

Corzo Zamora, Carmen Cecilia
cecicorzoxd@gmail.com

Documentación de prueba técnica – ERP/BI

Instrucciones generales:

El objetivo es crear un mini-dashboard que muestre transacciones financieras y un reporte de Business Intelligence.

Ejemplo sería registrar compras de algún tipo de inventario y la venta de parte de él. Un sistema de compras y ventas y una pestaña donde se pueda ver estadísticas gráficas de los movimientos.

Backend (Python con FastApi)

Instrucciones:

Utilizar Python con FastAPI.

Los datos pueden almacenarse en memoria (una lista de diccionarios) o en un archivo JSON. No es necesaria una base de datos real para esta prueba.

1. El modelo de la transacción debe tener (al menos):
 - id: int
 - date: str (Ej: "2025-11-07")
 - description: str
 - amount: float (usar números positivos para ingresos, negativos para egresos)
 - category: str (Ej: "Ventas", "Salarios", "Marketing")
2. Crear los siguientes endpoints:
 - POST /transactions: Agrega una nueva transacción.
 - GET /transactions: Lista todas las transacciones.
 - GET /transactions/summary: (Endpoint de analítica) Debe retornar un resumen de los datos, como:

```
JSON {  
    "total_income": 15000.00,  
    "total_expense": -8000.00,  
    "net_total": 7000.00  
}
```

Bonus (Opcional):

- Validación de datos usando Pydantic.
- GET /transactions/summary_by_category: Un endpoint que agrupe los totales por categoría.

Los pasos a seguir fueron:

1. Instalar dependencias:

- Abrir el backend
 - Crear un entorno virtual: **py -m venv venv**
 - Instalar FastAPI y Unicorn con: **python -m pip install fastapi uvicorn**

2. Ejecutar el servidor (necesario NO estar en venv)

- En la terminal levantar server: **py -m unicorn main:app --reload --port 8000**

```
File Edit Selection View Go Run ... ← → [ ] Prueba
```

main.py 3 README.md App.js transactions.json Dashboard.jsx

1 README.md > 2 Funcionalidades
70 # Endpoints Disponibles
75 ## GET /transactions

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS QUERY RESULTS

```
PS C:\Users\DELL\Notebook\Downloads\Prueba\backend> py -m uvicorn main:app --reload --port=8000
```

INFO: Started server process [17288]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: 127.0.0.1:8000 - "GET /index.html" 200 OK
INFO: 127.0.0.1:8000 - "GET /transaction.json" 200 OK
INFO: 127.0.0.1:54233 - "GET /transactions" 200 OK
INFO: 127.0.0.1:54233 - "GET /uvicon.ico" 404 Not Found
INFO: 127.0.0.1:5802 - "GET /transactions/summary" 200 OK
INFO: 127.0.0.1:5802 - "GET /transactions" 200 OK
INFO: 127.0.0.1:5802 - "GET /transactions/summary" 200 OK
INFO: 127.0.0.1:5746 - "GET /HTTP/1.1" 404 Not Found
INFO: 127.0.0.1:5795 - "OPTIONS /transactions" 200 OK
INFO: 127.0.0.1:5795 - "POST /transactions" 200 OK
INFO: 127.0.0.1:5795 - "PUT /transactions" 200 OK
INFO: 127.0.0.1:57931 - "GET /transactions/summary" 200 OK
INFO: 127.0.0.1:57931 - "GET /transactions/summary" 200 OK
INFO: Uvicorn running on http://127.0.0.1:8000 (Press Ctrl+C to quit)

