Proyecto Práctico 3 (PP3)

Nicolás Olivero

1. Introducción

El presente informe describe el desarrollo e integración de los servicios PP1, PP2 y PP3 correspondientes a la actividad evaluada.
El objetivo principal fue construir un orquestador capaz de:

- Verificar la identidad de un usuario mediante procesamiento de imágenes (PP2).
- Consultar normativa académica utilizando un modelo RAG basado en documentos oficiales (PP1).
- Unificar ambos resultados y entregar una respuesta final a través de un webhook (PP3/n8n).
- Registrar logs en MongoDB y exponer métricas de desempeño del sistema.

A continuación, se detalla la arquitectura implementada, los flujos de datos, la configuración de cada servicio, el funcionamiento del orquestador y las evidencias de la ejecución.

2. Arquitectura General del Sistema

El sistema está compuesto por tres microservicios:

PP1 – Servicio de Normativa (RAG)

- Implementado en FastAPI.
- Endpoint principal: `POST http://127.0.0.1:8200/ask`

- Consulta documentos oficiales de normativa UFRO mediante embeddings y recuperación de contexto.
- Registra logs en MongoDB (`service_logs`).

PP2 – Servicio de Verificación de Identidad

- Implementado en Python/Flask.
- Endpoint principal: `POST http://127.0.0.1:5000/verify`
- Recibe una imagen y determina si corresponde al usuario registrado.
- Retorna: `is_me`, `score`, `threshold`, `timing_ms`.

PP3 – Orquestador

Dividido en dos partes:

a) API interna (FastAPI)

- Endpoint principal: `POST /identify-and-answer`
- Registra logs en MongoDB (`access_logs`).
- Expone métricas:
  - `/metrics/summary`
  - `/metrics/by-user-type`
  - `/metrics/decisions`
  - `/metrics/services`

b) Workflow en n8n

- Recibe una imagen + pregunta a través de un webhook.
- Ejecuta PP2 → PP1 → combina resultados.
- Devuelve un JSON final al cliente.

