

Actividad integradora de Docker

Esta actividad integradora consiste en:

- Elegir una tecnología de nuestra preferencia y para crear su imagen de servidor con Dockerfile e incluirlo en un Compose.
 - La imagen debe tener al menos una instrucción personalizada que se imprima por pantalla.
 - El Docker Compose debe contener una instrucción build a nuestro Dockerfile y un componente de persistencia que interactúe con nuestro server.
-

1. Descripción general

Este proyecto consiste en un **agente multifunción** desarrollado en **Python 3.11**, que permite interactuar con herramientas para:

- Registrar y listar transacciones financieras (gastos, ingresos, préstamos) en una base de datos PostgreSQL.

El agente se ejecuta como un **servidor web** y puede integrarse en un entorno de desarrollo local mediante **Docker** y **Docker Compose**.

Tecnologías elegidas:

- **Python 3.11:** lenguaje de programación principal, cómodo y ampliamente usado en proyectos laborales y educativos.
 - **PostgreSQL:** base de datos relacional para persistencia de transacciones.
 - **Docker + Docker Compose:** para empaquetar el agente y su base de datos en contenedores reproducibles.
 - **Requests:** para consultas a APIs externas (clima, zona horaria).
-

2. Estructura de archivos

```
agent/
├── Dockerfile
├── docker-compose.yml
├── requirements.txt
├── README.md
├── .env
├── .gitignore
├── init.sql
├── multi_tool_agent/
│   ├── __init__.py
│   ├── agent.py
│   └── tools/
│       ├── __init__.py
│       ├── balance.py
│       └── transactions.py
```

```
| | | agent_db.py
| | | date_tools.py
|--- README.md
```

3. Docker y Docker Compose

Dockerfile

- Imagen base: `python:3.11-slim`
- Crea un **virtualenv** `.venv`
- Instala dependencias de Python desde `requirements.txt`
- Expone el puerto `8000`

docker-compose.yml

- Define dos servicios:
 1. **agent**: nuestro servidor Python basado en Dockerfile.
 2. **db**: contenedor PostgreSQL con persistencia en un volumen local.
- El servicio `agent` depende de `db` y se conecta a él mediante variables de entorno.

4. Uso

Si no tenemos el repositorio en el local

1. Clonar el repositorio.

Clonar el siguiente repositorio publico.

```
git clone https://github.com/rvegabaldiviezo/agent.git
```

2. Entrar del directorio de la App.

```
cd agent
```

Si ya tenemos el repositorio en el local

3. Cambiar a la rama develop y traerse los ultimos cambios

```
git checkout -b develop
git pull origin develop
```

4. Construir y levantar los servicios:

```
docker-compose up --build -d
```

Containers [Give feedback](#)

View all your running containers and applications. [Learn more](#)

Container CPU usage
0.22% / 1200% (12 CPUs available)

Container memory usage
300.77MB / 15.21GB

Only show running containers

<input type="checkbox"/>	Name	Container ID	Image	Port(s)	CPU (%)
<input type="checkbox"/>	agent	-	-	-	0.22%
<input type="checkbox"/>	finanzas_db	2f29883bf156	postgres:15	5432:5432	0%
<input type="checkbox"/>	finance_agent	d33a5f18acb6	agent-agent	8000:8000	0.22%

5. Acceder al servidor

Abre tu navegador y entra a: <http://localhost:8000/>

Agent Development Kit

multi_tool_agent

Trace

Events

State

Artifacts

Sessions

Eval

Invocations

buena

que día es hoy

gaste en el día de hoy unos 15000 pesos en frutas para comer

calcula la fecha

no

si es una gasto

la fecha es la de hoy y el monto y descripción es esa

lista

que clima hay en salta capital?

de buenos aires?

porque?

SESSION ID: 5a5a2553-9b9a-4a74-b3a1-1ccb3911ada6

Token Streaming

New Session

Hola! Recuerda que puedo:

- registrar tus ingresos, gastos y préstamos.
- calcular el balance actual.
- listar las transacciones recientes (por defecto texto). También puedo generar una imagen de la tabla de transacciones agrupadas por categoría.
- mostrar la fecha de hoy.

¿En qué te puedo ayudar?

