**Project Steps**

**Step 1 – Find data sources**  
Choose any dataset you want to analyze. These initial files **cannot be Parquet**; they must come in another format such as JSON or CSV.

**Step 2 – Partition the data**  
Split the dataset into *N* parts. For the demo, use **N = 4**.

**Step 3 – Upload raw data to S3**  
Create a bucket (self-hosted or AWS S3) with a folder called /raw.

* The bucket acts as your file repository and can be accessed through an API.
* Use a **Python script** that calls the bucket’s API to upload the JSON/CSV files into folder /raw.

**Step 4 – Convert raw data to Parquet via Lambda**  
Create another folder /parquet in the same bucket.

* Write a **Python AWS Lambda function** that is triggered whenever new files are uploaded to /raw.
* The Lambda should read the incoming JSON/CSV, convert it to **Parquet**, and save the converted file into folder /parquet in the bucket.

**Step 5 – Query the Parquet data with DuckDB**  
Now that you have Parquet files stored in the bucket, use **DuckDB** to read them directly from the bucket S3 and run SQL queries for your analysis. To know that this works use the DuckDB Client before going to the next step.

**Step 6 – Visualize with Grafana**  
Set up **Grafana** to connect directly to DuckDB.

* DuckDB will fetch the data from your S3 bucket.
* Grafana will let you build dashboards, visualize queries, and interact with the analysis results.

An end-to-end pipeline should look like this:  
Data source → S3 (/raw) → Lambda → S3 (/parquet) → DuckDB → Grafana.

**Deliverables**

The final deliverables of the project will include:

1. **GitHub Repository**
   * All source code (upload scripts, Lambda function, DuckDB queries, Grafana configuration).
   * A **README.md** file with clear instructions on how to set up and run the project step by step.
2. **Project Documentation (PDF)**
   * Explanation of what was done and why.
   * Description of the analysis performed.
   * Tools and technologies used for each part, and how they were used.
   * Presentation of the results obtained.
3. **Presentation & Demo**
   * A **PowerPoint presentation** for the oral defense.
   * A **live demo** showing the pipeline working end-to-end (upload → Lambda conversion → DuckDB queries → Grafana visualization).