Documento de Propuesta de Diseño de Software I, II y II

**Componente de mensajería**

**Autores**: Jaider Martinez Paternina - María Camila Salgado Montiel - Luis Manuel Cogollo Ayazo - Jhoan Sebastián Fernández Julio.



Descripción del software

Este documento detalla la planificación y desarrollo de un componente de mensajería innovador que incorpora un chat-bot basado en inteligencia artificial y un servicio de soporte con expertos. La creación de este componente tiene como objetivo principal mejorar la experiencia del usuario al proporcionar respuestas rápidas y efectivas a sus preguntas.

[**Etapa 1 Diseño de la Aplicación y Análisis de Requisitos 6**](#_gjdgxs)

[**1.**](#_30j0zll) **Introducción 6**

[Propósito del Documento 6](#_1fob9te)

[Alcance del Proyecto Módulo de Pizarra Compartida 8](#_3znysh7)

[Definiciones y Acrónimos 8](#_3mzq4wv)

[**2.**](#_2et92p0) **Descripción General 10**

[Objetivos del Sistema 10](#_tyjcwt)

[Funcionalidad General 10](#_3dy6vkm)

[Usuarios del Sistema 11](#_1t3h5sf)

[Restricciones 12](#_2250f4o)

[**3.**](#_4d34og8) **Requisitos Funcionales 12**

[Casos de Uso 13](#_2s8eyo1)

[Diagramas de Flujo de Casos de Uso 14](#_17dp8vu)

[Descripción detallada de cada caso de uso 14](#_3rdcrjn)

[Prioridad de Requerimientos 16](#_26in1rg)

[**4.**](#_35nkun2) **Requisitos No Funcionales 18**

[Requisitos de Desempeño 18](#_1ksv4uv)

[Requisitos de Seguridad 19](#_44sinio)

[Requisitos de Usabilidad 20](#_2jxsxqh)

[Requisitos de Escalabilidad 20](#_z337ya)

[**5.**](#_3j2qqm3) **Modelado E/R 21**

[Diagrama de Entidad-Relación 21](#_1y810tw)

[Diagrama Relacional 22](#_4i7ojhp)

[Script de modelo relacional 23](#_2xcytpi)

[Descripción de Entidades y Relaciones 24](#_1ci93xb)

[Reglas de Integridad Referencial 25](#_3whwml4)

[Colecciones (NoSLQ) 28](#_2bn6wsx)

[**6.**](#_qsh70q) **Anexos 29**

[Diagramas Adicionales 29](#_3as4poj)

[Referencias 29](#_1pxezwc)

[**Etapa 2: Persistencia de Datos con Backend 30**](#_49x2ik5)

[**7.**](#_2p2csry) **Introducción 30**

[Propósito de la Etapa 30](#_147n2zr)

[Alcance de la Etapa 30](#_3o7alnk)

[Definiciones y Acrónimos 30](#_23ckvvd)

[**8.**](#_ihv636) **Diseño de la Arquitectura de Backend 30**

[Descripción de la Arquitectura Propuesta 30](#_32hioqz)

[Componentes del Backend 30](#_1hmsyys)

[Diagramas de Arquitectura 30](#_41mghml)

[**9.**](#_2grqrue) **Elección de la Base de Datos 30**

[Evaluación de Opciones (SQL o NoSQL) 31](#_vx1227)

[Justificación de la Elección 31](#_3fwokq0)

[Diseño de Esquema de Base de Datos 31](#_1v1yuxt)

[**10.**](#_4f1mdlm) **Implementación del Backend 31**

[Elección del Lenguaje de Programación 31](#_2u6wntf)

[Creación de la Lógica de Negocio 31](#_19c6y18)

[Desarrollo de Endpoints y APIs 31](#_3tbugp1)

[Autenticación y Autorización 31](#_28h4qwu)

[**11.**](#_nmf14n) **Conexión a la Base de Datos 31**

[Configuración de la Conexión 32](#_37m2jsg)

[Desarrollo de Operaciones CRUD 32](#_1mrcu09)

[Manejo de Transacciones 32](#_46r0co2)

[**12.**](#_2lwamvv) **Pruebas del Backend 32**

[Diseño de Casos de Prueba 32](#_111kx3o)

[Ejecución de Pruebas Unitarias y de Integración 32](#_3l18frh)

