

# Integración de Chatwoot en CitaPlanner

---

## Resumen Ejecutivo

---









CitaPlanner ahora cuenta con una integración completa de **Chatwoot**, una plataforma de soporte y chat en vivo de código abierto. Esta integración permite a cada tenant (y opcionalmente a cada sucursal) tener su propio canal de comunicación con clientes, con identificación automática de usuarios y atributos personalizados.

**Versión de Integración:** v1.11.0

**Fecha de Implementación:** Noviembre 2024

**Branch:** `feature/chatwoot-integration`

## Características Principales

-  **Multi-tenant:** Cada tenant puede configurar su propia instancia de Chatwoot
-  **Por sucursal:** Soporte opcional para configuración específica por branch
-  **Identificación automática:** Los usuarios autenticados son identificados automáticamente
-  **Personalizable:** Widget configurable con posición, idioma, colores
-  **Atributos personalizados:** Envío de metadata del tenant, rol, sucursal
-  **Seguro:** Configuración por base de datos con validación de permisos
-  **API REST:** Endpoints para gestionar configuraciones
-  **Flexible:** Compatible con Chatwoot Cloud o self-hosted

## ¿Qué es Chatwoot?

---

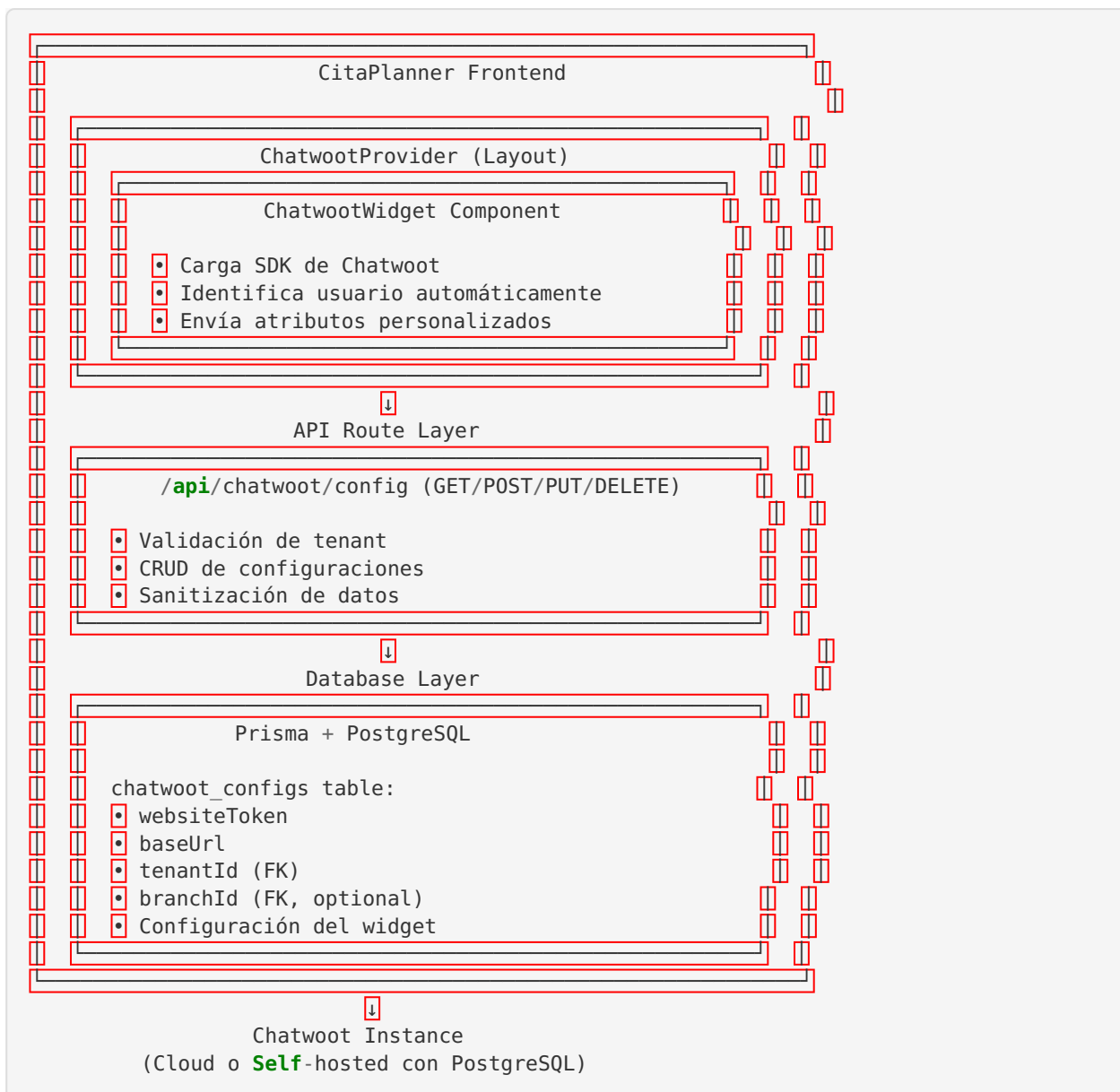
**Chatwoot** (<https://www.chatwoot.com/>) es una alternativa de código abierto a Intercom, Zendesk y otras plataformas de soporte al cliente. Ofrece:

- **Chat en vivo** integrado en tu aplicación web
- **Multi-canal:** WhatsApp, Facebook, Twitter, Email, SMS
- **Automatización** con bots y respuestas automáticas
- **Reportes y analytics** de conversaciones
- **CRM integrado** para gestión de clientes
- **Equipos y colaboración** entre agentes de soporte
- **API completa** para integraciones personalizadas

## ¿Por qué Chatwoot en CitaPlanner?

1. **Soporte directo:** Los clientes pueden obtener ayuda sin salir de la aplicación
  2. **Multi-tenant:** Cada negocio tiene su propio canal de comunicación
  3. **Contexto rico:** Los agentes ven información del tenant, rol, sucursal
  4. **Open Source:** Sin costos de licencia, puede ser self-hosted
  5. **Escalable:** Desde pequeños negocios hasta empresas grandes
-

## Arquitectura de la Integración



### Flujo de Datos

1. **Inicialización:** El `ChatwootProvider` se monta en el layout principal
2. **Fetch Config:** Se obtiene la configuración del tenant desde `/api/chatwoot/config`
3. **Load SDK:** Se carga dinámicamente el SDK de Chatwoot si está habilitado
4. **Identify User:** Se identifica al usuario autenticado con sus datos
5. **Custom Attributes:** Se envían atributos como `tenantId`, `role`, `branchId`
6. **Widget Ready:** El widget de chat aparece en la página

## Configuración Paso a Paso

### 1. Crear Cuenta en Chatwoot

Tienes dos opciones:

## Opción A: Chatwoot Cloud (Recomendado para empezar)

1. Ve a <https://www.chatwoot.com/> (<https://www.chatwoot.com/>)
2. Crea una cuenta gratuita o de pago
3. Crea un “Inbox” de tipo “Website”
4. Obtén el **Website Token**

## Opción B: Self-Hosted (Mayor control)

1. Sigue la [guía de instalación oficial](https://www.chatwoot.com/docs/self-hosted) (<https://www.chatwoot.com/docs/self-hosted>)
2. Usa Docker/Docker Compose o Heroku
3. Configura PostgreSQL, Redis
4. Accede a tu instancia y crea un Inbox
5. Obtén el **Website Token**

## 2. Obtener el Website Token

En tu dashboard de Chatwoot:

1. Ve a **Settings** → **Inboxes**
2. Selecciona tu inbox de Website
3. Ve a la pestaña **Configuration**
4. Copia el **Website Token** (similar a ABC123xyz456 )
5. Copia también la **Base URL** de tu instancia


### Ejemplo de Base URL:

- Chatwoot Cloud: `https://app.chatwoot.com`
- Self-hosted: `https://tu-dominio.com` o `https://chatwoot.tu-empresa.com`

## 3. Configurar Variables de Entorno (Opcional)

Estas variables son opcionales y sirven como configuración por defecto:

```
# .env o variables de entorno en Easypanel
NEXT_PUBLIC_CHATWOOT_WEBSITE_TOKEN=tu_website_token_aqui
NEXT_PUBLIC_CHATWOOT_BASE_URL=https://app.chatwoot.com
```