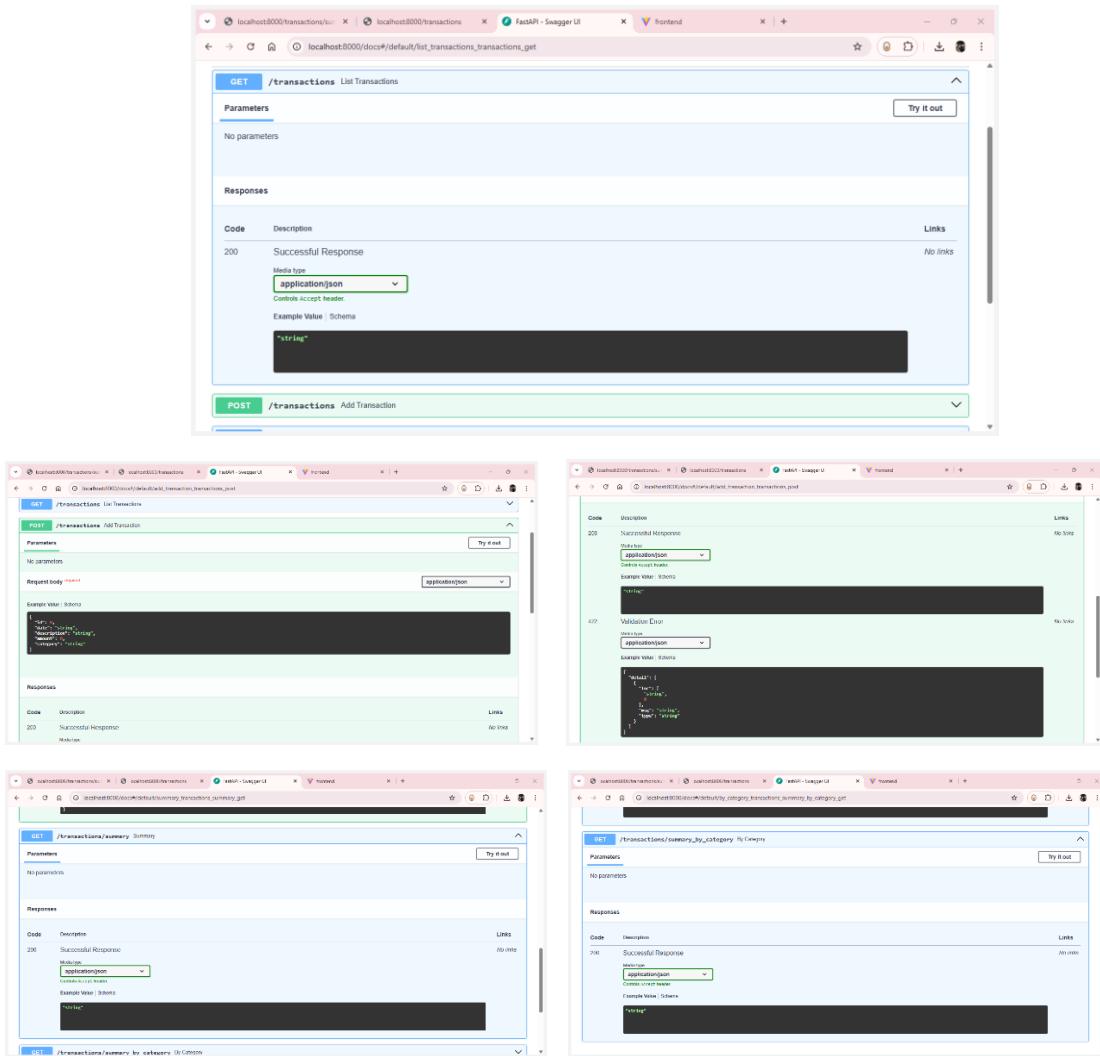
INFO: Started reload process [17289] using StarletteReload
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: 127.0.0.1:8000 - "GET /index.html" 200 OK
INFO: 127.0.0.1:8000 - "GET /transaction.json" 200 OK
INFO: 127.0.0.1:54233 - "GET /transactions" 200 OK
INFO: 127.0.0.1:54233 - "GET /uvicon.ico" 404 Not Found
INFO: 127.0.0.1:5802 - "GET /transactions/summary" 200 OK
INFO: 127.0.0.1:5802 - "GET /transactions" 200 OK
INFO: 127.0.0.1:5802 - "GET /transactions/summary" 200 OK
INFO: 127.0.0.1:5746 - "GET /HTTP/1.1" 404 Not Found
INFO: 127.0.0.1:5795 - "OPTIONS /transactions" 200 OK

3. Encontrar backend en los links:

- <http://localhost:8000>
 - Swagger: <http://localhost:8000/docs>



<http://localhost:8000/docs#/>



The image displays five separate browser tabs, each showing a different endpoint from a FastAPI application's Swagger UI documentation. The tabs are arranged in two rows: the top row contains three tabs, and the bottom row contains two tabs.

