# Paso 0: Entendimiento de la Solución

## 🛡️ Entendimiento: Arquitectura de Jarvis Web (Cortex)

Jarvis Web es una solución de asistente conversacional local que integra entrada de voz 🎤, procesamiento de lenguaje 🧠 y respuesta hablada 🔊, todo corriendo en infraestructura propia (sin depender de la nube).

## 📈 Visión General

Componentes Principales:

* - Frontend (React): Interfaz web moderna tipo chat + micrófono.
* - Backend (FastAPI): API que conecta audio, texto y Cortex para procesamiento.
* - Servidor de Modelos (Cortex.so): Motor de lenguaje local (ej. LLaMA 3.2, Mistral, Phi-4).
* - Infraestructura: PostgreSQL, Redis, NGINX.
* - Docker Compose: Orquestación de todos los servicios.

## 🧠 Flujo de Trabajo Resumido

[Usuario habla 🎤] → [Frontend (React + JS)] → [Backend FastAPI] → [Whisper / modelo] → [Cortex.so - LLM local] → [edge-tts] → [Frontend] → [Usuario escucha 🔊]

## 📦 Capas de la solución

|  |  |  |
| --- | --- | --- |
| Capa | Tecnología | Rol Principal |
| Frontend | React + JS | Chat, micrófono, visualización de conversación |
| Backend API | FastAPI | Control de flujo, conexión Cortex, Whisper, edge-tts |
| Motor IA | Cortex.so | Generación de respuestas de lenguaje |
| Infraestructura | Docker, NGINX, PostgreSQL, Redis | Seguridad, almacenamiento, comunicación |
| Despliegue | Docker Compose | Levanta todo el ecosistema automáticamente |

## 🧩 Esquema Visual Rápido

[ Navegador Web (Usuario) ]  
 ↓  
[ Frontend (React) ]  
 ↓ API REST  
[ Backend (FastAPI) ]  
 ↙ ↘  
[ Whisper (Voz->Texto)] [Cortex.so (Texto Inteligente)]  
 ↘ ↙  
[ edge-tts (Texto->Voz) ]  
 ↓  
[ Reproduce audio en navegador ]

## 🎯 Objetivo de este Proyecto

- 100% ejecución en infraestructura local 🏠  
- No requiere servicios externos (salvo certificados SSL si quieres HTTPS)  
- Modular, escalable, seguro  
- Rápido y de baja latencia al no depender de OpenAI ni APIs remotas