1. 🧱 **Arquitectura y módulos**
2. 📐 **Estructura de carpetas y archivos**
3. 🚦 **Fases del proyecto**
4. 📋 **Tareas por sprint (para dev junior 4h/día)**
5. 📊 **Métricas y entregables por fase**

# **🧠 VISIÓN GENERAL DEL PROYECTO**

### **🎯 Objetivo general:**

Desarrollar una **extensión de navegador para WhatsApp Web** que permita a usuarios gestionar mejor sus contactos, mensajes y tareas de venta, con funcionalidades similares a Segsmart Web Plus.

## **🔧 1. ARQUITECTURA FUNCIONAL**

scss

CopiarEditar

Frontend (Extensión Chrome)

│

├── UI inyectada en WhatsApp Web (sidebar, modales, paneles)

├── Scripts de control del DOM de WhatsApp

│

Backend (opcional)

│

├── API para autenticación y sincronización

├── Base de datos de usuarios, etiquetas, plantillas

│

Infraestructura

│

├── Chrome Web Store

├── Hosting (Vercel / Firebase)

├── Licenciamiento y autenticación

## **📂 2. ESTRUCTURA DE CARPETAS**

bash

CopiarEditar

/whatsapp-extension

├── manifest.json # Configuración de la extensión

├── background.js # Script de fondo (si es necesario)

├── content.js # Script inyectado en WhatsApp Web

├── sidebar.html # Interfaz de la barra lateral

├── sidebar.css

├── sidebar.js

├── assets/ # Íconos y recursos gráficos

├── services/

│ ├── tagsService.js

│ ├── templatesService.js

│ ├── authService.js

├── api/

│ ├── apiClient.js # Llamadas al backend

├── utils/

│ ├── domUtils.js

│ ├── logger.js

├── locales/

│ ├── es.json

│ ├── pt-br.json

├── popup/ # Interfaz emergente (opcional)

│ ├── popup.html

│ ├── popup.js

│ └── popup.css

## **🧩 3. MÓDULOS FUNCIONALES**

| **Módulo** | **Descripción** |
| --- | --- |
| 🧠 Sidebar | Panel lateral inyectado sobre WhatsApp Web |
| 🏷️ Etiquetas | Crear y asignar etiquetas a chats |
| 📄 Plantillas | Mensajes prediseñados |
| 📊 Dashboard | Visualización de métricas y actividad |
| 🔁 Automatizaciones | (Futuro) Respuestas automáticas o recordatorios |
| 👤 Usuarios | Login, registro, autenticación |
| 🧩 API externa | Sincronización con base de datos remota o CRM |

## **📈 4. FASES DEL PROYECTO**

### **🔹 Fase 1 - Infraestructura del Proyecto**

* Crear estructura base del proyecto.
* Configurar manifest.json.
* Montar extensión básica con botón que inyecte algo en WhatsApp Web.

### **🔹 Fase 2 - Sidebar UI**

* Inyectar barra lateral persistente.
* Navegación entre botones funcionales.
* Crear diseño responsive y limpio.

### **🔹 Fase 3 - Etiquetas & Plantillas (LocalStorage)**

* Crear etiquetas desde el panel.
* Asignar etiquetas a chats abiertos.
* Guardar plantillas de mensaje y enviarlas manualmente.

### **🔹 Fase 4 - Conexión a Backend**

* Crear login y sincronización de datos en la nube.
* Usar Firebase, Supabase o Node.js API + PostgreSQL.
* Vincular usuario con sus etiquetas y plantillas.

### **🔹 Fase 5 - Licenciamiento y publicación**

* Control de acceso por plan.
* Subir extensión a Chrome Web Store (modo privado o listado).
* Preparar tutorial y onboarding.

## **🧪 5. PLAN DE TRABAJO POR SPRINT**

### **🗓 Sprint 1: Fundaciones**

* Configurar proyecto, carpeta, git.
* manifest.json y archivos básicos.
* Crear botón flotante de prueba en WhatsApp Web.

✅ Entrega: extensión instalada que inyecta algo en pantalla.

### **🗓 Sprint 2: UI Sidebar**

* Crear barra lateral con HTML/CSS.
* Inyectarla en WhatsApp Web (posición fija).
* Añadir navegación entre botones.
* Comenzar estructura modular (sidebar.js).

✅ Entrega: barra lateral usable, con navegación básica.

### **🗓 Sprint 3: Etiquetas**

* Crear formulario para añadir etiquetas.
* Guardarlas en LocalStorage.
* Detectar nombre del chat activo.
* Mostrar etiquetas asignadas al chat.

✅ Entrega: sistema funcional de etiquetas locales.

### **🗓 Sprint 4: Plantillas**

* Crear y guardar plantillas en LocalStorage.
* Inyectar plantilla en chat activo con clic.
* Crear interfaz para gestionar plantillas.

✅ Entrega: sistema de respuestas rápidas funcional.

### **🗓 Sprint 5: Login y Backend**

* Integrar Firebase Auth o Supabase.
* Crear backend simple (Firebase Firestore o Supabase DB).
* Sincronizar plantillas y etiquetas por usuario.

✅ Entrega: cada usuario ve sus propios datos en cualquier navegador.

### **🗓 Sprint 6: Publicación y control de licencia**

* Sistema de validación de plan desde backend.
* Subida a Chrome Web Store (modo privado).
* Crear documentación mínima para usuario.

✅ Entrega: sistema distribuible.

## **📊 6. MÉTRICAS DE ÉXITO**

| **Métrica** | **Objetivo** |
| --- | --- |
| Extensión funcional MVP | antes de semana 6 |
| Inyección sin errores en WhatsApp Web | < 1 segundo de carga |
| Acciones básicas por chat (etiquetar, responder) | < 3 clics |
| Tasa de bugs/crashes | < 5% en QA |
| Tiempo de configuración de un nuevo usuario | < 3 minutos |