 **Nota:** La configuración por variables de entorno es global. Para multi-tenant, es mejor usar la base de datos.

## 4. Ejecutar Migración de Prisma

La migración ya está creada. Solo necesitas aplicarla:

```
# En desarrollo local
cd app
npx prisma migrate deploy

# En producción (Easypanel)
# Las migraciones se ejecutan automáticamente en el entrypoint
```

Esto crea la tabla `chatwoot_configs` con:

- `id` : UUID único
- `websiteToken` : Token del inbox de Chatwoot
- `baseUrl` : URL de la instancia de Chatwoot

- `isActive` : Si está habilitado o no
- `isDefault` : Si es la configuración por defecto del tenant
- `position` : Posición del widget (left/right)
- `locale` : Idioma (es, en, etc.)
- `widgetColor` : Color personalizado del widget
- `tenantId` : Relación con el tenant
- `branchId` : Relación opcional con una sucursal

## 5. Configurar por Tenant en Base de Datos

### Usando la API (Recomendado)

#### Crear configuración:

```
POST /api/chatwoot/config
Content-Type: application/json
Authorization: Bearer <token>

{
  "websiteToken": "ABC123xyz456",
  "baseUrl": "https://app.chatwoot.com",
  "isActive": true,
  "position": "right",
  "locale": "es",
  "widgetColor": "#1f93ff"
}
```

#### Obtener configuración:

```
GET /api/chatwoot/config
Authorization: Bearer <token>
```

#### Actualizar configuración:

```
PUT /api/chatwoot/config
Content-Type: application/json
Authorization: Bearer <token>

{
  "isActive": false
}
```

#### Eliminar configuración:

```
DELETE /api/chatwoot/config?configId=<id>
Authorization: Bearer <token>
```

## Directamente en PostgreSQL

```
-- Insertar configuración para un tenant
INSERT INTO chatwoot_configs (
  id,
  "websiteToken",
  "baseUrl",
  "isActive",
  "isDefault",
  position,
  locale,
  "tenantId",
  "createdAt",
  "updatedAt"
) VALUES (
  gen_random_uuid(),
  'ABC123xyz456',
  'https://app.chatwoot.com',
  true,
  true,
  'right',
  'es',
  '<tenant_id_aqui>',
  NOW(),
  NOW()
);
```

## 6. Verificar que Funciona

1. Inicia sesión en CitaPlanner con un usuario del tenant configurado
2. Deberías ver el widget de chat en la esquina inferior derecha
3. Abre el chat y envía un mensaje de prueba
4. Ve a tu dashboard de Chatwoot y verifica que llegó el mensaje
5. Responde desde Chatwoot y verifica que aparece en el widget



## Uso de la API

### Endpoints Disponibles

GET /api/chatwoot/config

Obtiene la configuración de Chatwoot para el tenant actual.

#### Headers requeridos:

Authorization: Bearer <jwt\_token>

#### Response exitoso (200):

```
{
  "success": true,
  "config": {
    "id": "uuid",
    "websiteToken": "ABC123xyz456",
    "baseUrl": "https://app.chatwoot.com",
    "isActive": true,
    "position": "right",
    "locale": "es",
    "widgetColor": "#1f93ff",
    "tenantId": "tenant_uuid",
    "branchId": null
  }
}
```

#### Response sin configuración (200):

```
{
  "success": true,
  "config": null
}
```

#### POST /api/chatwoot/config

Crea una nueva configuración de Chatwoot.

#### Headers requeridos:

```
Authorization: Bearer <jwt_token>
Content-Type: application/json
```

#### Body (ejemplo):

```
{
  "websiteToken": "ABC123xyz456",
  "baseUrl": "https://app.chatwoot.com",
  "isActive": true,
  "isDefault": true,
  "position": "right",
  "locale": "es",
  "widgetColor": "#1f93ff",
  "branchId": "branch_uuid" // opcional
}
```

#### Response exitoso (201):

```
{
  "success": true,
  "config": { ... }
}
```

#### PUT /api/chatwoot/config

Actualiza una configuración existente.

