

Configuración Paso a Paso - ImageProcessor

FASE 1: VERIFICACIÓN DE HERRAMIENTAS

Paso 1.1: Verificar Node.js

```
bash

# Abrir terminal/cmd y escribir:
node --version
npm --version

# Debe mostrar algo como:
# v18.17.0 o superior
# 9.6.7 o superior
```

Si no está instalado:

- Ir a <https://nodejs.org>
- Descargar versión LTS
- Instalar y reiniciar terminal

Paso 1.2: Verificar Python

```
bash

# En terminal escribir:
python --version
pip --version

# Debe mostrar:
# Python 3.11.0 o superior
# pip 23.0 o superior
```

Si no está instalado:

- Ir a <https://python.org>
 - Descargar Python 3.11+
 - Instalar marcando "Add to PATH"
-

FASE 2: BACKEND PRIMERO

Paso 2.1: Crear estructura backend

```
bash

# Crear carpeta principal
mkdir imageprocessor
cd imageprocessor

# Crear carpeta backend
mkdir backend
cd backend
```

Paso 2.2: Crear entorno virtual Python

```
bash

# Crear entorno virtual
python -m venv venv

# Activar entorno virtual
# En Windows:
venv\Scripts\activate

# En Mac/Linux:
source venv/bin/activate

# Verificar que está activo (debe aparecer (venv) al inicio de la línea)
```

Paso 2.3: Instalar dependencias Python

```
bash

# Instalar una por una y verificar
pip install Flask==2.3.3
pip install Flask-CORS==4.0.0
pip install Pillow==10.0.0
pip install rembg==2.0.50

# Verificar instalación
pip list
```

Paso 2.4: Crear app.py básico

```
python
```

Crear archivo app.py con este contenido:

```
from flask import Flask, request, jsonify
from flask_cors import CORS
import os

app = Flask(__name__)
CORS(app)

@app.route('/api/health', methods=['GET'])
def health_check():
    return jsonify({"status": "Backend funcionando correctamente", "port": 8000})

@app.route('/api/test', methods=['POST'])
def test_connection():
    return jsonify({"message": "Conexión exitosa con backend"})

if __name__ == '__main__':
    print("Iniciando servidor backend en puerto 8000...")
    app.run(debug=True, port=8000, host='0.0.0.0')
```

Paso 2.5: Probar backend

```
bash

# Ejecutar desde carpeta backend con entorno virtual activo
python app.py

# Debe mostrar:
# * Running on all addresses (0.0.0.0)
# * Running on http://127.0.0.1:8000
```

Verificar en navegador:

- Ir a: <http://localhost:8000/api/health>
- Debe mostrar: `{"status": "Backend funcionando correctamente", "port": 8000}`

FASE 3: FRONTEND DESPUÉS

Paso 3.1: Crear proyecto React (nueva terminal)

```
bash
```

Abrir NUEVA terminal

```
cd imageprocessor
```

Crear proyecto frontend con Vite

```
npm create vite@latest frontend -- --template react
```

```
cd frontend
```

Instalar dependencias base

```
npm install
```

Paso 3.2: Instalar librerías específicas

```
bash
```

Instalar una por una

```
npm install react-dropzone@14.2.3
```

```
npm install jszip@3.10.1
```

```
npm install axios@1.5.0
```

```
npm install react-router-dom@6.15.0
```

Verificar instalación

```
npm list
```

Paso 3.3: Configurar Vite

```
javascript
```

// Editar vite.config.js

```
import { defineConfig } from 'vite'
```

```
import react from '@vitejs/plugin-react'
```

```
export default defineConfig({
```

```
  plugins: [react()],
```

```
  server: {
```

```
    port: 3000,
```

```
    proxy: {
```

```
      '/api': {
```

```
        target: 'http://localhost:8000',
```

```
        changeOrigin: true
```

```
      }
```

```
    }
```

```
  }
```

```
})
```

Paso 3.4: Probar frontend

```
bash

# Ejecutar desde carpeta frontend
npm run dev

# Debe mostrar:
# Local: http://localhost:3000/
# press h to show help
```

Verificar en navegador:

- Ir a: <http://localhost:3000>
 - Debe mostrar la página de Vite + React
-

FASE 4: VERIFICACIÓN COMPLETA

Paso 4.1: Comprobar que ambos funcionan

Terminal 1 (Backend):

```
bash

cd imageprocessor/backend
# Activar entorno virtual si no está activo
source venv/bin/activate # Mac/Linux
# venv\Scripts\activate # Windows
python app.py
```

Terminal 2 (Frontend):

```
bash

cd imageprocessor/frontend
npm run dev
```

Paso 4.2: Probar comunicación

- Frontend: <http://localhost:3000>
 - Backend: <http://localhost:8000/api/health>
 - Ambos deben funcionar simultáneamente
-

ESTRUCTURA FINAL CREADA

```
imageprocessor/  
├── backend/  
│   ├── venv/          # Entorno virtual  
│   ├── app.py         # Servidor Flask  
│   └── (otros archivos que agregaremos)  
└── frontend/  
    ├── src/           # Código React  
    ├── package.json   # Dependencias  
    ├── vite.config.js # Configuración  
    └── (otros archivos Vite)
```

Una vez que tengas esto funcionando, confirma:

1. ¿Backend en puerto 8000 responde?
2. ¿Frontend en puerto 3000 carga?
3. ¿Ambos terminales están corriendo sin errores?

Entonces crearemos las páginas que me mostraste en las imágenes.