Guía de Preparación de Docker para Principiantes

Tabla de Contenidos

Introducción

Esta guía te enseñará paso a paso cómo configurar las **variables de entorno** para que tu proyecto funcione correctamente con Docker. Las variables de entorno son como "configuraciones secretas" que tu aplicación necesita para funcionar (API keys, contraseñas, URLs, etc.).

¿Qué es Docker? Docker es una herramienta que empaqueta tu aplicación en "contenedores" que funcionan igual en cualquier computadora, sin importar el sistema operativo.

¿Por qué archivos `.env.local`? Los archivos `.env.local` guardan información sensible (como contraseñas y API keys) que NO debe subirse a GitHub por seguridad.

Requisitos Previos

1. Instalar Docker Desktop

Windows / Mac:

- Descarga: https://www.docker.com/products/docker-desktop/
- Instala Docker Desktop y asegúrate de que esté corriendo (verás el icono de Docker en tu barra de tareas)

Linux:

- Guía: https://docs.docker.com/engine/install/

2. Verificar Instalación

Abre tu terminal (PowerShell en Windows, Terminal en Mac/Linux) y ejecuta:

```
docker --version
docker-compose --version
```

Deberías ver algo como:

```
Docker version 24.0.0
Docker Compose version v2.20.0
```

✅ Si ves versiones, ¡estás listo! ❌ Si no, reinicia Docker Desktop o tu computadora.

Estructura del Proyecto

Tu proyecto tiene esta estructura:

```
proyecto/
├── backend/
│   ├── .env.local          ← Configuración del backend
│   ├── .env.example        ← Plantilla de ejemplo
│   ├── Dockerfile
│   └── ... (código backend)
├── frontend/
│   ├── .env.local          ← Configuración del frontend
│   ├── .env.example        ← Plantilla de ejemplo
│   ├── Dockerfile
│   └── ... (código frontend)
└── docker-compose.yml      ← Orquestador de servicios
```

Configuración del Backend

Paso 1: Crear el archivo `.env.local` en la carpeta `backend`

Si no existe, crea un archivo llamado `.env.local` dentro de la carpeta `backend/`.

En Windows:

```
cd backend
New-Item -Path .env.local -ItemType File
```

En Mac/Linux:

```
cd backend
touch .env.local
```

Paso 2: Copiar la configuración

Abre el archivo `backend/.env.local` con tu editor de código favorito (VS Code, Notepad++, etc.) y pega esto:

```
# Puerto del servidor
PORT=5000

# Conexión a MongoDB
MONGODB_URI=mongodb://mongo:27017/ai-interview

# Secreto para tokens JWT (cámbialo por uno único)
JWT_SECRET=tu_secreto_super_seguro_cambiame_123456

# Entorno de desarrollo
NODE_ENV=development

# API Key de OpenAI (Legacy - ya no se usa)
OPENAI_API_KEY=

# API Key de Google Gemini (IMPORTANTE - consigue una gratis)
GEMINI_API_KEY=TU_API_KEY_DE_GEMINI_AQUI

# Configuración de PayPal
PAYPAL_MODE=sandbox
PAYPAL_CLIENT_ID=tu_paypal_client_id_aqui
PAYPAL_SECRET=tu_paypal_secret_aqui

# URL del frontend
FRONTEND_URL=http://localhost:3000

# Idiomas soportados
SUPPORTED_LANGUAGES=en,es,fr,de,pt,it,ja,zh
```

Paso 3: Configurar las Variables

JWT_SECRET (OBLIGATORIO)

Este es un secreto para proteger las contraseñas de tus usuarios.

Cómo generar uno:

Opción 1 - Online:

- Ir a: https://www.grc.com/passwords.htm
- Copia cualquier cadena de caracteres larga
- Pégala en `JWT_SECRET=`

Opción 2 - Terminal:

```
# En PowerShell (Windows)
[Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes((Get-Random -Maximum 999999999)
```

```
# En Mac/Linux
openssl rand -base64 32
```

Ejemplo:

```
JWT_SECRET=mi_secreto_super_seguro_generado_aleatoriamente_12345
```

GEMINI_API_KEY (OBLIGATORIO para IA)

Esta es la clave para usar la inteligencia artificial de Google Gemini (genera preguntas de entrevista, evalúa respuestas, etc.).

Cómo obtenerla (GRATIS):

1. **Ir a Google AI Studio:**
   - https://aistudio.google.com/app/apikey
2. **Iniciar sesión con tu cuenta de Google**
3. **Hacer clic en "Create API Key"**
   - Selecciona un proyecto existente o crea uno nuevo
   - Google te dará una clave como: `AIzaSy...` (cadena de ~39 caracteres)
4. **Copiar la clave y pegarla:**

```
GEMINI_API_KEY=TU_API_KEY_AQUI_ejemplo_AIzaSy...
```

⚠ IMPORTANTE:

- ✅ La API de Gemini tiene un **plan gratuito generoso**
- ✅ No necesitas tarjeta de crédito para empezar
- ⚠ No compartas esta clave con nadie
- ⚠ No la subas a GitHub

Si no quieres usar IA ahora: Deja la línea vacía:

```
GEMINI_API_KEY=
```

El proyecto funcionará, pero sin las funciones de inteligencia artificial.