- Top Row, Left Tab:** Shows the `GET /transactions` endpoint for listing transactions. It indicates "No parameters" and "No links". The "Responses" section shows a successful response (200) with a media type of `application/json` and example value `"string"`.
- Top Row, Middle Tab:** Shows the `POST /transactions` endpoint for adding a transaction. It indicates "No parameters" and "No links". The "Responses" section shows a successful response (200) with a media type of `application/json` and example value `"string"`.
- Top Row, Right Tab:** Shows the `GET /transactions/summary` endpoint for a financial summary. It indicates "No parameters" and "No links". The "Responses" section shows a successful response (200) with a media type of `application/json` and example value `"string"`.
- Bottom Row, Left Tab:** Shows the `GET /transactions/summary_by_category` endpoint for a financial summary by category. It indicates "No parameters" and "No links". The "Responses" section shows a successful response (200) with a media type of `application/json` and example value `"string"`.
- Bottom Row, Right Tab:** Shows the same `GET /transactions/summary_by_category` endpoint as the left tab, indicating it is a duplicate entry.

Explicación:

- **POST /transactions:** Agregar una transacción
- **GET /transactions:** Lista de transacciones
- **GET /transactions/summary:** Retorna el resumen financiero
- **GET /transactions/summary by category:** Totales por categoria

Frontend (React)

Instrucciones:

Usar React con functional components y hooks (useState, useEffect).

1. Crear dos secciones o "páginas" (puedes usar un simple router o solo pestañas):
 - Sección "Transacciones":
 - Un formulario simple para agregar nuevas transacciones
 - (consumiendo POST /transactions).
 - Una tabla o lista que muestre todas las transacciones
 - (consumiendo GET /transactions).
 - Sección "Dashboard":
 - Mostrar los datos del resumen (Total Ingresos, Total Egresos) consumiendo el endpoint GET /transactions/summary del backend.
 - Embeber un reporte de Looker Studio.
2. Requisito Específico (Looker Studio):
Para esta prueba, no necesitas crear un reporte. Simplemente embebe este reporte público de ejemplo usando un <iframe>:
 - URL de ejemplo para embeber:

HTML

<https://lookerstudio.google.com/embed/reporting/0B5FF6JBKbLCSaW1VbDI2S3prYUE/page/6SXD>

El objetivo es demostrar que puedes integrar un reporte externo de BI dentro de la aplicación de React.

Bonus (Opcional):

- Usar una librería de componentes simple (como Bootstrap o Tailwind CSS) para que se vea ordenado.
- Mostrar un mensaje de "Cargando..." mientras se obtienen los datos.

Los pasos a seguir fueron:

1. Instalar dependencias para node

- En la terminal: npm install

2. Ejecutar el servidor de desarrollo

- En la terminal: cd "C:\Users\DELL Notebook\Downloads\Prueba\frontend"
- Luego copilar: npm run dev

3. Frontend queda disponible en:

<http://localhost:5173>

The image displays two screenshots of a financial dashboard application, likely built with a Node.js backend and a frontend framework like React.

Screenshot 1 (Top): Nueva Transacción (New Transaction) Form

This screenshot shows a modal window titled "Nueva Transacción" (New Transaction). It contains fields for "ID" (text input), "Fecha" (date input), "Descripción" (text input), "Monto" (text input), and "Categoria" (text input). Below the form is a button labeled "Agregar Transacción" (Add Transaction). To the right is a table titled "Transacciones Registradas" (Registered Transactions) with columns: ID, Fecha, Descripción, Monto, and Categoría. The table contains the following data:

ID	Fecha	Descripción	Monto	Categoría
1	2025-II-29	Venta de prueba	150	ventas
2	2025-II-29	Alquiler de local	-15	egresos
3	2025-II-29	Venta de prueba	-150	ventas
1	2025-02-13	505	50	20

Screenshot 2 (Bottom): Resumen Financiero (Financial Summary) Section

This screenshot shows a summary section titled "Resumen Financiero". It displays three boxes: "Ingresos: Q200", "Egresos: Q-165", and "Neto: Q35". Below this is a section titled "Reporte BI (Looker Studio)" with a placeholder "Añadir título del informe" (Add report title). A dropdown menu "Selecciona un periodo" (Select a period) is shown. The final part is a chart titled "Monto por Categoría" (Amount by Category) showing a bar chart with categories "Ventas" and "Egresos".

IMPORTANTE:

PARA ESTA PRUEBA SE UTILIZO VISUAL STUDIO CODE