[Manejo de Errores y Excepciones 32](#_206ipza)

[**Etapa 3: Consumo de Datos y Desarrollo Frontend 33**](#_4k668n3)

[**13.**](#_2zbgiuw) **Introducción 33**

[Propósito de la Etapa 33](#_1egqt2p)

[Alcance de la Etapa 33](#_3ygebqi)

[Definiciones y Acrónimos 33](#_2dlolyb)

[**14.**](#_sqyw64) **Creación de la Interfaz de Usuario (UI) 33**

[Diseño de la Interfaz de Usuario (UI) con HTML y CSS 33](#_3cqmetx)

[Consideraciones de Usabilidad 33](#_1rvwp1q)

[Maquetación Responsiva 33](#_4bvk7pj)

[**15.**](#_2r0uhxc) **Programación Frontend con JavaScript (JS) 33**

[Desarrollo de la Lógica del Frontend 34](#_1664s55)

[Manejo de Eventos y Comportamientos Dinámicos 34](#_3q5sasy)

[Uso de Bibliotecas y Frameworks (si aplicable) 34](#_25b2l0r)

[**16.**](#_kgcv8k) **Consumo de Datos desde el Backend 34**

[Configuración de Conexiones al Backend 34](#_34g0dwd)

[Obtención y Presentación de Datos 34](#_1jlao46)

[Actualización en Tiempo Real (si aplicable) 34](#_43ky6rz)

[**17.**](#_2iq8gzs) **Interacción Usuario-Interfaz 34**

[Manejo de Formularios y Validación de Datos 35](#_xvir7l)

[Implementación de Funcionalidades Interactivas 35](#_3hv69ve)

[Mejoras en la Experiencia del Usuario 35](#_1x0gk37)

[**18.**](#_4h042r0) **Pruebas y Depuración del Frontend 35**

[Diseño de Casos de Prueba de Frontend 35](#_2w5ecyt)

[Pruebas de Usabilidad 35](#_1baon6m)

[Depuración de Errores y Optimización del Código 35](#_3vac5uf)

[**19.**](#_2afmg28) **Implementación de la Lógica de Negocio en el Frontend 35**

[Migración de la Lógica de Negocio desde el Backend (si necesario) 36](#_pkwqa1)

[Validación de Datos y Reglas de Negocio en el Frontend 36](#_39kk8xu)

[**20.**](#_1opuj5n) **Integración con el Backend 36**

[Verificación de la Comunicación Efectiva con el Backend 36](#_48pi1tg)

[Pruebas de Integración Frontend-Backend 36](#_2nusc19)

[ANEXOS 36](#_1302m92)

# Etapa 1 Diseño de la Aplicación y Análisis de Requisitos

# Introducción

## Propósito del Documento

El propósito de este documento es recopilar de manera exhaustiva la información relacionada con la creación del componente de mensajería que se integrará en la plataforma Creavi. Esta documentación se enfoca en detallar las distintas fases necesarias para la obtención de un producto final, haciendo uso de la metodología de Design Thinking estructurada en 6 módulos, con el objetivo de clarificar y organizar las ideas de manera efectiva.

## Alcance del Proyecto componente de mensajería

El proyecto tiene como objetivo principal desarrollar un componente de mensajería dentro de la plataforma Creavi, que permitirá a los usuarios interactuar de manera efectiva y eficiente con un chat-bot de respuestas rápidas, el soporte técnico (creador de la página) y otros usuarios. La implementación del componente se llevará a cabo en tres fases durante tres semestres académicos. En el primer semestre, se está enfocando en la documentación detallada de los componentes y sus funcionalidades.

Funcionalidades del componente:

1. **Chat Bot de Respuestas Rápidas:**

Los usuarios podrán interactuar con un chat-bot que proporcionará respuestas rápidas a preguntas frecuentes y consultas comunes.

1. **Soporte Técnico:**

Los usuarios podrán comunicarse con el soporte técnico, que es el creador de la página, para resolver problemas técnicos, recibir asistencia y hacer consultas específicas relacionadas con la plataforma Creavi.

1. **Comunicación entre Usuarios:**

Los usuarios podrán enviar mensajes entre ellos, facilitando la interacción y la colaboración en la plataforma.