get_today_date

get_today_date

Hoy es 2025-09-01

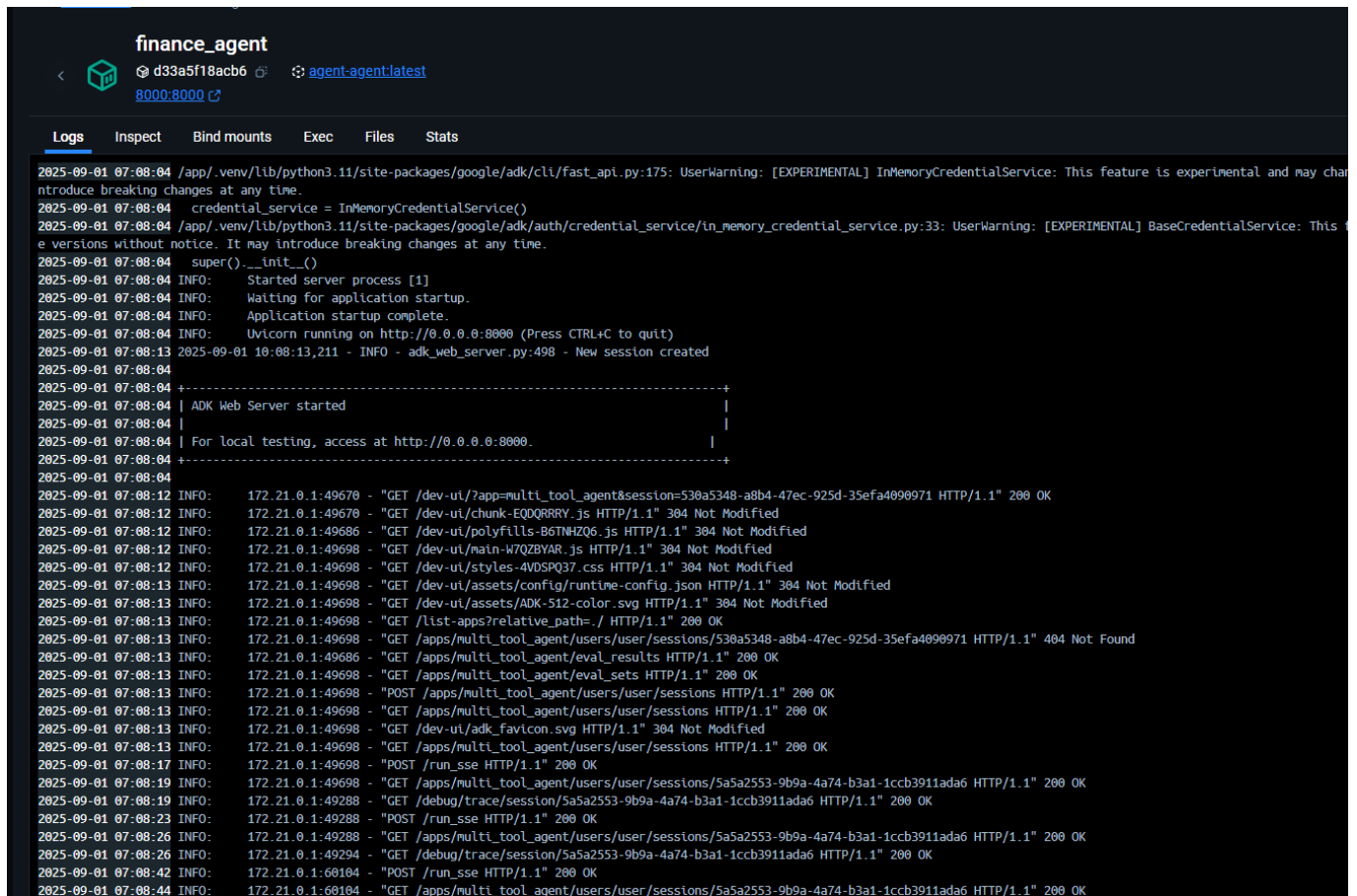
gaste en el día de hoy unos 15000 pesos en frutas para comer