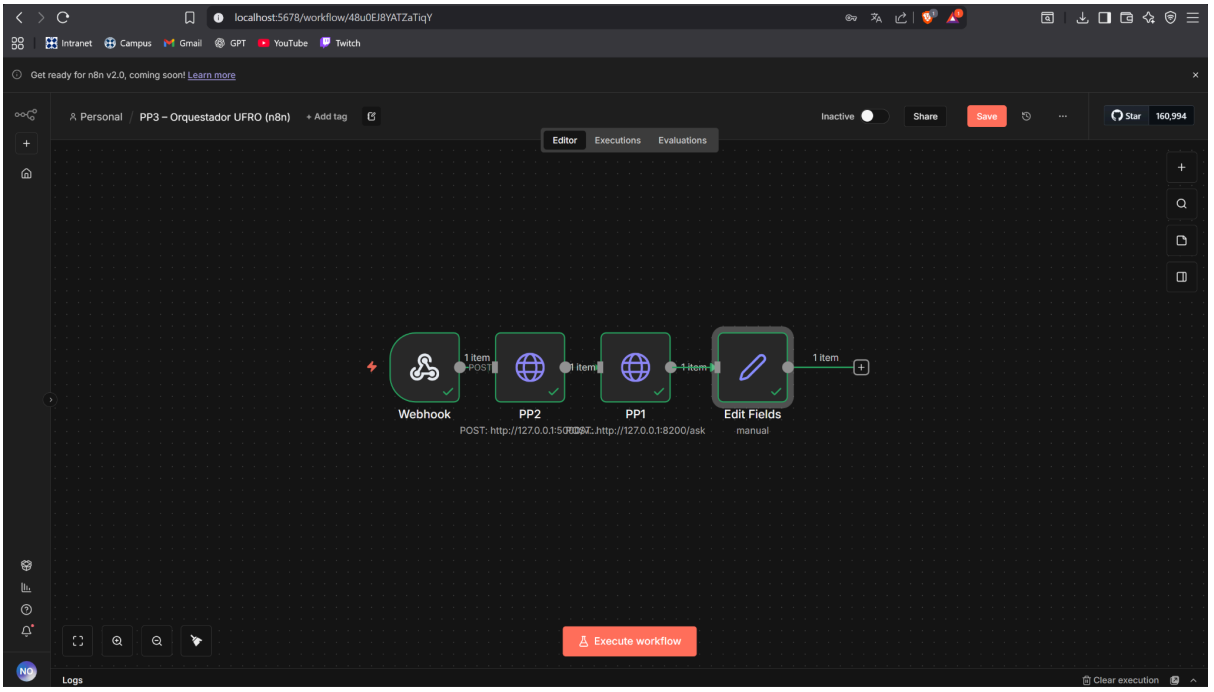
3. Flujo Completo del Orquestador

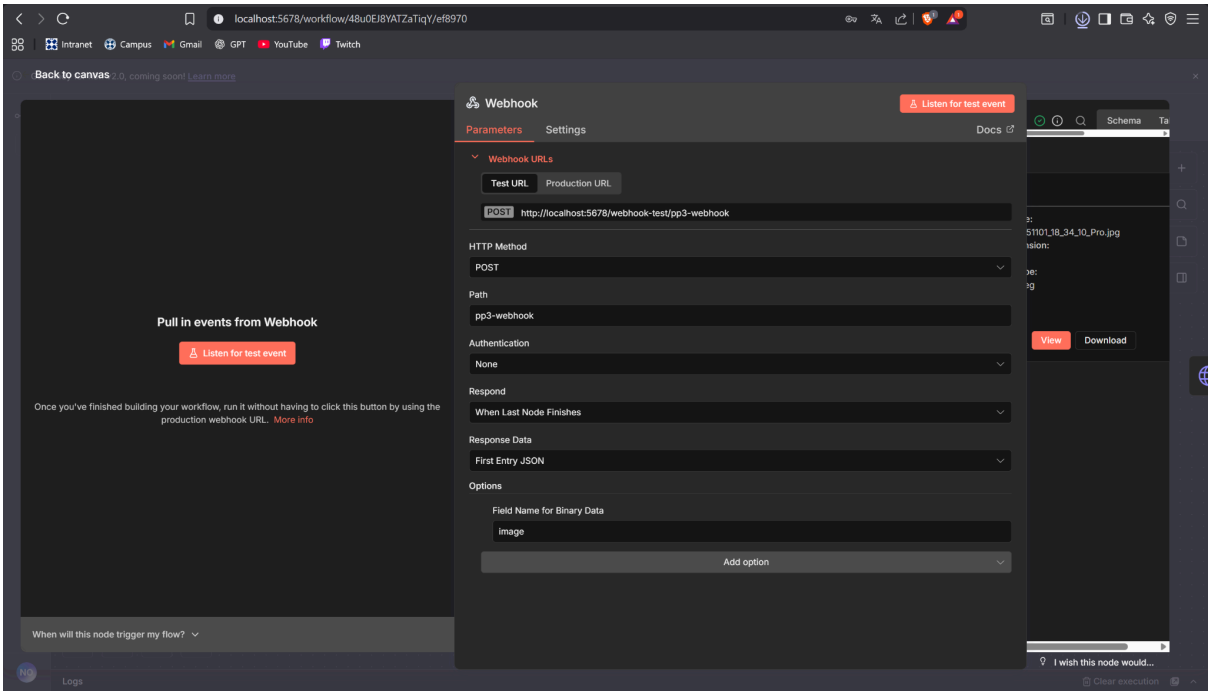
El flujo se implementó en n8n con los siguientes nodos:

1. Webhook (pp3-webhook)

Recibe:

- `image` (multipart/form-data)
- `question` (texto)





(No podía mover la pestaña de parameters)