Visión Futura:

* El componente de mensajería proporcionará una base sólida para futuras mejoras y expansiones en Creavi. Se podrían considerar las siguientes mejoras a futuro:
* Integración de Multimedia: Permitir a los usuarios enviar y recibir imágenes, vídeos y otros archivos multimedia a través del componente de mensajería.
* Funcionalidades Avanzadas del Chat Bot: Mejorar la inteligencia artificial del chat bot para comprender consultas más complejas y proporcionar respuestas más detalladas.
* Interacción con Otras Plataformas: Integrar el componente de mensajería con otras plataformas y servicios externos para una experiencia de usuario más completa.

Este alcance del proyecto proporciona una visión clara de las funcionalidades a implementar y ver el enfoque por fases para su desarrollo exitoso en el transcurso de tres semestres.

1. **Creavi:** Nombre de la plataforma para la cual se está desarrollando el componente de mensajería. Se refiere al entorno digital donde los usuarios interactúan y colaboran.
2. **Chat-Bot:** Programa informático diseñado para simular una conversación con usuarios, especialmente a través de Internet. En este contexto, se refiere al sistema automatizado que proporciona respuestas rápidas a las preguntas de los usuarios.
3. **Soporte Técnico:** Equipo o individuo responsable de brindar asistencia técnica a los usuarios de la plataforma Creavi. Se refiere al creador de la página o al equipo dedicado a resolver problemas y consultas técnicas.
4. **Componente de Mensajería:** Módulo o funcionalidad específica dentro de Creavi que permite a los usuarios enviar, recibir y gestionar mensajes de texto y otros tipos de contenido a través del chat-bot, el soporte técnico y otros usuarios.

# Descripción General

## Objetivos del Sistema

**Chat-Bot**

1. Interactividad y Engage:

Generar respuestas interactivas y atractivas para los usuarios, manteniendo su interés y participación en la conversación.

1. Comprensión y Respuesta Precisa:

Mejorar la capacidad de comprensión del chat-bot para proporcionar respuestas precisas y relevantes a las preguntas y consultas de los usuarios.

1. **Personalización:**

Personalizar las respuestas del chat-bot según el historial y preferencias del usuario para brindar una experiencia más individualizada.

**Chat Usuario**

**Experiencia del Usuario:**

Garantizar una interfaz de usuario intuitiva y fácil de usar que permita a los usuarios comunicarse de manera efectiva y cómoda.

**Seguridad y Privacidad:**

Garantizar la seguridad y privacidad de las conversaciones de los usuarios, implementando medidas adecuadas de encriptación y protección de datos.

**Chat de Soporte:**

**Eficiencia en la Respuesta:**

Proporcionar respuestas rápidas y precisas a las consultas y problemas de los usuarios para mejorar la satisfacción del cliente.

**Resolución de Problemas:**

Garantizar que el chat de soporte tenga la capacidad de resolver problemas y consultas de los usuarios de manera efectiva y un tiempo hábil de 3 días.

**Reducción de Cargas de Trabajo:**

Automatizar tareas rutinarias para reducir la carga de trabajo del personal de soporte y permitirles enfocarse en problemas más complejos y estratégicos.

**Recopilación de Datos y Retroalimentación:**

Recopilar datos y comentarios de los usuarios para identificar áreas de mejora en el servicio de soporte y optimizar la calidad de las respuestas.

## Funcionalidad General

## La funcionalidad general de un chat, en el contexto de las aplicaciones de mensajería y comunicación en línea, incluye una serie de características que permiten a los usuarios interactuar y comunicarse entre sí. Algunas de las funcionalidades típicas de un chat son las siguientes:

## Capacidad de enviar y recibir mensajes en tiempo real.

## Capacidad de reconocer y procesar las intenciones del usuario.

## Capacidad de responder automáticamente a las preguntas frecuentes.

## Capacidad para hacer preguntas al chat-bot

## Chat-bot redirige al usuario para enviar un mensaje a soporte

## Capacidad para hacer pregunta al chat soporte

## Usuarios del Sistema

Los siguientes usuarios pueden interactuar con el chat dependiendo de las funcionalidades.

| **Funcionalidad** | **Sistema** | **Usuario** | **Chat-bot** | **Soporte** |
| --- | --- | --- | --- | --- |
| Enviar mensajes | ✓ | ✓ | ✓ | ✓ |
| Recibir mensajes | ✓ | ✓ | ✓ | ✓ |
| Responder automáticamente a las preguntas frecuentes | ✓ |  | ✓ |  |
| Reconocer intención del usuario | ✓ |  | ✓ |  |