#### Body (ejemplo):

```
{
  "isActive": false,
  "widgetColor": "#ff0000"
}
```

**Response exitoso (200):**

```
{
  "success": true,
  "config": { ... }
}
```

**DELETE** /api/chatwoot/config?configId=<uuid>

Elimina una configuración.

**Response exitoso (200):**

```
{
  "success": true,
  "message": "Configuración eliminada"
}
```

## Validaciones y Errores

- **401 Unauthorized:** No hay sesión o token inválido
- **403 Forbidden:** Usuario no tiene permisos de ADMIN/SUPERADMIN
- **404 Not Found:** Configuración no encontrada
- **400 Bad Request:** Datos inválidos o faltantes
- **500 Internal Server Error:** Error del servidor

## Personalización del Widget

### Opciones de Configuración

El widget puede personalizarse tanto en el backend (base de datos) como en el frontend:

#### En la Base de Datos

```
UPDATE chatwoot_configs
SET
  position = 'left',          -- 'left' o 'right'
  locale = 'en',             -- 'es', 'en', 'fr', etc.
  "widgetColor" = '#ff6b6b'  -- Color hexadecimal
WHERE "tenantId" = '<tenant_id>';
```

## En el Componente (settings)

```
<ChatwootWidget
  config={chatwootConfig}
  settings={{
    position: 'left',
    locale: 'es',
    hideMessageBubble: false,
    type: 'standard',
    darkMode: 'auto',
    launcherTitle: '¿Necesitas ayuda?'
  }}
/>
```

## Posiciones Disponibles

- `right` (por defecto): Esquina inferior derecha
- `left` : Esquina inferior izquierda

## Idiomas Soportados

Chatwoot soporta más de 30 idiomas. Los más comunes:

- `es` : Español
- `en` : Inglés
- `fr` : Francés
- `de` : Alemán
- `pt` : Portugués
- `it` : Italiano
- `ca` : Catalán

## Modos de Color

- `auto` : Se adapta al tema del sistema
- `light` : Siempre modo claro

## Estilos CSS Personalizados

Puedes agregar estilos globales en tu aplicación:

```
/* En tu CSS global */
#chatwoot-widget-holder {
  z-index: 9999 !important;
}

.woot-widget-bubble {
  bottom: 80px !important; /* Ajustar posición vertical */
}
```



## Atributos Personalizados

CitaPlanner envía automáticamente estos atributos a Chatwoot:



## Atributos por Defecto

Atributo	Tipo	Descripción
tenantId	string	UUID del tenant
tenantName	string	Nombre del negocio
role	string	Rol del usuario (ADMIN, USER, etc.)
branchId	string	UUID de la sucursal (opcional)
branchName	string	Nombre de la sucursal (opcional)

## Ver Atributos en Chatwoot

1. Ve a una conversación en el dashboard
2. Panel derecho → **Contact Information**
3. Sección **Custom Attributes**
4. Verás todos los atributos enviados

## Agregar Atributos Personalizados

Puedes enviar atributos adicionales:

```
<ChatwootWidget
  config={chatwootConfig}
  customAttributes={{
    plan: 'premium',
    subscriptionEnd: '2024-12-31',
    totalAppointments: 150
  }}
/>
```

## Usar Atributos en Chatwoot

Los atributos personalizados te permiten:

- **Segmentar conversaciones:** Filtrar por tenant, sucursal, plan
- **Automatizaciones:** Respuestas automáticas basadas en atributos
- **Reportes:** Analytics por tenant o sucursal
- **Macros:** Usar atributos en respuestas predefinidas

---

## Identificación de Usuarios

### Datos Enviados

CitaPlanner identifica automáticamente a los usuarios con:

```
{
  identifier: user.id,           // UUID único del usuario
  name: user.name,             // Nombre completo
  email: user.email,           // Email
  avatar_url: user.image,       // URL del avatar
  phone_number: user.phone     // Teléfono (opcional)
}
```

## Flujo de Identificación

1. Usuario inicia sesión en CitaPlanner
2. ChatwootWidget detecta la sesión
3. Extrae datos del usuario desde NextAuth
4. Llama a `window.$chatwoot.setUser()`
5. Usuario aparece identificado en Chatwoot

