Project Abstract and Reflection

Abstract

This project delivers a fully containerized, batch-processing data architecture designed to support machine learning workflows. The system ingests time-stamped stock price data through Apache NiFi, processes the data using Docker-managed microservices, and delivers processed files via a lightweight FastAPI interface. All services run locally within Docker containers, ensuring a modular, reproducible, and portable pipeline.
The architecture was built using Docker Compose and includes services for data ingestion (`nifi`), preprocessing (`spark` or shell-based NiFi processing), and delivery (`fastapi`). Data is ingested from local CSV files and passed through a NiFi flow which splits records and logs metadata. Processed data is then stored and made accessible via API.

Technical Reflection

Designing modular microservices enabled clear separation of responsibilities and easier troubleshooting. Dockerization ensured consistency across environments. Apache NiFi's visual interface simplified data flow orchestration, though care was needed to avoid common issues like dead-end processors or improper flowfile routing. FastAPI proved to be a powerful yet minimal tool for exposing data.
Testing validated that data could be reliably split, stored, and accessed via API. Error handling was built into the system using NiFi's retry and queue mechanisms. All system components communicated successfully, confirming the architectural soundness.

Personal Reflection

This project greatly enhanced my practical experience with batch data pipelines and containerized infrastructure. Troubleshooting container networking, flow misconfigurations, and NiFi errors helped develop problem-solving and debugging skills. I gained hands-on experience with DevOps practices such as container orchestration and service separation.
Moreover, presenting a reproducible pipeline with clean documentation, logging, and modularity helped me understand the production expectations in real-world data engineering environments.
Overall, this project not only met its technical objectives but also served as an excellent learning platform for applied data engineering.