# Requisitos Funcionales

## Casos de Uso

1. Enviar mensajes
2. Recibir mensajes
3. Responder automáticamente las preguntas frecuentes
4. Reconocer intenciones del usuario

Diagrama de caso de uso

***Diagrama

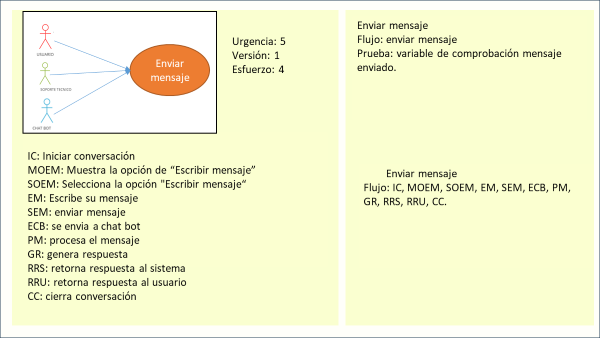
Descripción generada automáticamente***

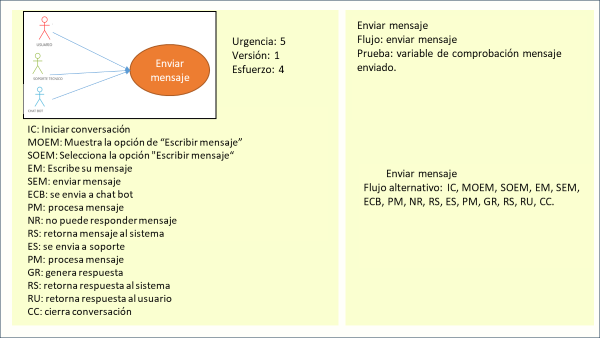
https://app.creately.com/d/start/dashboard

## 

## Diagramas de Flujo de Casos de Uso

**CU-1**





**CU-2**

Escala de tiempo

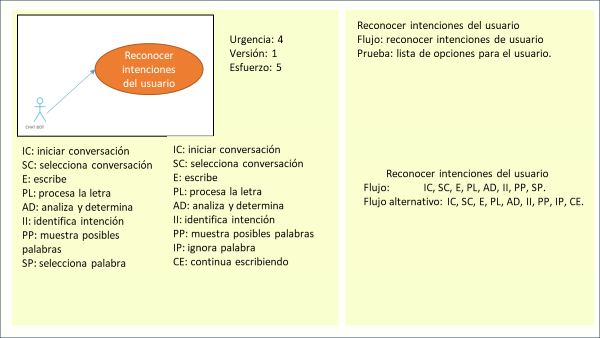
Descripción generada automáticamente

**CU-3**

Texto

Descripción generada automáticamente con confianza baja

**CU-4**



Descripción detallada de cada caso de uso

**CASO No. 1 Enviar mensajes**

| ID: | CU-1 | | | |
| --- | --- | --- | --- | --- |
| Nombre | Enviar mensajes | | | |
| Actores | Usuario, chat-bot, soporte | | | |
| Objetivo | Permitir enviar mensajes | | | |
| Urgencia | 5 | | | |
| Esfuerzo | 4 | | | |
| Pre-condiciones | Estar autenticado | | |  |
| **Flujo Normal** | **USUARIO** | **SISTEMA** | **CHAT-BOT** | **SOPORTE** |
| Iniciar conversación |  |  |  |
|  | Muestra la opción de “Escribir mensaje” |  |  |
| Selecciona la opción "Escribir mensaje" |  |  |  |
| Escribe su mensaje |  |  |  |
| Enviar mensaje |  |  |  |
|  | Se envía a chat-bot |  |  |
|  |  | Procesa el mensaje |  |
|  |  |  | Genera respuesta |  |
|  |  |  | Retorna respuesta al sistema |  |
|  |  | Retorna respuesta al usuario |  |  |
|  | Cierra conversación |  |  |  |
| **Flujo Alternativo 1** | Iniciar conversación |  |  |  |
|  | Muestra la opción de “Escribir mensaje” |  |  |
| Selecciona la opción "Escribir mensaje" |  |  |  |
| Escribe su mensaje |  |  |  |
| Enviar mensaje |  |  |  |
|  | Se envía a chat-bot |  |  |
|  |  | Procesa el mensaje |  |
|  |  |  | No puede responder al mensaje |  |
|  |  |  | Retorna mensaje al sistema |  |
|  |  | Se envía a soporte |  |  |
|  |  |  |  | Procesa el mensaje |
|  |  |  |  | Genera respuesta |
|  |  |  |  | Retorna respuesta al sistema |
|  |  | Retorna respuesta al usuario |  |  |
|  | Cierra conversación |  |  |  |