## Ventajas de la Identificación

- **Contexto completo:** Los agentes ven quién está hablando
- **Historial:** Todas las conversaciones del usuario en un solo lugar
- **CRM integrado:** Perfil del cliente con datos de CitaPlanner
- **Sin duplicados:** Mismo identifier = mismo contacto

## Usuarios No Autenticados

Si un usuario no está autenticado:

- El widget aún funciona
- Chatwoot crea un contacto anónimo
- Se puede pedir email en la primera interacción



## Troubleshooting Común

### El widget no aparece

Posibles causas:

#### 1. No hay configuración activa

```
sql
-- Verificar en la base de datos
SELECT * FROM chatwoot_configs
WHERE "tenantId" = '<tu_tenant_id>'
AND "isActive" = true;
```

#### 2. Variables de entorno incorrectas

```
bash
# Verificar que las URLs sean correctas (sin trailing slash)
echo $NEXT_PUBLIC_CHATWOOT_BASE_URL
```

#### 3. Error al cargar el SDK

- Abre las DevTools del navegador
- Pestaña Network

- Busca errores al cargar `sdk.js`
- Verifica que la baseUrl sea accesible

#### 4. Bloqueador de ads

- Algunos bloqueadores bloquean Chatwoot
- Prueba desactivando extensiones

## El widget aparece pero no funciona

### 1. Verificar token

```
javascript
// En la consola del navegador
console.log(window.chatwootSettings);
```

### 2. Verificar que el SDK se cargó

```
javascript
console.log(typeof window.$chatwoot); // Debe ser 'object'
```

### 3. Ver errores en consola

- Abre DevTools → Console
- Busca errores rojos relacionados con Chatwoot

## Los usuarios no se identifican

### 1. Verificar que hay sesión

```
javascript
// En componente React
console.log(session);
```

### 2. Verificar datos del usuario

```
javascript
console.log(session?.user);
```

### 3. Verificar que se llamó setUser

- Pon un breakpoint en `ChatwootWidget.tsx`
- Línea donde se llama `window.$chatwoot.setUser()`

## Los atributos no aparecen

### 1. Verificar que se enviaron

```
javascript
// Después de identificar usuario
window.$chatwoot.setCustomAttributes({
  test: 'value'
});
```

### 2. Refrescar conversación en Chatwoot

- A veces tarda unos segundos en actualizar

### 3. Verificar permisos

- Algunos campos pueden requerir permisos especiales en Chatwoot

## Error CORS

Si ves errores de CORS en la consola:

### 1. Verificar la baseUrl

- Debe ser exactamente como está configurada en Chatwoot
- Sin trailing slash

### 2. Configurar en Chatwoot

- Ve a Settings → Inbox → Configuration
- Verifica que el dominio de CitaPlanner esté permitido

## Widget en posición incorrecta

```
/* Ajustar manualmente si es necesario */
.woot-widget-bubble {
  bottom: 20px !important;
  right: 20px !important;
}
```



## Ejemplos de Código

### Uso Básico en Layout

```
// app/layout.tsx o app/(dashboard)/layout.tsx
import { ChatwootProvider } from '@components/chatwoot';

export default function RootLayout({ children }) {
  return (
    <html>
      <body>
        <SessionProvider>
          <ChatwootProvider>
            {children}
          </ChatwootProvider>
        </SessionProvider>
      </body>
    </html>
  );
}
```

## Uso con Configuración Personalizada

```
'use client';

import { useEffect, useState } from 'react';
import { ChatwootWidget } from '@components/chatwoot';

export default function CustomChatwoot() {
  const [config, setConfig] = useState(null);

  useEffect(() => {
    async function loadConfig() {
      const res = await fetch('/api/chatwoot/config');
      const data = await res.json();
      setConfig(data.config);
    }
    loadConfig();
  }, []);

  return (
    <ChatwootWidget
      config={config}
      settings={{
        position: 'left',
        locale: 'es',
        launcherTitle: 'Chatea con nosotros'
      }}
      customAttributes={{
        page: 'dashboard',
        feature: 'appointments'
      }}
    />
  );
}
```

