסון ולשימוש עתידי. כאן אנחנו מבטיחים שהנתונים בנויים בצורה שמתאימה למסד נתונים טבלאי ולשאלות פשוטות, כך שניתן יהיה לשאול עליהם שאלות בצורה אמינה.

בפרויקט זה, כל שלב מיושם בשירות נפרד, כדי שתוכלו להבין מה **קורה לדאטה בכל רגע** ומדוע כל שלב חשוב בפני עצמו.

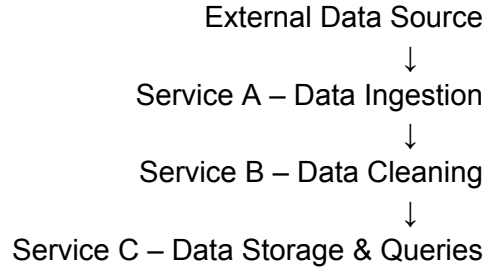
מקור הנתונים החיצוני — Open-Meteo Weather API

בפרויקט זה נשתמש בשירות מזג אוויר חיצוני בשם **Open-Meteo Weather API** — ממשק HTTP שמספק נתוני מזג אוויר אמיתיים עבור מיקומים בכל רחבי העולם. ה-API מאפשר לבקש מידע כמו טמפרטורה, מהירות רוח ולחות בפורמט **JSON**, ללא צורך במפתח API או הרשמה מיוחדת. הנתונים זמינים במסגרת שירות ציבורי חופשי ונגיש, ומתאימים לשימוש במערכי צד-שרת כמו זה שאנחנו בונים בפרויקט זה.

תיעוד רשמי: <https://open-meteo.com/en/docs>

השליפה מה-API תעשה בשירות ה-Ingestion שלנו, ומה שמתקבל ישמש כבסיס ל-Data Cleaning ו-Data Preparation לפני שמירה ב-MySQL ולבסוף לשאלות SQL המבקשות לענות על שאלות סביבתיות פשוטות בהמשך הפרויקט.

מבנה המערכת (High Level)



כל שירות:

- עצמאי
- בעל תחום אחריות ברור
- מתקשר עם שירותים אחרים דרך HTTP בלבד

Service A – Data Ingestion

תפקיד השירות

אחראי על הבאת נתונים משירות חיצוני והעברתם למערכת.

נקודות קצה

POST /ingest

מימוש שלב ה-Ingestion

בשלב זה תיישמו מנגנון Ingestion המבוסס על **רשימת מיקומים**. המערכת תרוץ בלולאה על רשימת מיקומים מוגדרת מראש, ותבצע עבור כל מיקום שתי קריאות חיצוניות:

1. תרגום שם המיקום לקואורדינטות (Geocoding)

2. שליפת נתוני מזג אוויר שעתיים עבור הקואורדינטות

תוצאות כל הקריאות יאספו ל-**רשימה אחת במבנה אחיד**, ויוחזרו כקלט לשלב העיבוד הבא, ללא ניקוי, חישוב או שינוי מבנה.

לצורך מיקוד והפחתת מורכבות, יסופקו לכם **שתי דוגמאות קוד** לקריאות ה-Geocoding (Weather API ו-), אשר ישמשו כנקודת התחלה בלבד. האחריות לשילוב הלוגיקה, ניהול הלולאה והחזרת הפלט הסופי — היא שלכם.

דגשים:

- אין ניקוי
- אין חישובים

Service B – Data Cleaning & (Normalization (pandas

תפקיד השירות

להכין נתונים גולמיים לאחסון במסד נתונים טבלאי.

נקודת קצה

POST /clean

תיאור:

נקודת קצה זו אחראית על קבלת נתונים גולמיים כפי שהתקבלו משלב ה-Ingestion, והפיכתם לנתונים **נקיים, אחידים ובעלי מבנה טבלאי ברור**. בשלב זה הנתונים עוברים מעבר מ־JSON מורכב ורב־מיקומי לייצוג שטוח, עקבי וקל לעבודה, המותאם לשמירה במסד נתונים ולשאלות SQL בהמשך.

העבודה בשלב זה כוללת:

- מעבר על רשימת הנתונים שהתקבלה
- יצירת **DataFrame** אחד אחיד
- טיפול בבעיות בסיסיות בדאטה הכנה להעברה לשירות האחסון

בשלב זה, לאחר ניקוי ונרמול הנתונים, עליכם להוסיף ל-DataFrame שתי עמודות נגזרות פשוטות, שמטרתן לתאר את תנאי מזג האוויר בצורה מילולית וברורה, ללא חישובים מורכבים.

