

Weekly Project

Mini Data Engineering Pipeline – API & SQL Focus

מטרת הפרויקט

בפרויקט זה我们将 בנו מערכת מבוססת שירותים (Service-Based Architecture) שمدמה Data Pipeline בעולם האמיתי.

הדגש הוא על:

- תכנון נכון של שירותיים
- זרימת>Dataflow בדומה
- שימוש מושכל ב-pandas ו-`SQL`
- תרגום **שאלות עסקיות פשוטות** לנקודות קצה

Tactical Weather Intelligence

יחידת רקיע פועלת כיחידה תכנון ותפעול מבצעית, האחראית על הבנת המרחב האויריאנט והסבירתי שבו מתקיימות הפעולות.

כשם שהרקייע משקף את תנאי הסביבה הרחבים — רוחות, עננות ושינויים אקלימיים — כך תפקידה של היחידה הוא לספק מידע עלין, יציב וברור על תנאי השטח.

היחידה אינה עוסקת בלבד בחינה ישירה, אלא **ב-איסוף מידע סביבתי** המשמש בקבלה החלטות טקטיות: תזמון תנועה, הרכבת סיכונים סביבתיים והתאמת פעילות לתנאי מג האויר בפועל. גם שינויים קטנים ברוח או בטמפרטורה עשויים להשפיע על דיווק, יציבות ויכולת תמרן של אמצעים מבצעיים.

המערכת שאותם מפתחים משמשת כ**תשתיית נתונים מרכזית** עבור יחידת רקיע: היא מרכזת נתונים מג אויר גלמיים, מנקה ומארגנת אותם, שמורת אותם בצורה מסודרת בסיס נתונים, ומאפשרת שאלות בסיסיות המספקות תמונה מצב סביבתית אמינה ונגישה.

מבט כללי על תהליכי העבודה עם נתונים

לפנינו שנצלו לפרטים הטעניים, 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 מביבאים נתונים ממוקב חיצוני אל תוך המערכת שלנו, כמו שהם. הנתונים עדיין גולמיים, לא מסודרים, ולא בהכרח מתאימים לשימוש.
- **Data Cleaning / Transformation** – זהו שלב הסידור וההתאמאה.
בשלב זה אנו מתקנים בעיות בסיסיות בנתונים: פורמטים, טיפוסים, ערכים חסרים או כפולים. המטרה אינה לנתח את הנתונים, אלא להפוך אותם לנקיים, עקובים וברורים.
- **Data Preparation** – זהו שלב ההכנה לאחסון ולשימוש עתידי.
כאן אנחנו מבטיחים שהנתונים בניים בצורה שמתאימה למדן נתונים טבלאי ולשאילתות פשוטות, כך שניתן יהיה לשאול עליהם שאלות בצורה אמינה.

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

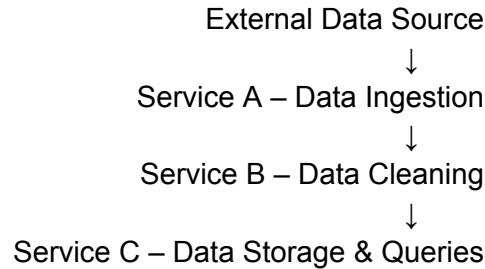
מקור הנתונים החיצוני — Open-Meteo Weather API

בפרויקט זה נשתמש בשירות מגז אוויר חיצוני בשם **Open-Meteo Weather API** — ממשק HTTP שמספק נתונים מגז אוויר אמיתיים עבור מקומות בכל רחבי העולם. ה-API מאפשר לבקש מידע כמו טמפרטורה, מהירות רוח ולחות בפורמט **JSON**, ללא צורך בפתח API או הרשמה מיוחדת. הנתונים זמינים במסגרת שירות ציבורי חופשי ונגיש, ומתאימים לשימוש במערכות צד-שרת כמו זה שאנו עושים בפרויקט זה.

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

השליפה מה-API תעשה בשירותי **Ingestion** שלנו, ומה שמתקיים ישמש כבסיס ל-**Data Cleaning**-ו-**Data Preparation** לפני שמירה ב-MySQL ולבסוף לשאלות SQL המבוקשות לענות על שאלות סביבתיות פשוטות בהמשך הפרויקט.

