

# Guía completa para desplegar una aplicación Flask en AWS EC2

Esta guía detalla paso a paso cómo desplegar una aplicación Flask conectada a Supabase en una instancia EC2 de Amazon Web Services (AWS). Incluye los comandos utilizados, las dificultades reales encontradas y cómo se solucionaron.


---

## 1. Crear cuenta y lanzar una instancia EC2

### ◇ Pasos:

1. Crear cuenta en <https://aws.amazon.com>
2. Ir a EC2 > Lanzar instancia
3. AMI: Ubuntu Server 24.04 LTS (64-bit)
4. Tipo de instancia: t2.micro (gratuita durante 12 meses)
5. Crear par de claves: flask-key.pem
6. Crear nuevo grupo de seguridad con reglas:
  - SSH (TCP 22) desde 0.0.0.0/0
  - HTTP (TCP 80) desde 0.0.0.0/0
  - HTTPS (TCP 443) desde 0.0.0.0/0
  - TCP personalizado 5000 desde 0.0.0.0/0

### Dificultades:


- AWS no dejaba poner puertos manualmente al lanzar la instancia.
  -  Solución: una vez lanzada la instancia, editamos el grupo de seguridad y añadimos las reglas manualmente.
- 

## 2. Conectarse a la instancia con SSH

### ◇ Desde PowerShell (Windows):

```
cd "C:\ruta\donde\guardaste\flask-key.pem"  
ssh -i flask-key.pem ubuntu@3.16.130.202
```

### Dificultades:


- El comando chmod no existe en Windows.
  -  Solución: usar directamente ssh -i en PowerShell, sin aplicar chmod.
-

### 3. Preparar entorno Python en EC2

#### ◇ Comandos:

```
sudo apt update && sudo apt install python3-pip python3-venv git -y
python3 -m venv venv
source venv/bin/activate
```

#### Dificultades:

- El archivo requirements.txt estaba vacío o no actualizado.
-  Solución: crear/actualizar el archivo con:

```
pip freeze > requirements.txt
```

O subir el correcto desde tu máquina local.

---

### 4. Clonar o actualizar proyecto desde GitHub


#### ◇ Opción 1: Clonar proyecto

```
git clone https://github.com/tuusuario/tu-repo.git
cd tu-repo
```

#### ◇ Opción 2: Actualizar cambios

```
cd ~/Trabajo_ADA
git pull origin main
```

#### Dificultades:

- Archivos como ruta.py o .csv no estaban.
-  Solución: subirlos con scp desde tu PC:

```
scp -i flask-key.pem archivo.csv ubuntu@3.16.130.202:~/Trabajo_ADA/
```


---

### 5. Instalar dependencias del proyecto

#### ◇ Ejecutar:

```
source venv/bin/activate
pip install -r requirements.txt
```

#### Dificultades:

- ModuleNotFoundError al correr Flask
-  Solución: instalar librerías faltantes, por ejemplo:

```
pip install python-dotenv flask-cors osmnx requests
```

---

## 6. Ejecutar el servidor Flask

### ◇ Comandos:

```
export FLASK_APP=app.py
sudo /home/ubuntu/Trabajo_ADA/venv/bin/flask run --host=0.0.0.0 --port=80
```

⚠ No uses flask run sin sudo en puerto 80 (restringido).

### ⚠ Dificultades:

- App no se mostraba en navegador.
- ☒ Solución:
  - Verificar IP pública en EC2
  - Abrir navegador: `http://3.16.130.202`
  - Agregar puerto si es diferente, como `:5000`

---

## 7. Ajustar el frontend para producción

### ◇ Reemplazar fetch con ruta relativa

```
// Antes (no funciona en producción):
fetch('http://127.0.0.1:5000/puntos')
```

```
// Después:
fetch('/puntos')
```

☒ Esto permite que funcione tanto local como en el servidor.

---

## 8. Problemas comunes y soluciones

Problema	Solución
No carga /puntos	Usar <code>fetch('/puntos')</code> , no IP local
Error 500 en Flask	Ver logs en terminal, revisar rutas o archivos faltantes
Flask se cierra al cerrar terminal	Usar <code>nohup</code> o montar con <code>Gunicorn + Nginx</code>

---

## 9. Verificar funcionamiento desde cualquier país

- Abre navegador y visita:  
`http://3.16.130.202`

- ✓ La app debe cargarse desde cualquier lugar del mundo.
- 

## Resultado final

- App Flask desplegada correctamente en EC2
  - Accesible vía IP pública
  - Conectada a Supabase
  - Backend con Flask + Python + rutas dinámicas funcionando
- 

¿Quieres que lo dejemos listo con dominio, HTTPS y Gunicorn + Nginx? ¡También te puedo guiar!