

Proyecto Extra

Aplicación Web con FastAPI y Neo4J

Equipo 5

Universidad Autónoma de Yucatán
Modelos de Datos LCC

Agosto–Diciembre 2025

Profesor: M. en C. Luis R. Basto Díaz

Agenda

Meta del Proyecto Extra

- Exponer las operaciones CRUD y el modelo de grafos en una aplicación web simple.
- Levantar API y Neo4J con Docker Compose para asegurar reproducibilidad.
- Ofrecer una UI ligera que permita: seed de datos, CRUD rápido y visualización interactiva del grafo.

Componentes principales

- **docker-compose.yml** con dos servicios:
 - **neo4j** (v5.15): monta neo4j-data/ y el CSV en /import/shopping_behavior.csv. Credenciales neo4j/test1234.
 - **app** (FastAPI): cliente oficial de Neo4J, expuesta en `http://localhost:8000`.
- Red interna: `proyecto-extra-net`.

Estructura del repositorio

- `proyecto-extra/docker-compose.yml`: orquesta API + Neo4J.
- `proyecto-extra/app/`:
 - `main.py`: endpoints **/seed**, CRUD, grafo, health.
 - `static/index.html`: UI HTML/JS con vis-network.
 - Dockerfile y requirements para construir la imagen de la API.
- `neo4j/`: scripts Cypher originales (montados en el contenedor).

Dataset y seed

- Dataset: `data/shopping_behavior.csv` (3,900 transacciones).
- Endpoint **/seed**:
 - Crea constraints (Customer, Product, Category).
 - Carga nodos y relaciones con LOAD CSV.
- Acceso Neo4J: `bolt://localhost:7687`, usuario `neo4j`, password `test1234`.

Endpoints CRUD

- CREATE: /customers, /categories, /products, /purchases, /products/with-category.
- READ: clientes >50, top productos, clientes por categoría, resumen por método de pago, clientes premium.
- UPDATE: edad, suscripción por ubicación, rating promedio, incremento de compras previas, actualización de producto.
- DELETE: cliente + relaciones, compras con rating bajo, productos sin compras, relación producto-categoría, clientes inactivos.
- Utilidades: /seed, /health, Swagger en /docs.

Endpoints de grafo

- `/graph/options`: devuelve valores disponibles para categoría, producto o cliente (pobla los selects de la UI).
- `/graph/sample`: genera un subgrafo centrado en el nodo elegido, con profundidad y límite configurables; devuelve nodos y relaciones para pintar en la UI.

UI (index.html)

- Botones rápidos: **seed**, **health**, **top productos**.
- Formularios CRUD: crear cliente, actualizar edad, eliminar cliente.
- Visor de grafo (vis-network):
 - Selecciona centro (categoría/producto/cliente) y valor desde listas dinámicas.
 - Ajusta profundidad y límite; clic en nodos muestra propiedades.
 - Física ajustada para separar nodos y reducir solapamiento de aristas.

Paso a paso

- ❶ `docker compose up -d` (en `proyecto-extra/`).
- ❷ Ejecutar `/seed` desde la UI o vía HTTP si es la primera vez.
- ❸ Probar health y un READ (`/read/top-products` o el botón en la UI).
- ❹ CRUD rápido desde la UI (crear/actualizar/eliminar cliente).
- ❺ Visualizar el grafo: elegir centro/valor/profundidad y explorar nodos.

Conclusiones

- La app web encapsula el modelo y CRUD en una interfaz sencilla.
- Docker Compose garantiza reproducibilidad: API + Neo4J listos con un comando.
- El visor de grafo permite explorar relaciones cliente-producto-categoría directamente en el navegador.

¡Gracias!