**Resumen de avances — Portafolio Backend & Datos 2025**

(Actualizado: 2025-08-19 23:49)

**1) Pitch breve para entrevista (30–45 s)**

Implementé un monorepo profesional con una API de inventario en Django conectada a MariaDB. Dejé la base de datos lista con Docker Compose (MariaDB + Adminer), configuré variables con .env y cargué datos de ejemplo. Documenté todo en el README y preparé pruebas y CI básicas. El siguiente paso es exponer los endpoints de Órdenes con validación de stock.

**2) Logros clave (para RR. HH.)**

• Monorepo organizado y versionado en GitHub (rama main, convenciones de commits).

• README con badges y secciones separadas para RR. HH. y para desarrolladores.

• Base de datos levantada con un solo comando (Docker Compose) y panel visual con Adminer.

• Datos de ejemplo cargados (20 productos) para demo y pruebas.

• Entorno replicable: pasos claros para clonar, instalar y ejecutar.

**3) Detalle técnico (para desarrolladores)**

• Stack: Django + DRF, MariaDB en contenedor, python-dotenv; Swagger/CI opcional.

• Configuración por entorno mediante archivo .env cargado con python-dotenv.

• DB en contenedor: imagen mariadb:11.4 publicada en el puerto del host 3307.

• Adminer en http://localhost:8080 (Servidor = db, usuario app, clave app).

• Seeds: seeds/products\_fixture.json (SKU, nombre, precio, stock).

• Código limpio: core.Product en admin; base lista para Customer, Order y OrderItem.

**4) Demostración rápida (60 s)**

• Arranque DB: docker compose --env-file .env.db up -d

• Conectar Django: DB\_HOST=127.0.0.1, DB\_PORT=3307 en .env

• Migrar y cargar semillas: python manage.py migrate | python manage.py loaddata ../seeds/products\_fixture.json

• Probar: python manage.py runserver → Admin en http://127.0.0.1:8000/admin

**5) Problemas resueltos y aprendizajes**

• Habilitación de WSL2 e integración con Docker Desktop (Ubuntu).

• Contenedor 'unhealthy' resuelto con down -v y healthcheck más paciente.

• Conexión correcta en Adminer usando 'Servidor = db' (red interna de Docker).

• Selección del intérprete adecuado en VS Code y carga de variables con python-dotenv.

**6) Comandos esenciales**

• git clone <repo>

• cd backend-data-portfolio-2025

• docker compose --env-file .env.db up -d

• cd orders\_inventory\_api

• .\.venv\Scripts\Activate.ps1

• pip install -r requirements.txt

• python manage.py migrate

• python manage.py loaddata ../seeds/products\_fixture.json

• python manage.py runserver

**7) Estado actual**

• Día 1 completado: estructura, GitHub, admin, seeds y documentación.

• Docker para la base de datos operativo (MariaDB + Adminer).

• Listo para implementar CRUD de Customers/Orders/Items y validación de stock.

**8) Siguientes pasos (Día 2)**

• Modelos Customer, Order y OrderItem + endpoints DRF.

• Descuento de stock y cálculo de subtotales.

• Swagger en /api/docs y colección Postman.