



Docker Microservices

Architecture Visualizer + Live Demo

Some services offline E-Commerce System

User Service GET POST

GET /users - list all users **TEST ▶**

POST /users/register - create user **TEST ▶**

Product Service GET POST

GET /products - list all products **TEST ▶**

POST /products - add product **TEST ▶**

Order Service GET POST

GET /orders - list all orders **TEST ▶**

POST /orders - place order **TEST ▶**

Move the cursor



Docker Microservices

Architecture Visualizer + Live Demo

Some services offline E-Commerce System

HTTP Request Async Event DB Query User Service Product Service Order Service

Register Flow

Order Flow

Docker Concepts

- 1 Client sends **POST /users/register** with name, email, password to the API Gateway on port **:8000**
- 2 Gateway checks auth headers, rate limits, then **routes** to User Service **:8001**
- 3 User Service validates data, hashes password, writes to isolated **users.db** – no other service touches this DB
- 4 User Service returns **201 Created** + JWT token back through Gateway to client

Move the

LIVE API TESTER

GATEWAY

:8000

offline

USERS

:8001

offline

PRODUCTS

:8002

offline

ORDERS

:8003

offline

API Response Terminal

```
# Docker Microservices – E-Commerce Demo
# Click any TEST button above to fire a real API request
# Make sure all 4 Python services are running first (see guide)

→ Architecture: Client → API Gateway → [Users|Products|Orders]
→ Each service has its own SQLite database (isolated)
→ Orders service calls Users + Products for cross-service validation

ready
```