## Control Programático del Widget

```
'use client';

import { useEffect } from 'react';

export default function ChatControls() {
  function openChat() {
    window.$chatwoot?.toggle('open');
  }

  function closeChat() {
    window.$chatwoot?.toggle('close');
  }

  function setLabel(label: string) {
    window.$chatwoot?.setLabel(label);
  }

  return (
    <div>
      <button onClick={openChat}>Abrir Chat</button>
      <button onClick={closeChat}>Cerrar Chat</button>
      <button onClick={() => setLabel('VIP')}>
        Marcar como VIP
      </button>
    </div>
  );
}
```

## Obtener Configuración en Server Component

```
// app/page.tsx (Server Component)
import { getChatwootConfig } from '@lib/chatwoot/server';
import { getServerSession } from 'next-auth';
import { authOptions } from '@app/api/auth/[...nextauth]/route';

export default async function HomePage() {
  const session = await getServerSession(authOptions);

  if (!session?.user?.tenantId) {
    return <div>No hay sesión</div>;
  }

  const chatwootConfig = await getChatwootConfig(
    session.user.tenantId
  );

  return (
    <div>
      <h1>Dashboard</h1>
      {chatwootConfig && (
        <p>Chatwoot está habilitado para tu tenant</p>
      )}
    </div>
  );
}
```

## API Route Personalizada

```
// app/api/my-chatwoot-status/route.ts
import { NextResponse } from 'next/server';
import { getServerSession } from 'next-auth';
import { authOptions } from '@app/api/auth/[...nextauth]/route';
import { getChatwootConfig } from '@lib/chatwoot/server';

export async function GET(req: Request) {
  const session = await getServerSession(authOptions);











  if (!session?.user?.tenantId) {
    return NextResponse.json(
      { error: 'Unauthorized' },
      { status: 401 }
    );
  }

  const config = await getChatwootConfig(session.user.tenantId);

  return NextResponse.json({
    hasConfig: !!config,
    isActive: config?.isActive ?? false,
    position: config?.position ?? 'right',
  });
}
```

## Screenshots y Diagramas

### Flujo de Usuario

1. Usuario  Inicia sesión en CitaPlanner
2. ChatwootProvider  Carga configuración del tenant
3. ChatwootWidget  Carga SDK de Chatwoot
4. Widget  Aparece en la página
5. Usuario  Click en el widget
6. Identificación  Datos del usuario enviados a Chatwoot
7. Conversación  Mensaje enviado
8. Chatwoot  Agente recibe **notificación**
9. Agente  Responde al usuario
10. Widget  Muestra respuesta del agente

## Estructura de Archivos

```

app/
├── lib/
│   ├── chatwoot/
│   │   ├── types.ts           # Tipos TypeScript
│   │   ├── config.ts         # Configuración y utilidades
│   │   ├── server.ts         # Funciones server-side
│   │   └── index.ts          # Exportaciones
│   └── components/
│       ├── chatwoot/
│       │   ├── ChatwootWidget.tsx # Componente del widget
│       │   ├── ChatwootProvider.tsx # Provider para layout
│       │   └── index.ts          # Exportaciones
│       └── api/
│           ├── chatwoot/
│           │   └── config/
│           │       └── route.ts    # API endpoints (GET/POST/PUT/DELETE)

```



## Deployment en Producción

### Variables de Entorno en Easypanel

1. Ve a tu proyecto en Easypanel
2. Sección **Environment Variables**
3. Agrega (opcional, si quieres config global):

```
NEXT_PUBLIC_CHATWOOT_WEBSITE_TOKEN=abc123
```

```
NEXT_PUBLIC_CHATWOOT_BASE_URL=https://app.chatwoot.com
```

4. **Deploy** para aplicar cambios

### Migración en Easypanel

Las migraciones se ejecutan automáticamente en el `docker-entrypoint.sh` :

```

# Verificar que la migración se aplicó
# Conecta a la base de datos y ejecuta:
SELECT table_name
FROM information_schema.tables
WHERE table_name = 'chatwoot_configs';

```



## Configurar Primer Tenant

```
# Usando psql en Easypanel
psql $DATABASE_URL

-- Insertar configuración
INSERT INTO chatwoot_configs (
  id, "websiteToken", "baseUrl", "isActive", "isDefault",
  position, locale, "tenantId", "createdAt", "updatedAt"
) VALUES (
  gen_random_uuid(),
  'TU_WEBSITE_TOKEN',
  'https://app.chatwoot.com',
  true,
  true,
  'right',
  'es',
  (SELECT id FROM tenants LIMIT 1),
  NOW(),
  NOW()
);
```