**CASO No. 2 Recibir mensajes**

| ID: | CU-2 | | | |
| --- | --- | --- | --- | --- |
| Nombre | Recibir mensajes | | | |
| Actores | Usuario, chat-bot, soporte | | | |
| Objetivo | Permitir recibir mensajes | | | |
| Urgencia | 4 | | | |
| Esfuerzo | 4 | | | |
| Pre-condiciones | Estar autenticado | | |  |
| **Flujo Normal** | **USUARIO** | **SISTEMA** | **CHAT-BOT** | **SOPORTE** |
| Emite mensajes |  |  |  |
|  | Procesa los mensajes |  |  |
|  | Envía a usuario |  |  |
| Recibe mensaje |  |  |  |
| **Flujo Alternativo 1** | Emite mensajes |  |  |  |
|  | Procesa los mensajes |  |  |
|  | Envía a Chat-bot |  |  |
|  |  | Recibe los mensajes |  |
| **Flujo Alternativo 2** | Emite mensajes |  |  |  |
|  | Procesa los mensajes |  |  |
|  | Envía a Soporte |  |  |
|  |  |  | Recibe los mensajes |

**CASO No. 3 Responder automáticamente las preguntas frecuentes**

| ID: | CU-4 | | | |
| --- | --- | --- | --- | --- |
| Nombre | Responder automáticamente las preguntas frecuentes | | | |
| Actores | Chat-bot | | | |
| Objetivo | Permitir responder preguntas frecuentes | | | |
| Urgencia | 5 | | | |
| Esfuerzo | 4 | | | |
| Pre-condiciones | Estar autenticado | | |  |
| **Flujo Normal** | **USUARIO** | **SISTEMA** | **CHAT-BOT** | **SOPORTE** |
| Iniciar una conversación con el chat-bot |  |  |  |
|  | Ejecuta al Chat-bot |  |  |
|  |  | Muestra opciones para enviar mensajes y preguntas frecuentes |  |
| Selecciona la opción "Preguntas frecuentes" |  |  |  |
|  |  | Muestra una lista de preguntas frecuentes |  |
| Selecciona una pregunta de la lista |  |  |  |
|  | Procesa la selección |  |  |
|  |  | Recibe la pregunta |  |
|  |  | Busca la respuesta correspondiente en la base de datos de preguntas frecuentes |  |
|  |  | Encuentra respuesta |  |
|  |  | Envía la respuesta correspondiente automáticamente |  |
|  | Despliega respuesta |  |  |
| Visualiza la respuesta |  |  |  |

**CASO No. 4 Reconocer intenciones del usuario**

| ID: | CU-7 | | | |
| --- | --- | --- | --- | --- |
| Nombre | Reconocer intenciones del usuario | | | |
| Actores | Chat-bot | | | |
| Objetivo | Permitir reconocer las intenciones del usuario | | | |
| Urgencia | 5 | | | |
| Esfuerzo | 5 | | | |
| Pre-condiciones | Estar autenticado | | |  |
| **Flujo Normal** | **USUARIO** | **SISTEMA** | **CHAT-BOT** | **SOPORTE** |
| Iniciar conversación |  |  |  |
| Selecciona conversación |  |  |  |
| Escribe |  |  |  |
|  | Procesa las letras |  |  |
|  | Envía letras a chat-bot |  |  |
|  |  | Analiza las letras para determinar su intención |  |
|  |  | Identifica intención |  |
|  | Muestra posibles palabras |  |  |
| Selecciona palabra |  |  |  |
| **Flujo alternativo** | Iniciar conversación |  |  |  |
| Selecciona conversación |  |  |  |
| Escribe |  |  |  |
|  | Procesa las letras |  |  |
|  | Envía letras a chat-bot |  |  |
|  |  | Analiza las letras para determinar su intención |  |
|  |  | Identifica intención |  |
|  | Muestra posibles palabras |  |  |
| Ignora palabras |  |  |  |
| Continúa escribiendo |  |  |  |

