

# [DOT & DOT] Guía de Despliegue

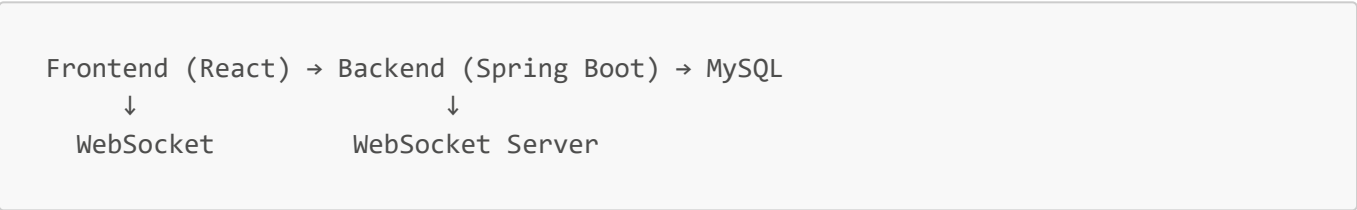
Este documento describe el proceso completo de despliegue del ecosistema DOT & DOT, tanto para entornos de desarrollo como de producción.

## 1. Arquitectura del Sistema

El sistema DOT & DOT se compone de tres componentes principales:

- 1. **Backend** - API REST con Spring Boot (Puerto 5000)
- 2. **Frontend** - Aplicación React con Vite (Puerto 3000 dev / 80 prod)
- 3. **Base de Datos** - MySQL 8.0 (Puerto 3306)

### Comunicación entre Componentes



## 2. Requisitos del Sistema

### 2.1. Hardware Mínimo

#### Desarrollo:

- CPU: 4 cores
- RAM: 8 GB
- Disco: 20 GB libres
- Conexión a Internet

#### Producción:

- CPU: 4 cores (recomendado 8)
- RAM: 16 GB (recomendado 32 GB)
- Disco: 50 GB libres (SSD recomendado)
- Conexión a Internet estable

### 2.2. Sistema Operativo

- Windows 10/11 (64-bit)
- Linux (Ubuntu 20.04+, Debian 11+, CentOS 8+)
- macOS 11+ (Big Sur o superior)

## 3. Aplicaciones Necesarias

---

### 3.1. Aplicaciones Base

#### Git

- **Versión:** 2.40+
- **Descarga:** <https://git-scm.com/downloads>
- **Verificación:** `git --version`
- **Uso:** Control de versiones del código fuente

#### Docker Desktop

- **Versión:** 24.0+
- **Descarga:** <https://www.docker.com/products/docker-desktop>
- **Verificación:** `docker --version` y `docker compose version`
- **Uso:** Contenedores para MySQL y despliegue en producción
- **Configuración recomendada:**
  - Memory: 4 GB (mínimo)
  - CPUs: 4
  - Swap: 2 GB
  - Enable WSL 2 (Windows)

#### MySQL 8.0

- **Versión:** 8.0.30+
- **Descarga:** <https://dev.mysql.com/downloads/installer/>
- **Verificación:** `mysql --version`
- **Uso:** Base de datos relacional
- **Configuración:**
  - Port: 3306
  - Root password: (configurar)
  - Character set: utf8mb4
  - Collation: utf8mb4\_unicode\_ci

## 3.2. IDEs Recomendados

### IntelliJ IDEA (Backend)

- **Versión:** 2024.1+ (Community o Ultimate)
- **Descarga:** <https://www.jetbrains.com/idea/download/>
- **Plugins recomendados:**
  - Spring Boot
  - Lombok
  - JPA Buddy
  - Database Navigator

### Visual Studio Code (Frontend)

- **Versión:** 1.85+
- **Descarga:** <https://code.visualstudio.com/>
- **Extensiones recomendadas:**
  - ESLint
  - Prettier
  - ES7+ React/Redux/React-Native snippets
  - Auto Rename Tag
  - GitLens
  - REST Client

## 4. Development Kits

---

### 4.1. Java Development Kit (JDK)

#### OpenJDK 21

- **Versión:** 21 LTS
- **Descarga:** <https://adoptium.net/>
- **Instalación:**

```
# Windows (con Chocolatey)
choco install openjdk21

# Linux (Ubuntu/Debian)
sudo apt update
sudo apt install openjdk-21-jdk

# macOS (con Homebrew)
brew install openjdk@21
```

- **Verificación:**

```
java -version
javac -version
```

- **Variables de entorno:**

```
# Windows
JAVA_HOME=C:\Program Files\Eclipse Adoptium\jdk-21.0.1.12-hotspot
PATH=%JAVA_HOME%\bin;%PATH%

# Linux/macOS
export JAVA_HOME=/usr/lib/jvm/java-21-openjdk
export PATH=$JAVA_HOME/bin:$PATH
```

## Apache Maven

- **Versión:** 3.9+
- **Descarga:** <https://maven.apache.org/download.cgi>
- **Instalación:**

```
# Windows (con Chocolatey)
choco install maven

# Linux (Ubuntu/Debian)
sudo apt install maven

# macOS (con Homebrew)
brew install maven
```

