This project implements a full-stack application that performs CRUD operations on a Neo4j graph database using natural language commands. The system uses an open-source LLM to translate natural language into GraphQL queries, executes them on a Neo4j Aura cloud database, and returns the results through a clean React-based interface.

Features ✓ LLM-Powered Query Translation

Uses an open-source LLM (Ollama / LM Studio)
Converts natural language → GraphQL query
Example: “Delete the movie Joker” → Generates a GraphQL mutation deleteMovie query

✓ CRUD Operations (GraphQL + Neo4j Aura)

Create Movie, Actor, Director, Genre nodes
Read/Retrieve (filter, search, match patterns)
Update movie attributes
Delete movies or relationships
All operations use Neo4j GraphQL library schemas

✓Full-Stack GRAND-like Architecture

Backend: Hono (Node.js) + Apollo Server
Database: Neo4j Aura cloud instance
Frontend: React + TypeScript + Vite

LLM Integration: Custom natural language prompt + translation layer

✓CSV Import for Assignment

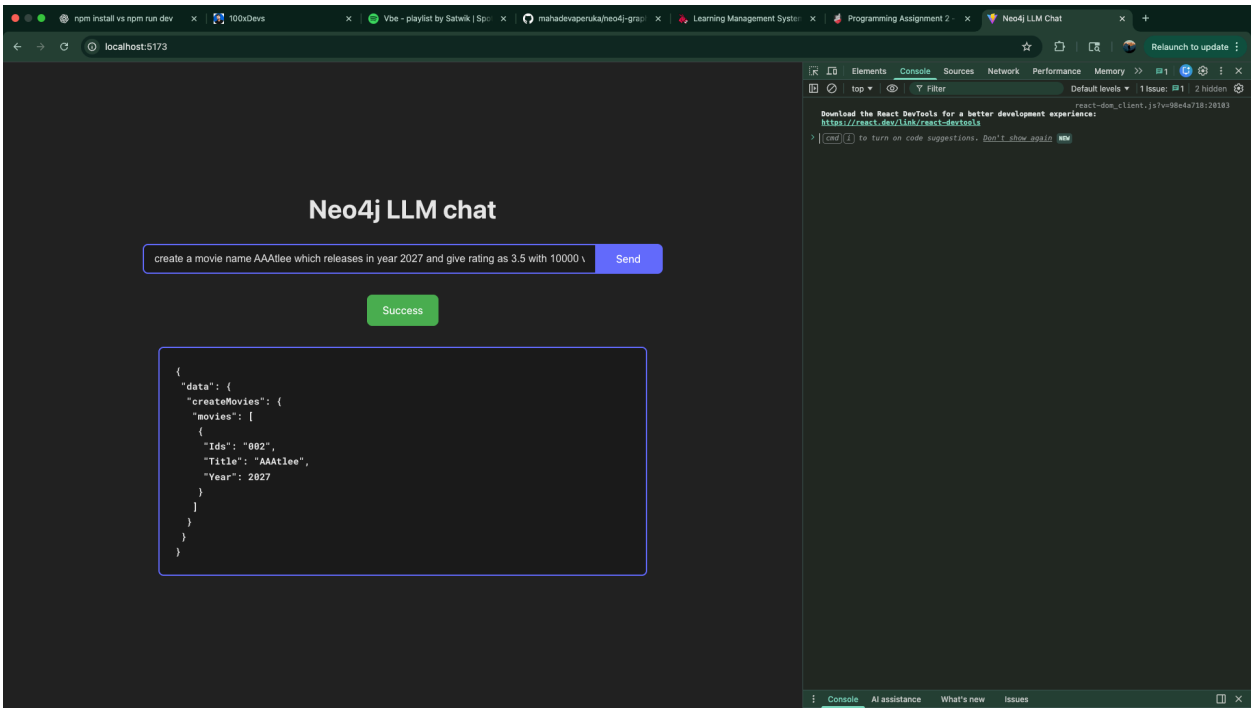
Supports the assignment dataset (IMDB-Movie-Data.csv)
Setup Instructions 1️⃣ Prerequisites
Install:
Node.js 18+
Neo4j Aura Free Tier database
Ollama or LM Studio (for open-source LLM)
Package managers: npm or yarn