PAYPAL_CLIENT_ID y PAYPAL_SECRET (OPCIONAL)

Estos son para pagos con PayPal (suscripciones premium).

Cómo obtenerlos:

1. **Ir a PayPal Developer:**

- https://developer.paypal.com/

2. **Crear una cuenta de desarrollador (gratis)**
3. **Ir a "Dashboard" → "Apps & Credentials"**
   - https://developer.paypal.com/dashboard/applications/sandbox
4. **Crear una aplicación Sandbox:**
   - Haz clic en "Create App"
   - Dale un nombre (ej: "AI Interview App")
   - Te dará un **Client ID** y un **Secret**
5. **Copiar y pegar:**

```
PAYPAL_CLIENT_ID=AbCdEfGhIjKlMnOpQrStUvWxYz123456
PAYPAL_SECRET=EFabcdefGHIjklmnOPQrstuv1234567890
```

Si no vas a usar pagos ahora: Deja las líneas como están:

```
PAYPAL_CLIENT_ID=tu_paypal_client_id_aqui
PAYPAL_SECRET=tu_paypal_secret_aqui
```

✅ Otras Variables (Ya Configuradas)

Estas variables ya están correctas, **NO las cambies** a menos que sepas lo que haces:

- `PORT=5000` → Puerto interno del servidor
- `MONGODB_URI=mongodb://mongo:27017/ai-interview` → Conexión a la base de datos
- `NODE_ENV=development` → Modo de desarrollo
- `PAYPAL_MODE=sandbox` → Modo de prueba de PayPal
- `FRONTEND_URL=http://localhost:3000` → URL del frontend
- `SUPPORTED_LANGUAGES=en,es,fr,de,pt,it,ja,zh` → Idiomas disponibles

Configuración del Frontend

Paso 1: Crear el archivo `.env.local` en la carpeta `frontend`

```
cd frontend
New-Item -Path .env.local -ItemType File  # Windows
# o
touch .env.local  # Mac/Linux
```

Paso 2: Copiar la configuración

Abre `frontend/.env.local` y pega esto:

```
# URL de la API del backend
REACT_APP_API_URL=http://localhost:5001/api

# API Key de Google Gemini (opcional para el frontend)
REACT_APP_GEMINI_API_KEY=TU_API_KEY_DE_GEMINI_AQUI

# Client ID de PayPal (opcional)
REACT_APP_PAYPAL_CLIENT_ID=tu_paypal_client_id_aqui
```

Paso 3: Configurar las Variables

REACT_APP_API_URL (YA ESTÁ CORRECTA)

Esta es la URL donde el frontend se conecta al backend.

NO CAMBIAR - Ya está configurada correctamente:

```
REACT_APP_API_URL=http://localhost:5001/api
```

Nota: En React, todas las variables deben empezar con `REACT_APP_` para que funcionen.

REACT_APP_GEMINI_API_KEY (OPCIONAL)

Usa la **misma clave** que pusiste en el backend:

```
REACT_APP_GEMINI_API_KEY=TU_API_KEY_AQUI_ejemplo_AIzaSy...
```

REACT_APP_PAYPAL_CLIENT_ID (OPCIONAL)

Usa el **mismo Client ID** que pusiste en el backend:

```
REACT_APP_PAYPAL_CLIENT_ID=AbCdEfGhIjKlMnOpQrStUvWxYz123456
```

Cómo Obtener las API Keys

Resumen Rápido

Servicio	¿Es Gratis?	¿Tarjeta de Crédito?	Enlace
Google Gemini	✅ Sí (generoso)	❌ No	aistudio.google.com
PayPal Sandbox	✅ Sí	❌ No	developer.paypal.com
JWT Secret	✅ N/A	❌ N/A	grc.com/passwords

Google Gemini (Para IA)

¿Para qué sirve?

- Generar preguntas de entrevista automáticamente
- Evaluar respuestas con inteligencia artificial
- Transcribir audio a texto
- Dar feedback inteligente a los candidatos

Pasos detallados:

1. Ve a: https://aistudio.google.com/app/apikey
2. Haz clic en "Get API key"
3. Selecciona "Create API key in new project" o usa uno existente
4. Copia la clave (formato: `AIzaSy...`)
5. Pégala en ambos archivos `.env.local`

Plan Gratuito:

- ✅ 60 solicitudes por minuto
- ✅ 1,500 solicitudes por día
- ✅ Más que suficiente para desarrollo y pruebas

Documentación:

- https://ai.google.dev/gemini-api/docs

PayPal Sandbox (Para Pagos)

¿Para qué sirve?

