Task 1: Choosing a suitable database and storing the data in batches

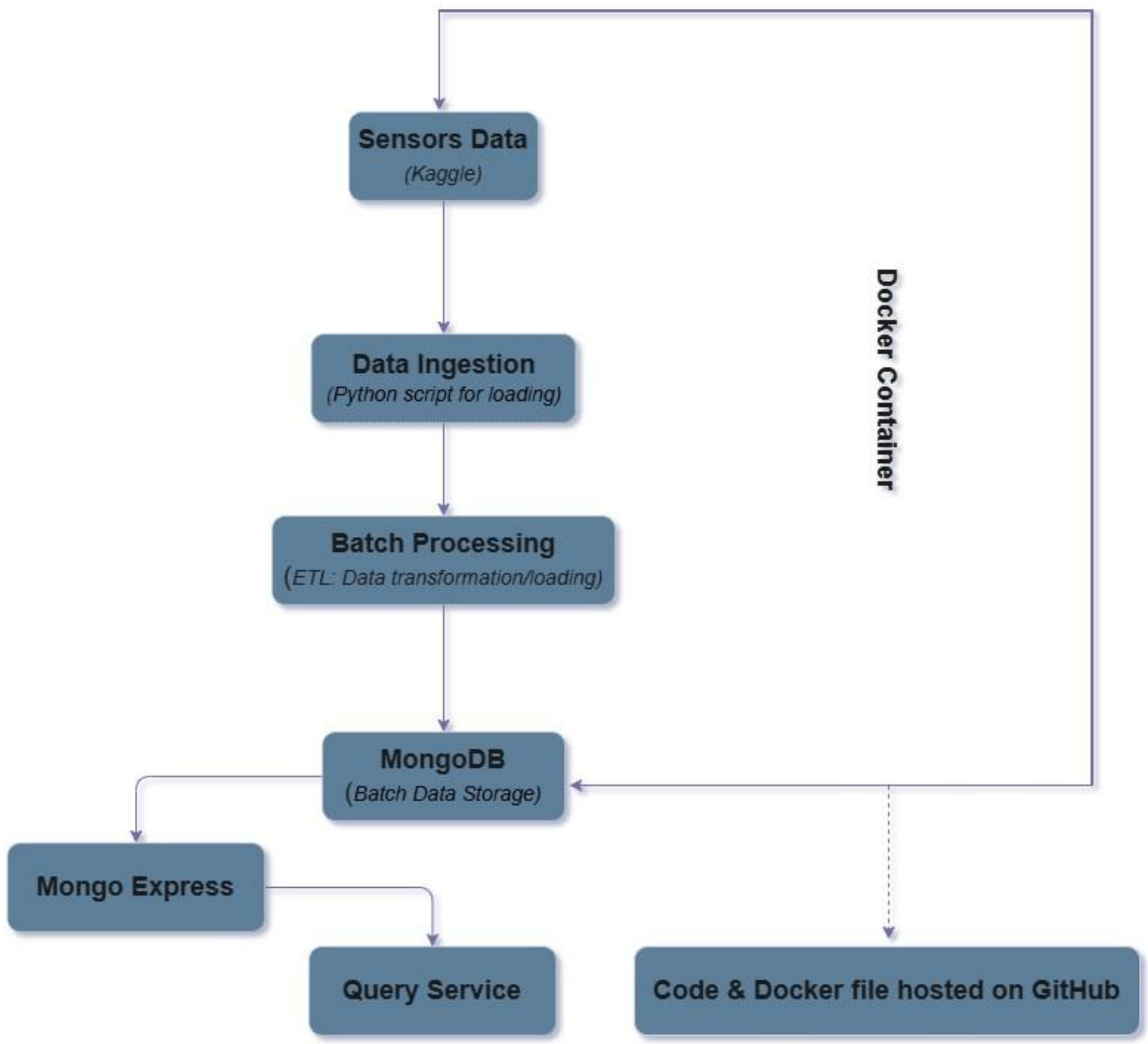
Submitted to: Pro. Dr. Sahar Qaadan

Submitted by: Dost Muhammad

Matriculation no: 92128703

Concept phase

Prototype representation:



Conceptual phase:

For this project, I selected a publicly available dataset from Kaggle titled *IoT Sensor Data*, which simulates environmental measurements such as temperature, humidity, smoke, gas, and light. This dataset is structured in CSV format with timestamped readings and will serve as the foundation for designing the data system. It fits the context of a municipality collecting sensor data to monitor urban environmental conditions.

The goal is to design a data system that can reliably store the collected sensor data in batches and provide flexible, scalable access to it for future analysis and applications. These applications might include citizen alerts and urban planning dashboards. The system must support dynamic data structures as the city plans to add more advanced sensors (e.g., CO_2, noise, fine dust) in the future.

To meet these requirements, I chose **MongoDB** as the database solution. MongoDB supports flexible document-based storage, allowing for evolving data schemas without significant redesign. It can be containerized with Docker for portability and is well-suited for horizontal scaling via cloud-based solutions, such as MongoDB Atlas. This ensures the system is reliable, easily maintainable, and future-proof.

Alternatives, such as relational databases (e.g., PostgreSQL or MySQL), were considered but are less suitable due to their rigid schemas. Time-series databases, such as InfluxDB, offer excellent support for temporal data, but introduce complexity in terms of setup and integration for generalized querying or document-based analysis.

The prototype implementation follows a modular design that separates ingestion, transformation, and storage for better maintainability and reusability. First, a Python script loads the sensor data from CSV format (data ingestion). Then, batch processing transforms and prepares the data (ETL). The processed data is stored in MongoDB. Mongo Express will be used for visual inspection of the database, and a basic query service will allow future integration. All components are containerized using Docker to ensure complete portability and ease of deployment. The source code and Docker setup will be hosted on GitHub.