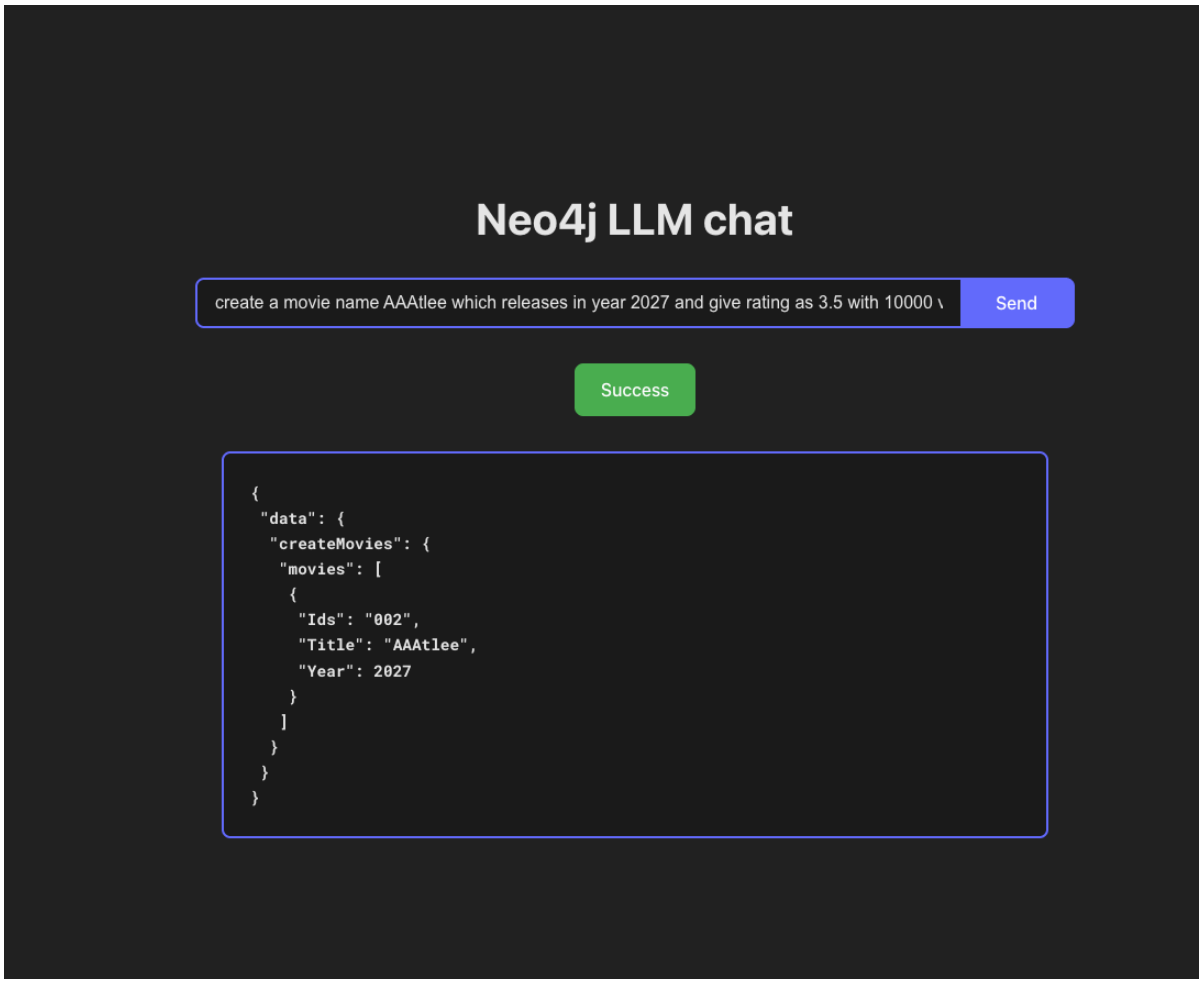
2)Neo4j Aura Setup Step 1 — Create Neo4j Aura DB
Go to: https://console.neo4j.io
Create a free database.
Step 2 — Add credentials to backend/.env
NEO4J_URI=neo4j+s://xxxxxxxx.databases.neo4j.io
NEO4J_USERNAME=neo4j NEO4J_PASSWORD=yourpassword
PORT=3000 REQ_PORT=http://localhost:5173
Step 3 — Import IMDB CSV
Use Neo4j Aura Browser:
LOAD CSV WITH HEADERS FROM 'https://…/IMDB-Movie-Data.csv' AS row CREATE (:Movie { ids: row.Ids, title: row.Title, year: toInteger(row.Year), runtime: toInteger(row.Runtime (Minutes)), rating: toFloat(row.Rating), votes: toInteger(row.Votes), revenue: toFloat(row.Revenue (Millions)), description: row.Description });
Add Actor, Director, Genre nodes similarly.