- **Verificación:**

```
mvn -version
```

- **Configuración** (~/.m2/settings.xml):

```
<settings>
  <localRepository>${user.home}/.m2/repository</localRepository>
  <mirrors>
    <mirror>
      <id>central</id>
      <url>https://repo.maven.apache.org/maven2</url>
      <mirrorOf>central</mirrorOf>
    </mirror>
  </mirrors>
</settings>
```

## 4.2. Node.js y npm

### Node.js

- **Versión:** 20 LTS (20.10.0+)
- **Descarga:** <https://nodejs.org/>
- **Instalación:**

```
# Windows (con Chocolatey)
choco install nodejs-lts

# Linux (Ubuntu/Debian)
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
sudo apt-get install -y nodejs

# macOS (con Homebrew)
brew install node@20
```

- **Verificación:**

```
node --version
npm --version
```

## 5. Configuración del Entorno de Desarrollo

---

### 5.1. Clonar el Repositorio

```
# Clonar repositorio
git clone https://github.com/hugomorunocastelar/TFC-Hugo-Moruno.git
cd TFC-Hugo-Moruno
```

### 5.2. Configuración de MySQL

#### MySQL con Docker

```
# Navegar a la carpeta de microservicios
cd microservicios/mysql

# Iniciar MySQL en contenedor
docker compose up -d

# Verificar que está corriendo
docker ps

# Ver logs
docker compose logs -f
```

## 5.3. Configuración del Backend

### Configurar Variables de Entorno

Editar `aplicaciones/dot&dot-server/src/main/resources/application.yml`:

```
spring:
  datasource:
    url: jdbc:mysql://localhost:3306/dotanddot?useSSL=false&serverTimezone=UTC
    username: dotanddot_user
    password: password
    driver-class-name: com.mysql.cj.jdbc.Driver

  jpa:
    hibernate:
      ddl-auto: update
    show-sql: true
    properties:
      hibernate:
        dialect: org.hibernate.dialect.MySQL8Dialect
        format_sql: true

  mail:
    host: smtp.gmail.com
    port: 587
    username: your-email@gmail.com
    password: your-app-password
    properties:
      mail:
        smtp:
          auth: true
          starttls:
            enable: true

  jwt:
    secret: your-secret-key-here-must-be-at-least-256-bits
    expiration: 86400000 # 24 horas

  server:
    port: 8080
```

## Instalar Dependencias y Compilar

```
# Navegar al directorio del backend
cd aplicaciones/dot&dot-server

# Limpiar y compilar con Maven
mvn clean install

# O usar el wrapper incluido (Windows)
mvnw.cmd clean install

# O usar el wrapper incluido (Linux/macOS)
./mvnw clean install
```

## Ejecutar el Backend

### Opción 1: Desde línea de comandos

```
# Ejecutar con Maven
mvn spring-boot:run

# O con el JAR generado
java -jar target/dotserver.jar
```

### Opción 2: Desde IntelliJ IDEA

1. Abrir el proyecto en IntelliJ
2. Esperar a que Maven descargue dependencias
3. Localizar la clase `DotServerApplication.java`
4. Click derecho → Run 'DotServerApplication'

### Verificar que está corriendo:

```
# Probar endpoint de health
curl http://localhost:5000/health

# Respuesta esperada: OK{1}
```

## 5.4. Configuración del Frontend

### Instalar Dependencias

```
# Navegar al directorio del frontend
cd aplicaciones/dotanddot-web

# Instalar dependencias con npm
npm install

# O con pnpm (más rápido)
pnpm install
```

### Configurar Variables de Entorno

Crear archivo `.env.development`:

```
VITE_BACK_IP=http://localhost:5000
```

Crear archivo `.env.production`:

```
VITE_BACK_IP=https://api.dotanddot.com
```

## Ejecutar el Frontend en Desarrollo

```
# Iniciar servidor de desarrollo
npm run dev

# O con pnpm
pnpm dev
```

### Verificar que está corriendo:

- Abrir navegador en: <http://localhost:3000>
- Deberías ver la pantalla de login de DOT & DOT

## Compilar para Producción

```
# Compilar aplicación
npm run build

# Vista previa de la compilación
npm run preview
```

Los archivos compilados estarán en la carpeta **dist/**.

## 6. Orden de Inicio del Sistema

---

### 6.1. Entorno de Desarrollo

Para iniciar el sistema completo en modo desarrollo, seguir este orden:

#### 1. Iniciar MySQL

```
# Si usas Docker
cd microservicios/mysql
docker compose up -d

# Si usas MySQL local, verificar que está corriendo
# Windows: Servicios → MySQL80
# Linux: sudo systemctl status mysql
# macOS: brew services list
```

#### 2. Iniciar Backend

```
cd aplicaciones/dot&dot-server

# Con Maven
mvn spring-boot:run

# O con IntelliJ IDEA: Run 'DotServerApplication'
```

**Esperar mensaje:**

```
Started DotServerApplication in X.XXX seconds
```

#### 3. Iniciar Frontend

```
cd aplicaciones/dotanddot-web

# Con npm
npm run dev
```