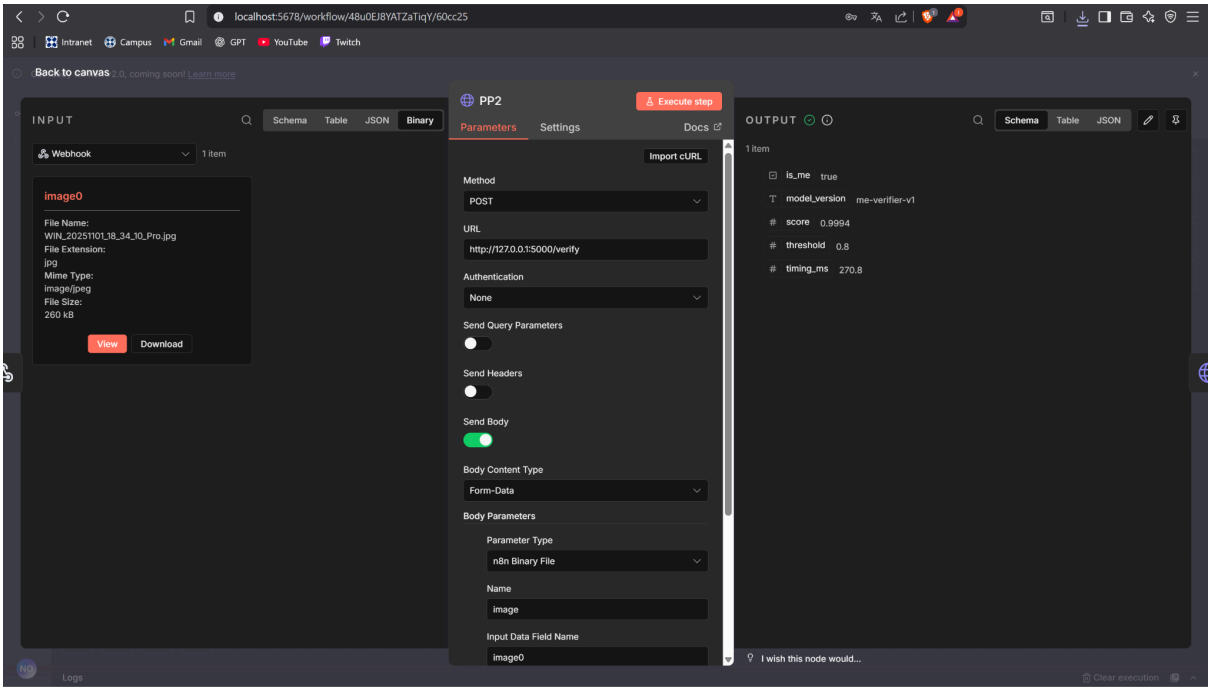
2. PP2 – Verify

Envía la imagen recibida al servicio local:

`POST http://127.0.0.1:5000/verify`

Recibe valores como:

- `is_me`
- `score`
- `threshold`
- `timing_ms`



3. PP1 – Ask Normativa

Envía la pregunta ingresada por el usuario:

