

# Arquitectura de Desplegament:

## Projecte Fittoni

El desplegament del projecte Fittoni es fonamenta en una **arquitectura de Proxy Invers** que garanteix seguretat, escalabilitat i un cicle de desenvolupament àgil. Utilitzem **Nginx** com a gestor de trànsit i **Docker** per a l'aïllament i la portabilitat dels serveis d'aplicació i base de dades.

### 1. Diagrama de Flux de Dades

Aquest diagrama il·lustra el recorregut d'una petició des del navegador de l'usuari fins als serveis interns del servidor.

Resum del Flux de Peticions:

Pas	Origen/Destí	Protocol/Port	Component	Funció Principal
1.	Usuari \$\\rightarrow\$ Servidor	HTTPS / WSS (443)	NGINX	Punt d'entrada, seguretat SSL.
2.	NGINX \$\\rightarrow\$ Backend	Proxy Pass (8080)	Docker Compose	Redirigeix trànsit d'API/Sockets.
3.	Backend \$\\rightarrow\$ Base de Dades	Xarxa interna Docker	Contenidors	Lògica de negoci i persistència.

### 2. Els Components Clau

#### A. El Frontend (Vue.js)

- Descripció:** Interfície visual del projecte (HTML, CSS, JavaScript).
- Procés de Desplegament:**
  - S'executa `npm run build` localment per generar el codi "cuinat" a la carpeta dist.
  - La carpeta dist es transfereix al servidor a la ruta `/var/www/fittoni`.
  - Nginx** serveix aquests arxius estàtics directament, cosa que assegura una velocitat

de càrrega òptima.

## B. El Backend (Node.js + Python)

- **Descripció:** El **cervell** de l'aplicació. Gestiona l'API REST (Express), el Servidor WebSocket (ws) i la lògica pesada d'IA (TensorFlow/Python).
- **Contenidorització:**
  - S'executa dins d'un **Contenidor Docker**.
  - Utilitza un Dockerfile personalitzat per instal·lar l'entorn dual: Node.js v20 i Python 3/pip (essencial per a llibreries d'IA com TensorFlow/Keras).
  - S'exposa internament dins del contenidor al port 8080.

## C. La Base de Dades (MySQL)

- **Descripció:** On s'emmagatzemen les dades persistents (usuaris, sessions d'entrenament, repeticions, etc.).
- **Contenidorització:**
  - Definida com un altre servei a docker-compose.yml.
  - **Persistència de Dades:** Utilitza un **volum Docker** (mysql\_data) que mapeja l'emmagatzematge físic al servidor, garantint que les dades no s'esborrin fins i tot si el contenidor es reinicia o s'actualitza.

## D. Nginx (La Peça Clau)

Nginx actua com a **Proxy Invers** i és el component d'infraestructura més crític. La seva configuració (/etc/nginx/conf.d/fittoni...conf) realitza tres funcions essencials:

1. **Seguretat SSL (HTTPS):** Gestiona els certificats de **Let's Encrypt**, xifrant tot el trànsit i proporcionant el "cadenat verd" a l'usuari.
2. **Servidor de Contingut Estàtic:** Si la petició és per a un arxiu (com index.html, una imatge o un arxiu JS), Nginx el serveix directament des de /var/www/fittoni.
3. **Redirecció de Trànsit (Proxying):** Si la petició és per a l'API (/api/\*) o el WebSocket (/ws/\*), Nginx s'adona que no és un arxiu i "passa" la petició al contenidor Docker del *backend* al port 8080.

## 3. Solució al Problema del WebSocket

El repte més gran del desplegament va ser assegurar que les peticions de **WebSocket** (WSS) poguessin travessar correctament Nginx per arribar al *backend*, mantenint la connexió oberta a llarg termini.

El problema era que Nginx no gestionava correctament la petició d'"Upgrade" del protocol.

La solució es va implementar mitjançant la directiva `try_files` a la *location* principal i la *location* anomenada `@backend`, afegint les capçaleres específiques de WebSocket.

## Configuració Nginx Rellevant

```
server {  
    # ... Configuració de certificats SSL i domini ...  
  
    location / {  
        # Intenta servir l'arxiu estàtic (frontend)  
        # Si no el troba, passa el control al bloc @backend (serveis dinàmics)  
        try_files $uri $uri/ @backend;  
    }  
  
    # Bloc de proxy per al backend i WebSockets  
    location @backend {  
        proxy_pass [http://127.0.0.1:8080](http://127.0.0.1:8080);  
  
        # 🗝 Directives CRÍTIQUES per a WebSocket:  
        proxy_http_version 1.1;  
        proxy_set_header Upgrade $http_upgrade;    # Notifica que és una petició de canvi de  
protocol  
        proxy_set_header Connection "upgrade";    # Manté el canal de comunicació obert  
        proxy_read_timeout 86400;                # Configura un timeout llarg (24h) per evitar  
tancaments prematurs  
    }  
}
```

## 4. Resum del Flux de Treball (Dia a Dia)

Tasca	Arxius afectats	Comanda/Acció
<b>Actualitzar Frontend</b>	HTML, CSS, JS	1. npm run build (local) 2. Pujar dist a /var/www/fittoni
<b>Actualitzar Backend/Lògica</b>	Arxius .js/.py	1. Pujar arxius al servidor. 2. Reiniciar Docker: docker-compose up -d --build
<b>Actualitzar Nginx</b>	/etc/nginx/conf.d/fittoni...conf	sudo nginx -t (test) i sudo systemctl reload nginx (reiniciar)