## Prioridad de Requerimientos

**A continuación, se presentan algunos requisitos**

**Obligatorios:**

* Capacidad de enviar y recibir mensajes de texto.
* Capacidad de enviar y recibir mensajes multimedia (imágenes, videos, etc.).
* Capacidad de enviar y recibir mensajes en tiempo real.
* Capacidad de enviar y recibir mensajes en grupos.
* Capacidad de reconocer y procesar las intenciones del usuario.
* Capacidad de responder automáticamente a las preguntas frecuentes.

**Importantes:**

* Capacidad de enviar y recibir mensajes encriptados.
* Capacidad de enviar y recibir mensajes programados.
* Capacidad de enviar y recibir mensajes con confirmación de lectura.
* Capacidad de personalizar la experiencia del usuario.
* Capacidad de integrarse con sistemas de gestión de tickets y soporte.

**Deseables:**

* Capacidad de enviar y recibir mensajes con traducción automática.
* Capacidad de enviar y recibir mensajes con filtros de spam.
* Capacidad de enviar y recibir mensajes con análisis de sentimiento.
* Capacidad de enviar y recibir mensajes con recomendaciones personalizadas.
* Capacidad de enviar y recibir mensajes con integración de chat-bot de terceros.

A partir del análisis de requerimientos, funcionalidades y el proceso de design thinking, se concreta la siguiente matriz de prioridad de requerimientos.

Para la interpretación se tiene en cuenta la siguiente escala con sus valores.

Eje de Urgencia:

* Obligatoria (5)
* Alta (4)
* Moderada (3)
* Menor (2)
* Baja (1)

Eje de Esfuerzo:

* Muy alto (5)
* Alto (4)
* Medio (3)
* Bajo (2)
* Muy bajo (1)

|  | Urgencia | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Impacto |  | 1-Baja | 2-Menor | 3-Moderada | 4-Alta | 5-Obligatoria |
| 5-Muy alto | 5 | 10 | 15 | 20 | 25 |
|  |  |  |  | **CU-1**  **CU-2** |
| 4-Alto | 4 | 8 | 12 | 16 | 20 |
|  |  |  | **CU-4** | CU-3 |
| 3-Medio | 3 | 6 | 9 | 12 | 15 |
|  |  |  |  |  |
| 2-Bajo | 2 | 4 | 6 | 8 | 10 |
|  |  |  |  |  |
| 1-Muy bajo | 1 | 2 | 3 | 4 | 5 |
|  |  |  |  |  |

https://asana.com/es/resources/priority-matrix

# Requisitos No Funcionales

## Requisitos de Desempeño

**Rendimiento de chat-bot:** Este debe proporcionar un rendimiento de preguntas rápidas en tiempo real la cual deben de reflejarse con el usuario para resolver múltiples problemas asignadas y calificadas como las que siempre preguntan simultáneamente por pasos para devolverse por caso de que la persona cometa un error.

**Rendimiento de chat (usuario con contactos):** en esta parte se cargarán rápido los mensajes de los usuarios para que no tengan inconvenientes al momento de chatear, en caso de fallar creavi cuenta la opción de chat-bot y chat al soporte en casos mayores.

**Rendimiento de chat de soporte:** Este debe proporcionar en tiempo real la ayuda brindada en caso de que no encuentren posibles soluciones con el chat-bot, en esta opción se comunicara un cliente que es llamado usuario con una persona administradora de la página creavi con conocimiento avanzado en el tema para brindarle la ayuda requerida siempre y cuando este en los parámetros de la página.

