An approach using Microservices,
HATEOS, and WebSocket.

Queue management system

Student: Gabriel Henrique Furtado Babrosa
Teacher: Gracon Lima
Subject: Distributed Systems Development

Topics:

What will be presented?

Solution overview

Microservice architecture

HATEOS

Real time comunication

demonstration

General Apresentation

What will be presented?

The project's objective was to develop a distributed queue management system using WebSocket technology for bidirectional communication. The system aims to monitor the service flow and dynamically calculate the average waiting time, ensuring real-time updates of all connected panels as soon as new tickets are requested or called.

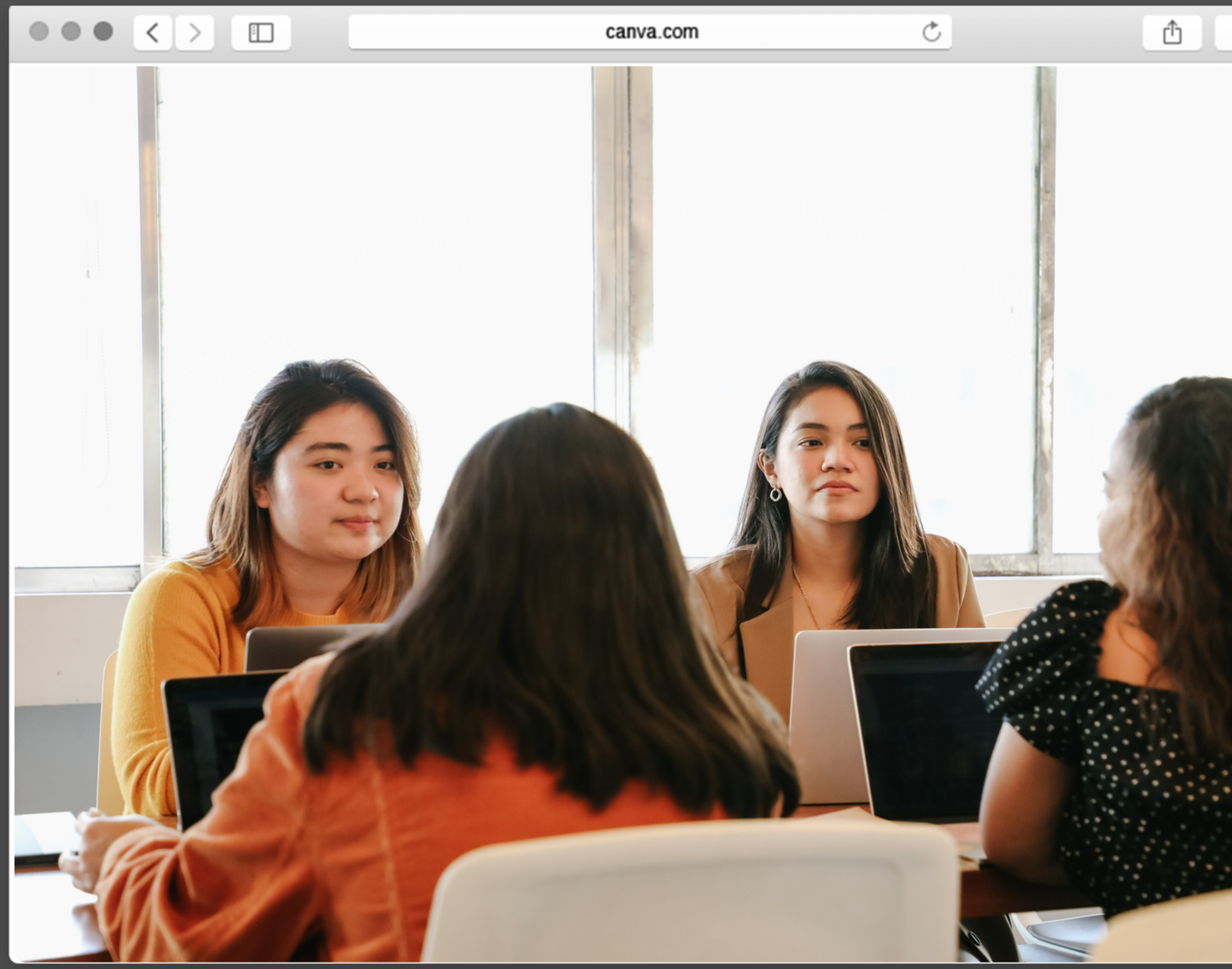
System Architecture

- API Gateway on port 8000
- Ticket Service on port 8001
- Stats Service on port 8002
- Data flow: The Gateway consumes internal services via HTTP and delivers the aggregated result.

HATEOS

The front-end lacks intelligence.

- If the user is an Admin → they receive the link “call_next”
- If the user is a Client → they receive the link “get_password”
- If the user is a Client with a password → they only receive the option to wait
- PS: The application still has a button to retrieve another password due to testing done in the project.



Real time with WebSocket

1. Problems with classic HTTP: the F5 key
2. Conection persistence with Client and Gateway
3. Gateway triggers PASSWORD_UPDATED event for all
4. Updates on multiple screens without loading

Technologies and Demonstration

Python 3.10+

FastAPI

Uvicorn (ASGI)

Python-HTTPx

HTML

JS Vanilla

Swagger UI

THANKS!