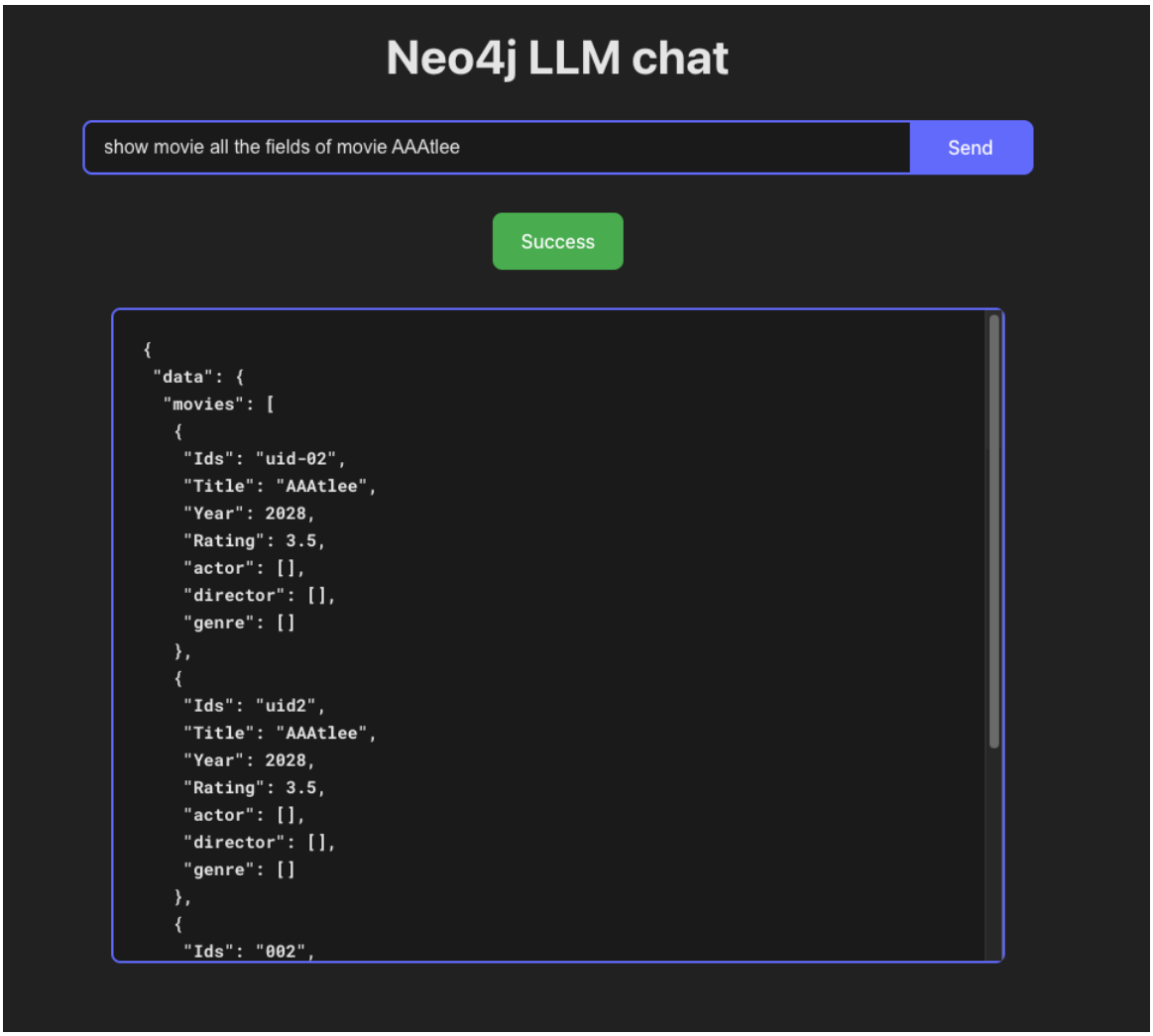
3) Install Dependencies Backend cd backend npm install
Frontend cd frontend npm install

4) Running the Application
Start Backend cd backend npm run build npm start
Start Frontend cd frontend npm run dev.

Sample example :




Neo4j LLM chat
create a movie name AAAtlee which releases in year 2027 and give rating as 3.5 with 10000 \
Send
Success
{
"data": {
"createMovies": {
"movies": [
{
"Ids": "002",
"Title": "AAAtlee",
"Year": 2027
}
]
}
}
}


Neo4j LLM chat
show movie all the fields of movie AAAtlee
Send
Success
{
"data": {
"movies": [
{
"Ids": "uid-02",
"Title": "AAAtlee",
"Year": 2028,
"Rating": 3.5,
"actor": [],
"director": [],
"genre": []
},
{
"Ids": "uid2",
"Title": "AAAtlee",
"Year": 2028,
"Rating": 3.5,
"actor": [],
"director": [],
"genre": []
},
{
"Ids": "002",