- Probar pagos sin dinero real
- Sistema de suscripciones (Free, Premium, Enterprise)
- Modo "sandbox" = ambiente de prueba

Pasos detallados:

1. Crea una cuenta en: https://developer.paypal.com/
2. Ve a "Dashboard" → "Apps & Credentials"
3. Asegúrate de estar en modo "Sandbox" (esquina superior derecha)
4. Haz clic en "Create App"
5. Dale un nombre a tu app
6. Copia el "Client ID" y el "Secret"
7. Pégalos en ambos archivos `.env.local`

Cuentas de Prueba: PayPal te da automáticamente cuentas de prueba para simular compradores y vendedores.

Documentación:

- https://developer.paypal.com/docs/api/overview/

Iniciar el Proyecto

Una vez configuradas las variables de entorno, sigue estos pasos:

1. Abrir Terminal en la Raíz del Proyecto

```
cd "C:\Users\tu-usuario\Desktop\...\Proyecto_Intermodular_Pablo(ay_diosmio)_y_Sergio"
```

2. Verificar que Docker Desktop esté Corriendo

- En Windows: Busca el icono de Docker en la barra de tareas
- Debe decir "Docker Desktop is running"

3. Construir e Iniciar los Contenedores

```
docker-compose up -d --build
```

¿Qué hace este comando?

- `docker-compose` → Usa el orquestador de contenedores
- `up` → Inicia los servicios
- `-d` → Modo "detached" (en segundo plano)
- `--build` → Reconstruye las imágenes (primera vez o después de cambios)

Salida esperada:

```
[+] Running 4/4
 ✔ Network ai_network          Created
 ✔ Container ai_interview_mongo     Started
 ✔ Container ai_interview_backend   Started
 ✔ Container ai_interview_frontend  Started
```

4. Verificar que Todo Esté Corriendo

```
docker ps
```

Deberías ver 3 contenedores:

- `ai_interview_mongo` (Puerto 27017)
- `ai_interview_backend` (Puerto 5001)
- `ai_interview_frontend` (Puerto 3000)

5. Ver los Logs (Opcional)

```
# Ver logs de todos los servicios
docker-compose logs -f

# Ver logs solo del backend
docker logs -f ai_interview_backend

# Ver logs solo del frontend
docker logs -f ai_interview_frontend
```

Presiona `Ctrl+C` para salir de los logs.

6. Acceder a la Aplicación

Abre tu navegador y ve a:

- **Frontend:** http://localhost:3000
- **Backend API:** http://localhost:5001/api
- **MongoDB:** localhost:27017 (si tienes MongoDB Compass)

Verificar que las Variables se Cargaron

Verificar Backend

```
docker exec ai_interview_backend printenv | findstr GEMINI
```

Deberías ver:

```
GEMINI_API_KEY=TU_API_KEY_AQUI
```

Verificar Frontend

```
docker exec ai_interview_frontend printenv | findstr REACT_APP
```

Deberías ver:

```
REACT_APP_API_URL=http://localhost:5001/api
REACT_APP_GEMINI_API_KEY=TU_API_KEY_AQUI
```

Detener el Proyecto

Cuando termines de trabajar:

```
docker-compose down
```

Esto detiene y elimina los contenedores, pero **conserva tus datos** (usuarios, entrevistas, etc.) en MongoDB.

Para eliminar TODO (incluyendo datos):

```
docker-compose down -v
```

⚠ Cuidado: Esto borra la base de datos.

❓ Solución de Problemas

Problema: "docker: command not found"

Causa: Docker no está instalado o no está en el PATH.

Solución:

1. Instala Docker Desktop: https://www.docker.com/products/docker-desktop/
2. Reinicia tu terminal
3. Reinicia tu computadora si es necesario

Problema: "Cannot connect to the Docker daemon"

Causa: Docker Desktop no está corriendo.

Solución:

1. Abre Docker Desktop
2. Espera a que diga "Docker Desktop is running"
3. Vuelve a intentar

Problema: "Port 5001 is already in use"

Causa: Otro programa está usando el puerto 5001.

Solución Rápida: Cambia el puerto en `docker-compose.yml`:

```
backend:
  ports:
    - "5002:5000"  # Cambiar 5001 por 5002
```

Y actualiza `frontend/.env.local`:

```
REACT_APP_API_URL=http://localhost:5002/api
```

Problema: "GEMINI_API_KEY not set" en los logs

Causa: No configuraste la API key de Gemini.