calcula la fecha

no

si es una gasto

la fecha es la de hoy y el monto y descripción es esa



```
finance_agent
d33a5f18acb6 agent-agent:latest
8000:8000

Logs Inspect Bind mounts Exec Files Stats

2025-09-01 07:08:04 /app/.venv/lib/python3.11/site-packages/google/adk/cli/fast_api.py:175: UserWarning: [EXPERIMENTAL] InMemoryCredentialService: This feature is experimental and may char
ntroduce breaking changes at any time.
2025-09-01 07:08:04 credential_service = InMemoryCredentialService()
2025-09-01 07:08:04 /app/.venv/lib/python3.11/site-packages/google/adk/auth/credential_service/in_memory_credential_service.py:33: UserWarning: [EXPERIMENTAL] BaseCredentialService: This
e versions without notice. It may introduce breaking changes at any time.
2025-09-01 07:08:04 super().__init__()
2025-09-01 07:08:04 INFO: Started server process [1]
2025-09-01 07:08:04 INFO: Waiting for application startup.
2025-09-01 07:08:04 INFO: Application startup complete.
2025-09-01 07:08:04 INFO: Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
2025-09-01 07:08:13 2025-09-01 10:08:13,211 - INFO - adk_web_server.py:498 - New session created
2025-09-01 07:08:04 +-----+
2025-09-01 07:08:04 | ADK Web Server started |
2025-09-01 07:08:04 | |
2025-09-01 07:08:04 | For local testing, access at http://0.0.0.0:8000. |
2025-09-01 07:08:04 +-----+
2025-09-01 07:08:04
2025-09-01 07:08:12 INFO: 172.21.0.1:49670 - "GET /dev-ut/7app=multi_tool_agent&session=530a5348-a8b4-47ec-925d-35efa4090971 HTTP/1.1" 200 OK
2025-09-01 07:08:12 INFO: 172.21.0.1:49670 - "GET /dev-ut/chunk-EQDQRRRY.js HTTP/1.1" 304 Not Modified
2025-09-01 07:08:12 INFO: 172.21.0.1:49686 - "GET /dev-ut/polyfills-B6TNH2Q6.js HTTP/1.1" 304 Not Modified
2025-09-01 07:08:12 INFO: 172.21.0.1:49698 - "GET /dev-ut/main-W7QZBYAR.js HTTP/1.1" 304 Not Modified
2025-09-01 07:08:12 INFO: 172.21.0.1:49698 - "GET /dev-ut/styles-4VDSPO37.css HTTP/1.1" 304 Not Modified
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "GET /dev-ut/assets/config/runtime-config.json HTTP/1.1" 304 Not Modified
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "GET /dev-ut/assets/ADK-512-color.svg HTTP/1.1" 304 Not Modified
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "GET /list-apps?relative_path=/ HTTP/1.1" 200 OK
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "GET /apps/multi_tool_agent/users/user/sessions/530a5348-a8b4-47ec-925d-35efa4090971 HTTP/1.1" 404 Not Found
2025-09-01 07:08:13 INFO: 172.21.0.1:49686 - "GET /apps/multi_tool_agent/eval_results HTTP/1.1" 200 OK
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "GET /apps/multi_tool_agent/eval_sets HTTP/1.1" 200 OK
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "POST /apps/multi_tool_agent/users/user/sessions HTTP/1.1" 200 OK
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "GET /apps/multi_tool_agent/users/user/sessions HTTP/1.1" 200 OK
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "GET /dev-ut/adk_favicon.svg HTTP/1.1" 304 Not Modified
2025-09-01 07:08:13 INFO: 172.21.0.1:49698 - "GET /apps/multi_tool_agent/users/user/sessions HTTP/1.1" 200 OK
2025-09-01 07:08:17 INFO: 172.21.0.1:49698 - "POST /run_sse HTTP/1.1" 200 OK
2025-09-01 07:08:19 INFO: 172.21.0.1:49698 - "GET /apps/multi_tool_agent/users/user/sessions/5a5a2553-9b9a-4a74-b3a1-1ccb3911ada6 HTTP/1.1" 200 OK
2025-09-01 07:08:19 INFO: 172.21.0.1:49288 - "GET /debug/trace/session/5a5a2553-9b9a-4a74-b3a1-1ccb3911ada6 HTTP/1.1" 200 OK
2025-09-01 07:08:23 INFO: 172.21.0.1:49288 - "POST /run_sse HTTP/1.1" 200 OK
2025-09-01 07:08:26 INFO: 172.21.0.1:49698 - "GET /apps/multi_tool_agent/users/user/sessions/5a5a2553-9b9a-4a74-b3a1-1ccb3911ada6 HTTP/1.1" 200 OK
2025-09-01 07:08:26 INFO: 172.21.0.1:49294 - "GET /debug/trace/session/5a5a2553-9b9a-4a74-b3a1-1ccb3911ada6 HTTP/1.1" 200 OK
2025-09-01 07:08:42 INFO: 172.21.0.1:60104 - "POST /run_sse HTTP/1.1" 200 OK
2025-09-01 07:08:44 INFO: 172.21.0.1:60104 - "GET /apps/multi_tool_agent/users/user/sessions/5a5a2553-9b9a-4a74-b3a1-1ccb3911ada6 HTTP/1.1" 200 OK
```

5. Inputs de prueba y outputs esperados

La interacción con el agente se realiza mediante chat: los inputs son preguntas o comandos, y los outputs son respuestas del agente.

Prueba 1: Consultar balance inicial

Input: ¿Cuál es mi balance actual?

Output esperado: 0 (sin transacciones registradas)

Prueba 2: Registrar un nuevo ingreso

Input: Registra un ingreso de 1500 por "venta de teclado"

Output esperado: Confirmación de que la transacción fue registrada exitosamente.

Prueba 3: Consultar balance después de la transacción

Input: ¿Cuál es mi balance actual?

Output esperado: 1500

Prueba 4: Verificar persistencia de datos

Instrucciones:

Detener contenedores: `docker-compose down`

Levantar contenedores de nuevo: `docker-compose up -d --build`

Input: ¿Cuál es mi balance actual?

Output esperado: 1500 (persistencia confirmada a través del volumen de PostgreSQL)

6. Modificación de Inputs para Otros Outputs

El agente responde según las instrucciones en lenguaje natural:

- Registrar otros valores:
 - Input: Registra un gasto de 200 por "compra de café"
 - Output: El balance se reduce según el gasto. Ejemplo: 1500 → 1300.
- Consultar historial:
 - Input: Muéstrame mis últimas transacciones
 - Output: Lista de todas las transacciones registradas.

La flexibilidad del agente permite múltiples variaciones de comandos y respuestas según los datos en la base de datos y el historial de la conversación.