Architecture & Data Flow

$$\begin{tikzcd}
\text{[IndexedDB (Browser Storage)]} \arrow[d, "\text{(JSON Fetch via REST / Ingest API)}"] \\
\text{[Node.js Ingest Server (Express + dotenv)]} \arrow[d, "\text{(HTTP POST /ingest + neo4j\_to\_mongo.js)}"] \\
\text{[MongoDB (agri\_lake.lake Collection)]} \arrow[d, leftrightarrow, "\text{(Cypher Query via Neo4j Driver)}"] \\
\text{[Neo4j (Graph DB: Farm → Device → Reading)]}
\end{tikzcd}$$

Key Points
IndexedDB holds local sensor readings (Lab 2).
Node.js server collects and pushes data via /ingest.
Neo4j stores graph-based relationships (Farm, Device, Reading).
Both data sources merged into MongoDB *agri_lake.lake*.
Each document tagged with _lake.sourceDB + timestamp.

Title: *Lessons Learned While Extending the Pipeline*
✅ Learned how to:
Integrate browser-side IndexedDB with backend Node/Express API.
Connect Neo4j using neo4j-driver and export query results.
Structure unified schemas in MongoDB (with metadata fields).
Handle CORS, async fetch, and dotenv configuration.

What Worked Well:
REST ingestion worked smoothly after fixing localhost port.
IndexedDB → Mongo flow validated with Compass quickly.
Metadata tagging (sourceDB, ingestedAt) simplified filtering.

Title: *Challenges, Fixes, and Future Improvements*
Challenges Faced:
ERR_CONNECTION_REFUSED due to wrong port.
metric key mismatch in Neo4j queries.
Mongo connection initially failed due to missing .env path.

What Didn't Work Initially:
Duplicate variable declarations (DB_NAME / result).
Wrong filter path (sourceDB instead of _lake.sourceDB).

Improvements / Future Work:
Automate ingestion (scheduled jobs).
Add validation for schema consistency.
Push data to a hosted MongoDB Atlas instance.