## Verificar en Producción

1. Abre tu aplicación en producción
2. Inicia sesión con un usuario del tenant
3. Verifica que aparece el widget
4. Envía un mensaje de prueba
5. Revisa en Chatwoot que llegó el mensaje



## Seguridad y Mejores Prácticas



### Recomendaciones

1. **Nunca expongas tokens sensibles**
  - Los website tokens son públicos (van al frontend)
  - NO uses API tokens de Chatwoot en el frontend
2. **Valida permisos en el backend**
  - Solo ADMINS pueden crear/editar configuraciones
  - Usuarios normales solo pueden ver la config de su tenant
3. **Sanitiza URLs**
  - Usa `sanitizeBaseUrl()` para limpiar URLs
  - Previene inyección de scripts
4. **Usa HTTPS siempre**
  - Chatwoot requiere HTTPS en producción
  - No funcionará con HTTP
5. **Limita tasa de requests**
  - Implementa rate limiting en `/api/chatwoot/config`
  - Previene abuso de la API






## 6. Monitorea el uso

- Revisa logs de Chatwoot regularmente
- Detecta spam o abuso

## 7. Configura CORS correctamente

- En Chatwoot, lista dominios permitidos
- Evita wildcards (\*)

## Cosas a Evitar

-  No uses el mismo inbox para todos los tenants
-  No expongas API keys de Chatwoot en el código
-  No permitas que usuarios normales cambien la configuración
-  No olvides validar y sanitizar inputs
-  No uses HTTP en producción

## Referencias y Recursos

### Documentación Oficial de Chatwoot

- **Documentación general:** <https://www.chatwoot.com/docs>
- **API Reference:** <https://www.chatwoot.com/developers/api>
- **SDK JavaScript:** <https://www.chatwoot.com/docs/product/channels/live-chat/sdk/setup>
- **Self-hosted guide:** <https://www.chatwoot.com/docs/self-hosted>
- **Integraciones:** <https://www.chatwoot.com/docs/product/integrations>

### Repositorios de Código

- **Chatwoot GitHub:** <https://github.com/chatwoot/chatwoot>
- **SDK JavaScript:** <https://github.com/chatwoot/chatwoot-javascript-sdk>

### Comunidad y Soporte

- **Discord de Chatwoot:** <https://discord.gg/cJXdrwS>
- **GitHub Discussions:** <https://github.com/chatwoot/chatwoot/discussions>
- **Stack Overflow:** Tag `chatwoot`

### Archivos de CitaPlanner

- `app/lib/chatwoot/types.ts` - Tipos TypeScript
- `app/lib/chatwoot/config.ts` - Configuración
- `app/lib/chatwoot/server.ts` - Funciones servidor
- `app/components/chatwoot/ChatwootWidget.tsx` - Widget
- `app/components/chatwoot/ChatwootProvider.tsx` - Provider
- `app/api/chatwoot/config/route.ts` - API endpoints
- `app/prisma/schema.prisma` - Modelo de datos
- `app/prisma/migrations/.../migration.sql` - Migración

## Preguntas Frecuentes (FAQ)

### ¿Chatwoot es gratuito?

Sí, Chatwoot es open source y gratuito si lo hosteas tú mismo. También tienen planes de pago en la nube:

- **Community (Cloud):** Gratuito hasta ciertos límites
- **Startup:** ~\$19/mes
- **Business:** ~\$49/mes
- **Self-hosted:** Gratis (solo pagas el hosting)

### ¿Puedo tener múltiples inboxes?

Sí, cada tenant puede tener su propio inbox en Chatwoot. También puedes:

- Tener un inbox por sucursal
- Tener inboxes por departamento (ventas, soporte)
- Conectar WhatsApp, Facebook, etc.

### ¿Funciona en mobile?

Sí, el widget de Chatwoot es responsive y funciona perfectamente en móviles.

