Coal Mining Analytics

1. Project Objective

This project is a complete, end-to-end data platform designed to ingest, process, and analyze production data for a coal mining operation.

---

2. System Architecture

The platform follows a **ETL (Extract, Transform, Load)** architecture. Data is extracted from various sources, transformed in a dedicated processing layer, and loaded into a central data warehouse for unified analysis.

Architectural Flow:

1. **Extract:** A Python service extracts raw production data from a source MySQL database, CSV file from IOT sensors, and weather API.
2. **Transform:** The core transformation logic resides in a Python ETL service. This service cleans the raw data, identifies anomalies based on business rules
3. **Load:** The final, cleaned, and enriched data is loaded into a ClickHouse data warehouse,

---

3. Key Features

- **Automated ETL Pipeline:** A robust Python pipeline handles the extraction, transformation (including data cleaning and API enrichment), and loading of data.
- **Anomaly Detection:** The pipeline automatically flags anomalous production and sensor data, separating it into dedicated tables for further analysis.
- **Predictive Forecasting:** A machine learning model is trained on historical data to forecast daily production metrics, allowing for proactive decision-making.
- **Interactive Dashboards:** A comprehensive Superset dashboard provides insights into daily production, equipment performance, quality grades, and the correlation between weather and operational efficiency.
- **Fully Containerized:** The entire stack is defined in Docker Compose, allowing for a seamless, one-command setup.

---

4. Tech Stack

- **Orchestration:** Docker Compose
- **Source Database:** MySQL
- **Data Warehouse:** ClickHouse
- **ETL & ML:** Python (Polars)
- **Business Intelligence:** Power BI

---

5. Getting Started

Prerequisites

- Docker & Docker Compose
- A SQL client compatible with ClickHouse (e.g., DBeaver, DataGrip)

Configuration

1. Clone this repository to your local machine.

How to Run

Step 1: Launch All Services

This command will build the custom images and start all services (database, warehouse, Superset, etc.) in the background.

Bash

```
docker compose up -d --build
```

Step 2: Initialize the Data Warehouse

The ClickHouse data warehouse must be initialized the first time you run the project. This creates the necessary database and tables.

Connect to the ClickHouse server using a SQL client with the following details:

- **Host:** localhost
- **Port:** 8123
- **User:** default

Open the /data/clickhouse/init.sql file from this project, copy its contents, and run the query in your client.

Or you can install clickhouse-client CLI and run this command

```
clickhouse-client -h localhost --multiquery --user default < ./data/clickhouse/init.sql
```

Step 3: Start the ETL Pipeline

Once the warehouse is ready, the ETL container will run automatically but there's a chance the clickhouse hasn't properly started yet. This will begin the process of moving and transforming data manually.

Bash

```
docker compose start etl-pipeline
```