עמודת מצב טמפרטורה (`temperature_category`)

עמודה זו מסווגת את מצב הטמפרטורה על פי ערך הטמפרטורה במעלות צלזיוס, בהתאם לכללים הבאים:

- טמפרטורה מעל `hot` $25^{\circ}\text{C} \rightarrow$
- טמפרטורה בין 18°C ל- 25°C (כולל) \rightarrow `moderate`
- טמפרטורה מתחת ל-`cold` $18^{\circ}\text{C} \rightarrow$

הסיווג נועד לייצג תיאור כללי של תנאי הסביבה, ולא מדד מדעי או תחזית.

עמודת מצב רוח (`wind_status`)

עמודה זו מתארת את עוצמת הרוח בצורה פשוטה, על בסיס מהירות הרוח:

- מהירות רוח מעל 10 \rightarrow `windy`
- מהירות רוח עד 10 \rightarrow `calm`

(יחידות המידה הן כפי שהתקבלו מה-API, ללא המרה נוספת.)

הערה חשובה (רמז מכוון):

בשלב זה תעבדו עם **DataFrame**, אך שימו לב: לא ניתן להעביר DataFrame "כמו שהוא" בין שירותים דרך HTTP.

חלק מהעבודה שלכם הוא **לחקור ולבחור דרך מתאימה** לייצג את הנתונים לאחר הניקוי, כך שניתן יהיה לשלוח אותם לשירות הבא בפורמט סטנדרטי ומקובל (לדוגמה: JSON).

זהו חלק מכוון בתרגיל, שמדמה בעיה אמיתית בעולם ה-Data Engineering.

Service C – Data Storage & Light Analytics

((SQL

תפקיד השירות

אחסון הנתונים ושאלות בסיסיות מעליהם.

נקודות קצה – אחסון

POST /records

בשלב האחסון (Service C) הנתונים שמתקבלים משירות העיבוד נשמרים במסד הנתונים MySQL בצורה טבלאית, אחידה ושטוחה, כך שכל שורה מייצגת מדידת מזג אוויר אחת עבור מיקום אחד בנקודת זמן אחת, ללא חישובים או שינוי לוגי נוסף, ומוכנים לשאלות SQL פשוטות.

מבנה הטבלה (weather_records):

- id – INT, PRIMARY KEY, AUTO_INCREMENT •
- timestamp – DATETIME •
- location_name – VARCHAR •
- country – VARCHAR •
- latitude – FLOAT •
- longitude – FLOAT •
- temperature – FLOAT •
- wind_speed – FLOAT •
- humidity – INT •
- temperature_category – VARCHAR •
- wind_category – VARCHAR •

נקודות קצה – שאליות (Light Analytics)

כל נקודת קצה עונה על שאלה אחת פשוטה בלבד

GET /records

תיאור:

שליפת רשומות גולמיות עם אפשרות סינון.

שאלה עסקית:

אילו נתונים נאספו בפרק זמן מסוים או באזור מסוים?

GET /records/count

תיאור:

החזרת מספר הרשומות לפי אזור.

שאלה עסקית:

מאיזה אזורים נאסף הכי הרבה מידע?

GET /records/avg-temperature

תיאור:

החזרת טמפרטורה ממוצעת לפי אזור (עבור סט הנתונים שיש לנו בנקודת הזמן הנוכחית במערכת).

שאלה עסקית:

מה הטמפרטורה הממוצעת שנמדדה בכל אזור?

GET /records/max-wind

תיאור:

החזרת מהירות רוח מקסימלית לפי אזור (עבור סט הנתונים שיש לנו בנקודת הזמן הנוכחית במערכת).

שאלה עסקית:

מה הייתה מהירות הרוח הגבוהה ביותר בכל אזור?

GET /records/extreme

תיאור:

החזרת מיקומים לפי סיווג של תנאים מאתגרים מבחינה מבצעית, על פי השילוב הנפוץ יותר:

- חם ורוח חלשה (hot & calm)
- קר ורוח חזקה (cold & windy)

שאלה עסקית:

באילו מקומות נמדדו תנאים חריגים שיהיו מאתגרים מבחינה מבצעית?