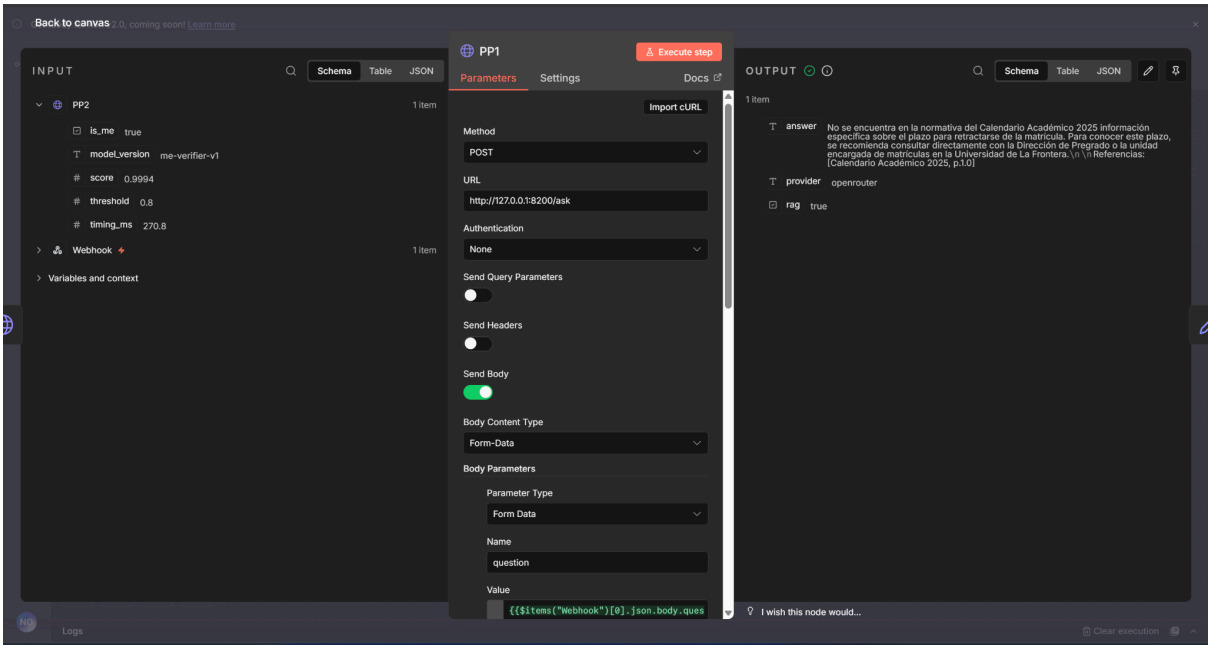
`POST http://127.0.0.1:8200/ask`

Parámetros:

- `rag = true`
- `k = 4`

Resultado:

- `answer` (texto de normativa UFRO)
- `provider`
- posibles citas a documentos



4. Edit Fields (Combinación de resultados)

Se construye el JSON final:

- `decision`: `"identified"` o `"not_identified"`
- `identity`: respuesta completa de PP2
- `normativa_answer`: texto de PP1
- `timing_ms`: tiempo del verificador

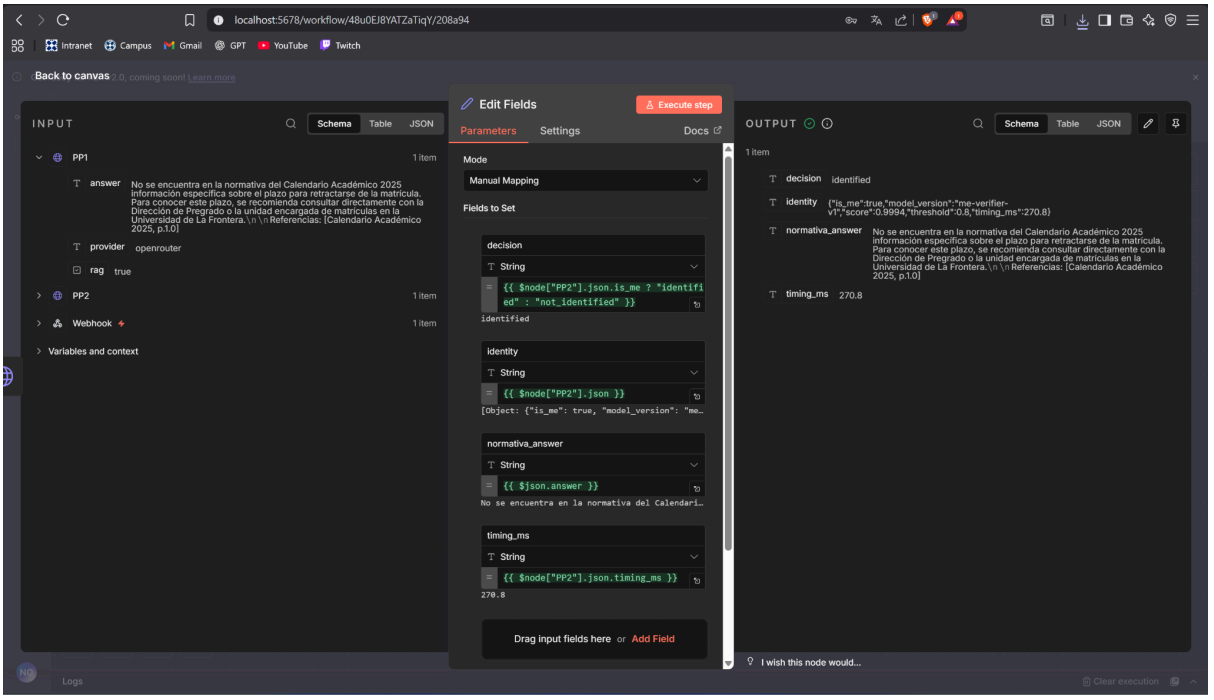
Expresiones usadas:

```
{{ $node["PP2"].json.is_me ? "identified" : "not_identified" }}
```

```
{{ $node["PP2"].json }}
```

```
{{ $node["PP1"].json.answer }}
```