### ¿Puedo personalizar los colores?

Sí, usa el campo `widgetColor` en la configuración:

```
UPDATE chatwoot_configs
SET "widgetColor" = '#FF5733'
WHERE id = '<config_id>;'
```

### ¿Cómo agrego más agentes?

En el dashboard de Chatwoot:

1. Ve a Settings → Agents
2. Click en "Add Agent"
3. Invita por email
4. Asigna al inbox correspondiente

### ¿Puedo integrar WhatsApp?

Sí, Chatwoot soporta WhatsApp Business API. Necesitas:

1. Cuenta de WhatsApp Business
2. Proveedor de API (Twilio, 360Dialog, etc.)
3. Configurar en Chatwoot

### ¿Cómo desactivo el widget?

```
UPDATE chatwoot_configs
SET "isActive" = false
WHERE "tenantId" = '<tenant_id>;'
```

O via API:

```
PUT /api/chatwoot/config
{ "isActive": false }
```

## ¿Los mensajes se guardan?

Sí, todas las conversaciones se guardan en la base de datos de Chatwoot. Puedes:

- Ver historial completo
- Buscar conversaciones antiguas
- Exportar datos
- Generar reportes

## ¿Necesito configuración por tenant?

No es obligatorio. Puedes:

- **Opción 1:** Variables de entorno globales (todos los tenants usan el mismo inbox)
- **Opción 2:** Configuración por tenant en DB (cada tenant su inbox)
- **Opción 3:** Híbrido (global por defecto + override por tenant)

## ¿Cómo migro de otro sistema de chat?

1. Exporta datos del sistema anterior
2. Usa la API de Chatwoot para importar contactos
3. Configura Chatwoot en CitaPlanner
4. Desactiva el sistema anterior
5. Notifica a tus usuarios



## Próximos Pasos y Mejoras Futuras

### Mejoras Planeadas

- [ ] **Panel de administración:** UI en CitaPlanner para gestionar configuraciones
- [ ] **Multi-idioma avanzado:** Detección automática del idioma del usuario
- [ ] **Webhooks:** Escuchar eventos de Chatwoot (mensaje nuevo, conversación cerrada)
- [ ] **Integración con WhatsApp:** Configurar desde CitaPlanner
- [ ] **Analytics:** Dashboard con métricas de conversaciones por tenant
- [ ] **Bots automáticos:** Configurar respuestas automáticas desde CitaPlanner
- [ ] **Notificaciones:** Alertas en CitaPlanner cuando hay mensaje nuevo
- [ ] **Chat interno:** Comunicación entre sucursales usando Chatwoot

### Contribuciones

Si quieres contribuir a mejorar la integración:

1. Crea un branch desde `develop`
2. Implementa tu mejora
3. Escribe tests
4. Crea un PR con descripción detallada
5. Etiqueta con `enhancement` y `chatwoot`

## Soporte

---

Si tienes problemas con la integración:

1. **Revisa esta documentación** y el troubleshooting
2. **Verifica logs** en DevTools del navegador
3. **Consulta logs del servidor** (Easypanel o Docker)
4. **Revisa la base de datos** ( `chatwoot_configs` table)
5. **Contacta al equipo** si el problema persiste

### Logs Útiles

```
# Logs del contenedor en Easypanel
docker logs <container_name> | grep -i chatwoot

# Logs de PostgreSQL
SELECT * FROM chatwoot_configs WHERE "tenantId" = '<tenant_id>';

# Logs del navegador (DevTools)
# Console → Busca errores de Chatwoot
```

---

## Checklist de Implementación

---

Para nuevos tenants que quieran activar Chatwoot:

- ☐ Crear cuenta en Chatwoot (Cloud o self-hosted)
- ☐ Crear inbox de tipo “Website”
- ☐ Obtener website token
- ☐ Insertar configuración en `chatwoot_configs` table
- ☐ Verificar que `isActive = true`
- ☐ Probar en desarrollo local
- ☐ Probar en producción
- ☐ Capacitar a agentes de soporte
- ☐ Configurar horarios de atención
- ☐ Configurar respuestas automáticas (opcional)
- ☐ Monitorear primeras conversaciones
- ☐ Recopilar feedback de usuarios

---

**Integración desarrollada con  para CitaPlanner**

**Versión:** 1.11.0

**Última actualización:** Noviembre 2024