

3 Critical Things You Might Not Realize Yet

Architecture Team

December 22, 2025

You asked: *"Is there anything I still don't realize?"* Yes. Here are the **3 most common specific traps** beginners fall into when moving to Docker/Microservices.

1 1. The "Localhost" Lie (Networking)

TRAP: "Connection Refused"

In your code, you might write: `connect('localhost:3004')` to reach the Inventory. **This will fail inside Docker.** Inside the "Orders Container", `localhost` is *itself*. It is NOT your laptop, and it is NOT the Inventory container.

SOLUTION: Use Service Names

In `docker-compose.yml`, you named your service `inventory-service`. That is its DNS name.

- **Bad:** `localhost:3004`
- **Good:** `http://inventory-service:3004`

2 2. The "Who are you?" Problem (Auth Propagation)

TRAP: Stateless Trust

The **Orders Service** receives a request with a JWT: `Bearer eyJ...`. It needs to know who the user is. **Trap:** Do NOT call the Auth Service database to check the token. That makes Auth a bottleneck.

SOLUTION: Shared Secret Key

Both **Auth Service** and **Orders Service** must have the EXACT SAME `JWT_SECRET` in their `.env` files.

1. **Auth** signs the token with "SecretKey123".
2. **Orders** verifies the signature with "SecretKey123".
3. If it matches, Orders **trusts** the payload (`userId: 1`) without asking the DB.

3 3. The "Silent Failure" (Async Debugging)

TRAP: "It worked... but nothing happened"

You send an event to RabbitMQ. The Orders API helper says "Success". But the Stock never updates. No error appears in the Orders terminal. **Why?** The consumer failed silently in the background.

SOLUTION: Logs are Life

You must check the logs of the **Inventory Container**, not just the Orders Container. `docker logs youshop_inventory_1` Always put a `try/catch` with a `console.error` in your Event Handlers, otherwise NestJS might swallow the error.