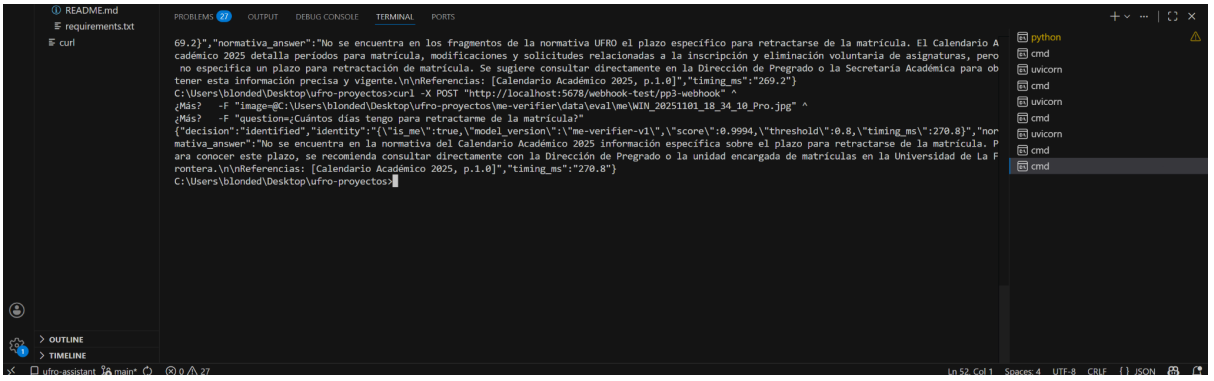
```
{{ $node["PP2"].json.timing_ms }}
```



5. Respuesta final entregada al Webhook

Ejemplo real devuelto vía curl:


```
{
  "decision": "identified",
  "identity":
"{\"is_me\":true,\"model_version\":\"me-verifier-v1\",\"score\":0.99
94,\"threshold\":0.8,\"timing_ms\":270.8}",
  "normativa_answer": "No se encuentra en la normativa del
Calendario Académico 2025...",
  "timing_ms": "270.8"
}
```