Solución:

1. Consigue una API key: https://aistudio.google.com/app/apikey
2. Pégala en `backend/.env.local`:
   ```
   GEMINI_API_KEY=TU_API_KEY_AQUI
   ```
3. Reinicia los contenedores:
   ```
   docker-compose restart backend
   ```

Problema: El frontend no se conecta al backend

Causa: La URL en `REACT_APP_API_URL` está mal.

Solución: Verifica en `frontend/.env.local`:

```
REACT_APP_API_URL=http://localhost:5001/api
```

Debe ser exactamente esa URL (con `/api` al final).

Problema: "MongoServerError: connect ECONNREFUSED"

Causa: MongoDB no está corriendo o no puede conectarse.

Solución:

```
# Reiniciar MongoDB
docker-compose restart mongo

# Verificar que esté corriendo
docker ps | findstr mongo

# Ver logs de MongoDB
docker logs ai_interview_mongo
```

Problema: Cambié el `.env.local` pero no se aplican los cambios

Causa: Docker usa valores en caché.

Solución:

```
# Detener todo
docker-compose down

# Reconstruir e iniciar
docker-compose up -d --build
```

Comandos Útiles de Docker

Gestión Básica

```
# Iniciar servicios
docker-compose up -d

# Detener servicios
docker-compose down

# Ver servicios corriendo
docker ps

# Ver todos los contenedores (incluso detenidos)
docker ps -a

# Ver logs en tiempo real
docker-compose logs -f

# Reiniciar un servicio específico
docker-compose restart backend
```

Depuración

```
# Entrar al contenedor del backend
docker exec -it ai_interview_backend sh

# Entrar al contenedor del frontend
docker exec -it ai_interview_frontend sh

# Ver variables de entorno del backend
docker exec ai_interview_backend printenv

# Ver uso de recursos
docker stats
```

Limpieza

```
# Eliminar contenedores detenidos
docker container prune

# Eliminar imágenes sin usar
docker image prune

# Eliminar todo (cuidado)
docker system prune -a

# Eliminar volúmenes (borra datos)
docker volume prune
```

Próximos Pasos

Una vez que tu proyecto esté corriendo:

1. **Crear un usuario de prueba:**

   ```
   node test-create-user.js
   ```

2. **Acceder al frontend:** http://localhost:3000

3. **Iniciar sesión con:**

   - Email: test@example.com
   - Password: Test123456

4. **Explorar las funcionalidades:**

   - Crear entrevistas personalizadas
   - Probar preguntas generadas por IA
   - Grabar respuestas de audio
   - Ver estadísticas

Recursos Adicionales

Documentación Oficial

- **Docker:** https://docs.docker.com/
- **Docker Compose:** https://docs.docker.com/compose/
- **Node.js:** https://nodejs.org/docs/
- **React:** https://react.dev/
- **MongoDB:** https://www.mongodb.com/docs/
- **Google Gemini:** https://ai.google.dev/gemini-api/docs

Tutoriales en Español

- **Docker para Principiantes:** https://www.youtube.com/watch?v=CV_Uf3Dq-EU
- **Variables de Entorno:** https://12factor.net/es/config
- **Docker Compose:** https://docs.docker.com/compose/gettingstarted/

¿Necesitas Ayuda?

Si tienes problemas:

1. **Revisa los logs:**

```
docker-compose logs -f
```

2. **Verifica las variables de entorno:**

```
docker exec ai_interview_backend printenv
```

3. **Reinicia desde cero:**

```
docker-compose down -v
docker-compose up -d --build
```

4. **Consulta la documentación de Docker**

✅ Checklist de Configuración

Antes de iniciar el proyecto, verifica:

- [ ] Docker Desktop instalado y corriendo
- [ ] Archivo `backend/.env.local` creado y configurado
- [ ] Archivo `frontend/.env.local` creado y configurado
- [ ] `JWT_SECRET` generado y configurado
- [ ] `GEMINI_API_KEY` obtenida y configurada
- [ ] PayPal configurado (opcional)
- [ ] Puerto 3000 disponible (frontend)
- [ ] Puerto 5001 disponible (backend)
- [ ] Puerto 27017 disponible (MongoDB)

Si marcaste todas las casillas, ¡estás listo para ejecutar!

```
docker-compose up -d --build
```

¡Listo!

Ahora tienes tu proyecto de entrevistas con IA corriendo completamente en Docker. Las variables de entorno están configuradas de forma segura en archivos `.env.local` que NO se suben a GitHub.

Recuerda:

- ✓ Nunca subas archivos `.env.local` a GitHub
- ✓ Nunca compartas tus API keys
- ✓ Usa `.env.example` como plantilla para otros desarrolladores
- ✓ Cambia `JWT_SECRET` en producción

¡Feliz desarrollo!

Última actualización: 25 de noviembre de 2025 **Versión:** 1.0