מבנה המערכת (High Level)



כל שירות:

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

Service A – Data Ingestion

תפקיד השירות

אחראי על הבאת נתונים משירותים חיצוניים והעברתם למערכת.

נקודות קצה

POST /ingest

מימוש שלב ה-ingestion

בשלב זה תישמו מנגנון Ingestion המבוסס על רשימת מיקומים.
המערכת תרוץ בוללה על רשימת מיקומים מוגדרת מראש, ותבצע עבור כל מיקום שתי קריאות חיצונית:

1. תרגום שם המיקום לקואורדינטות (Geocoding)
2. שילוף נתונים מג אויר שעתיים עבור הקואורדינטות

תוצאות כל הקריאות יאספו לרשימה אחת **במבנה אחד**, וווחזו כקלט לשלב העיבוד הבא, ללא ניקוי, חישוב או שינוי מבנה.

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

dagshim:

- אין ניקוי
- אין חישובים

Service B – Data Cleaning & (Normalization (pandas

תפקיד השירות

להcin נתונים גולמיים לאחסן במסד נתונים טבלי.

נקודות קצה

POST /clean

תיאור:

נקודות קצה זו אחראית על קבלת נתונים גולמיים כפי שהתקבלו משלב ה-ingestion, והפיכתם לנתונים נקיים, אחידים ובעל מבנה טבלי ברור. בשלב זה הנתונים עוברים מעבר מ-JSON מורכב ורב-מיקומי לייצוג שטוח, עקבי וקל לעובדה, המותאם לשימירה במסד נתונים ולשאילתות SQL בהמשך.

העבודה בשלב זה כוללת:

- מעבר על רשיימת הנתונים שהתקבלו
- ייצור DataFrame אחד אחד
- טיפול בעוות בסיסיות בDATA הינה להעברה לשירות האחסון

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

עמודת מצב טמפרטורה (`temperature_category`)

עמודה זו מסוגת את מצב הטמפרטורה על פי ערך הטמפרטורה במלואו צלזיוס, בהתאם לכללים הבאים:

- **טמפרטורה מעל 25°C** → `hot`
- **טמפרטורה בין 18°C ל- 25°C (כולל)** → `moderate`
- **טמפרטורה מתחת ל- 18°C** → `cold`

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

עמודת מצב רוח (`wind_status`)

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

- **מהירות רוח מעל 10** → `windy`
- **מהירות רוח עד 10** → `calm`

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

הערה חשובה (رمز מכון):

בשלב זה תעבדו עם **DataFrame**, אך שימוש לבן ניתן להעביר DataFrame "כמו שהוא" בין שירותים דרך HTTP.

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

זהו חלק מכון בתרג'il, שמדמה בעיה אמיתית בעולם ה-Data Engineering.

Service C – Data Storage & Light Analytics ((SQL

תפקיד השירות

אחסון הנתונים ושאלות בסיסיות מעלייהם.

נקודות קצה – אחסון

POST /records

בשלב האחסון (Service C) הנתונים שמתקבלים משירות העיבוד נשמרים במד הנתונים MySQL בצורה טבלאית, אחידה ושטוחה, כך שכל שורה מייצגת מידע מג אויר אחד עבור מקום אחד בנקודת זמן אחת, ללא חישובים או שינוי לוגי נוספת, ומוכנים לשאלות SQL פשוטות.

מבנה הטבלה (weather_records)

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

נקודות קצה – שאלות (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:

מיהי אזורים נאסף היכי הרבה מידע?

GET /records/avg-temperature

תיאור:

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

שאלה עסקית:

מה הייתה מהירות הרוח הגבוהה ביותר בכל אזור?

GET /records/extreme

תיאור:

החזרת מקומות מסווגים של תנאים מתאימים מבחינה מבצעית, על פי השימוש הנפוץ יותר:

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

שאלה עסקית:

באילו מקומות נמדד תנאים חריגיים שהיו מתאימים מבחינה מבצעית?