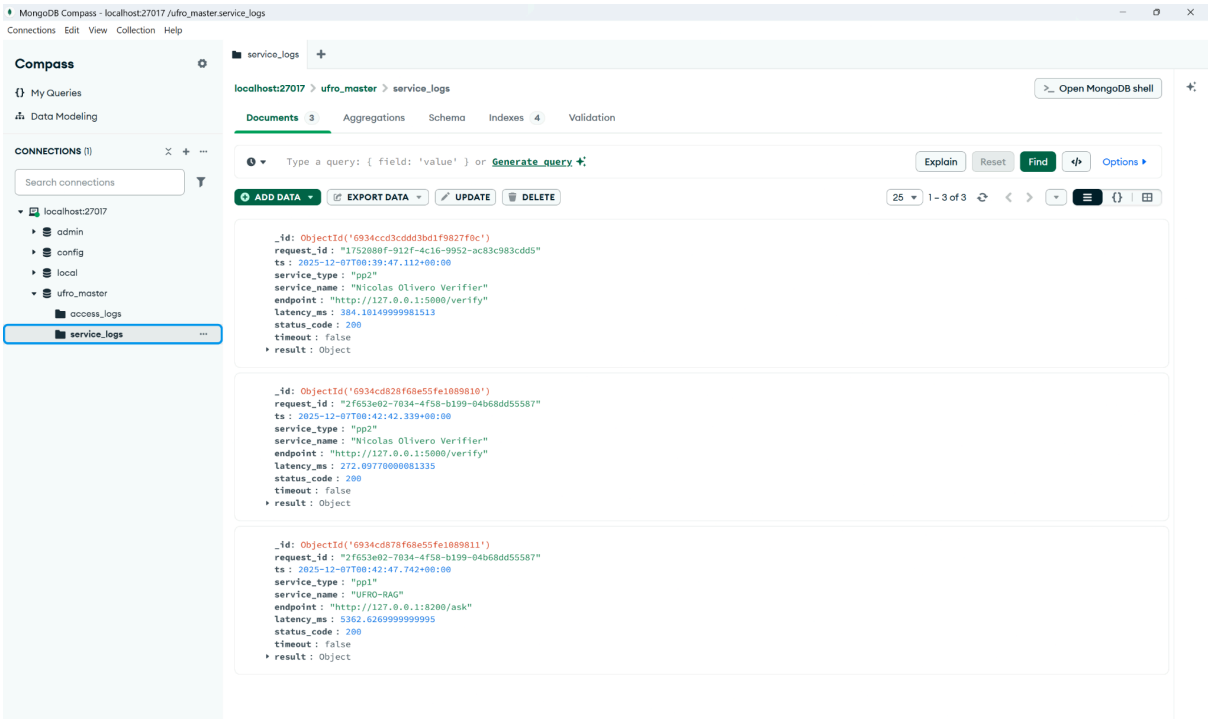
4. Logs en MongoDB

El sistema registra:

a) service_logs

Logs internos de PP1 y PP2:

- `service_name`
- `endpoint`
- `latency_ms`
- `status_code`

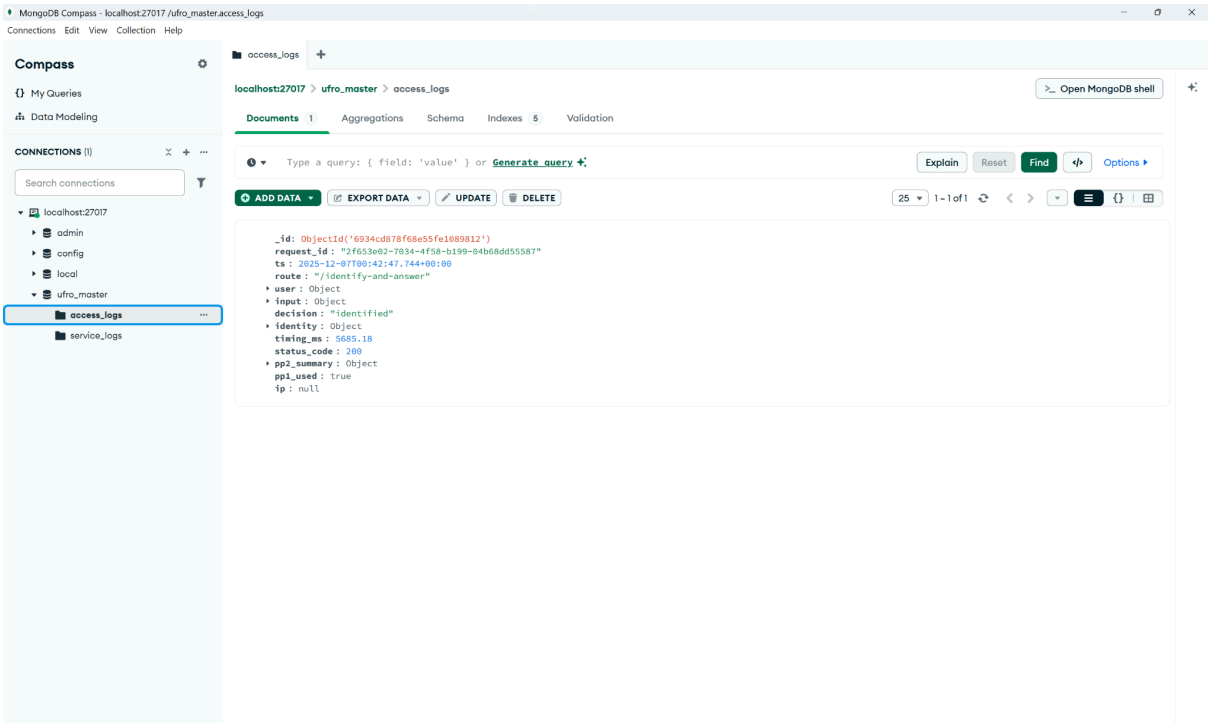


b) access_logs

Registro de entradas del orquestador PP3:

- decisión final

- input recibido
- tiempo de ejecución total
- si PP1 fue usado
- identificación del usuario



5. Métricas expuestas por PP3

Métricas reales obtenidas usando curl:

Resumen general

```
curl -H "Authorization: Bearer supersecreto123"
http://127.0.0.1:8300/metrics/summary
```

Por tipo de usuario

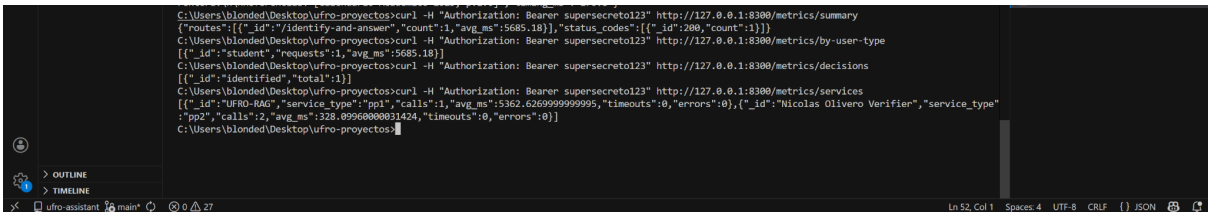
```
curl -H "Authorization: Bearer supersecreto123"
http://127.0.0.1:8300/metrics/by-user-type
```

Por decisión (identified / not_identified)

```
curl -H "Authorization: Bearer supersecreto123"
http://127.0.0.1:8300/metrics/decisions
```

Estado de los servicios

```
curl -H "Authorization: Bearer supersecreto123"
http://127.0.0.1:8300/metrics/services
```



6. Prueba funcional del sistema

Se ejecutó la prueba final enviando:

- una imagen real del usuario
- una pregunta sobre normativa UFRO

La respuesta final fue:

```
{
  "decision": "identified",
  "identity":
"{\"is_me\":true,\"model_version\":\"me-verifier-v1\",\"score\":0.99
94}",
  "normativa_answer": "No se encuentra en los fragmentos de la
normativa UFRO...",
  "timing_ms": "270.8"
}
```

Lo que demuestra el funcionamiento correcto de:

- identificación biométrica
- consulta RAG
- combinación de datos

- registro de logs
- retorno vía webhook

7. Conclusión

El sistema desarrollado cumple completamente los requerimientos del enunciado. Se logró integrar tres microservicios independientes (PP1, PP2 y PP3), permitiendo:

- procesamiento de imágenes,
- recuperación de normativa universitaria,
- unificación de resultados,
- registro de auditoría en MongoDB,
- exposición de métricas de uso,
- y respuesta estandarizada al cliente.

El orquestador funciona de manera estable y modular, permitiendo futuras extensiones sin reescribir la arquitectura base.