**Requisitos de seguridad:**

* **Acceso seguro:** Se debe implementar una autenticación segura para garantizar que solo usuarios autorizados tengan acceso a el chat. Esto puede incluir autenticación de 3 factores, inicio de sesión único, autenticación de seguridad con extremo a bot, y Autenticación de seguridad con extremo y soporte.
* **Alojamiento:** La interfaz del chat-bot es un aspecto a tener en cuenta. Si el bot se utiliza a través de una plataforma de chat como Creavi, la información del cliente quedará expuesta y registrada por esta plataforma. Esto puede o no ser un problema de seguridad del chat-bot, pero es evidente que hay que tenerlo en cuenta.
* **Registro de actividades:** El sistema debe mantener registros de actividades, lo que incluye registros de cambios en la plataforma más que todo el chat, acceso de usuarios y eventos relevantes para la seguridad.
* **Políticas y procedimientos:** Por supuesto, es fundamental para todo lo anterior que se establezcan políticas y procedimientos pertinentes que rijan las normas de seguridad de la información. La seguridad de la información no se establece una sola vez, sino que es una actividad continua. Estas políticas y procedimientos regirán no sólo la configuración del software pertinente, sino que también especificarán cuándo y cómo se llevarán a cabo las sesiones periódicas de formación y las pruebas de seguridad. Proteger un chat-bot no difiere de proteger cualquier otro programa informático. Al principio del proceso debe evaluarse el grado de confidencialidad de los datos subyacentes, lo que determinará las medidas que debe adoptar la organización para garantizar la seguridad de los datos.

## Requisitos de Seguridad

**Acceso seguro:** Se debe implementar una autenticación segura para garantizar que solo usuarios autorizados tengan acceso a el chat. Esto puede incluir autenticación de 3 factores, inicio de sesión único, autenticación de seguridad con extremo a bot, y Autenticación de seguridad con Extremo y soporte

**Alojamiento:** La interfaz del chat-bot es un aspecto a tener en cuenta. Si el bot se utiliza a través de una plataforma de chat como Creavi, la información del cliente quedará expuesta y registrada por esta plataforma. Esto puede o no ser un problema de seguridad del chat-bot, pero es evidente que hay que tenerlo en cuenta.

**Registro de actividades:** El sistema debe mantener registros de actividades, lo que incluye registros de cambios en la plataforma más que todo el chat, acceso de usuarios y eventos relevantes para la seguridad.

**Políticas y procedimientos:** Por supuesto, es fundamental para todo lo anterior que se establezcan políticas y procedimientos pertinentes que rijan las normas de seguridad de la información. La seguridad de la información no se establece una sola vez, sino que es una actividad continua. Estas políticas y procedimientos regirán no sólo la configuración del software pertinente, sino que también especificarán cuándo y cómo se llevarán a cabo las sesiones periódicas de formación y las pruebas de seguridad. Proteger un chatbot no difiere de proteger cualquier otro programa informático. Al principio del proceso debe evaluarse el grado de confidencialidad de los datos subyacentes, lo que determinará las medidas que debe adoptar la organización para garantizar la seguridad de los datos.

## Requisitos de Usabilidad

**Facilidad de Ingreso y Registro:**

* Los usuarios deben poder ingresar al chat de manera sencilla y rápida, sin obstáculos innecesarios.
* Proporciona opciones de registro simples, como un correo electrónico.

**Navegación Intuitiva:**

* El chat debe tener una interfaz de usuario fácil de usar y una navegación clara.
* Los usuarios deben poder alternar de manera intuitiva entre conversaciones con otros, Chat-bot y chat de soporte.

**Interfaz Conversacional:**

* El diseño del chat debe imitar una conversación real para que los usuarios se sientan cómodos.
* Utiliza avatares de quién está hablando para que las conversaciones sean fáciles de seguir.

**Retroalimentación Clara:** El sistema debe proporcionar retroalimentación clara, como indicadores de escritura, confirmaciones de envío y notificaciones de mensajes no leídos.

**integración de Chat-bot:** El chat- bot debe ser capaz de comprender las consultas de los usuarios y proporcionar respuestas útiles y precisas.

**Privacidad y Seguridad:** Garantiza la privacidad de las conversaciones y proporciona opciones de seguridad.

**Atención al Cliente Efectiva**: El chat de soporte debe ser accesible y brindar respuestas rápidas y útiles a las preguntas y problemas de los usuarios.

**Tiempo de Respuesta Rápida:** Tanto los usuarios como los chat-bot y el chat de soporte deben responder de manera eficiente para mantener las conversaciones fluidas.

## Requisitos de Escalabilidad

**Gestión de Conversaciones Múltiples:** El sistema debe ser capaz de gestionar múltiples conversaciones simultáneas entre usuarios y Chat-bot, así como entre usuarios y agentes de soporte. Esto implica mantener un estado eficiente para cada conversación y asegurarse de que las interacciones no se mezclen.

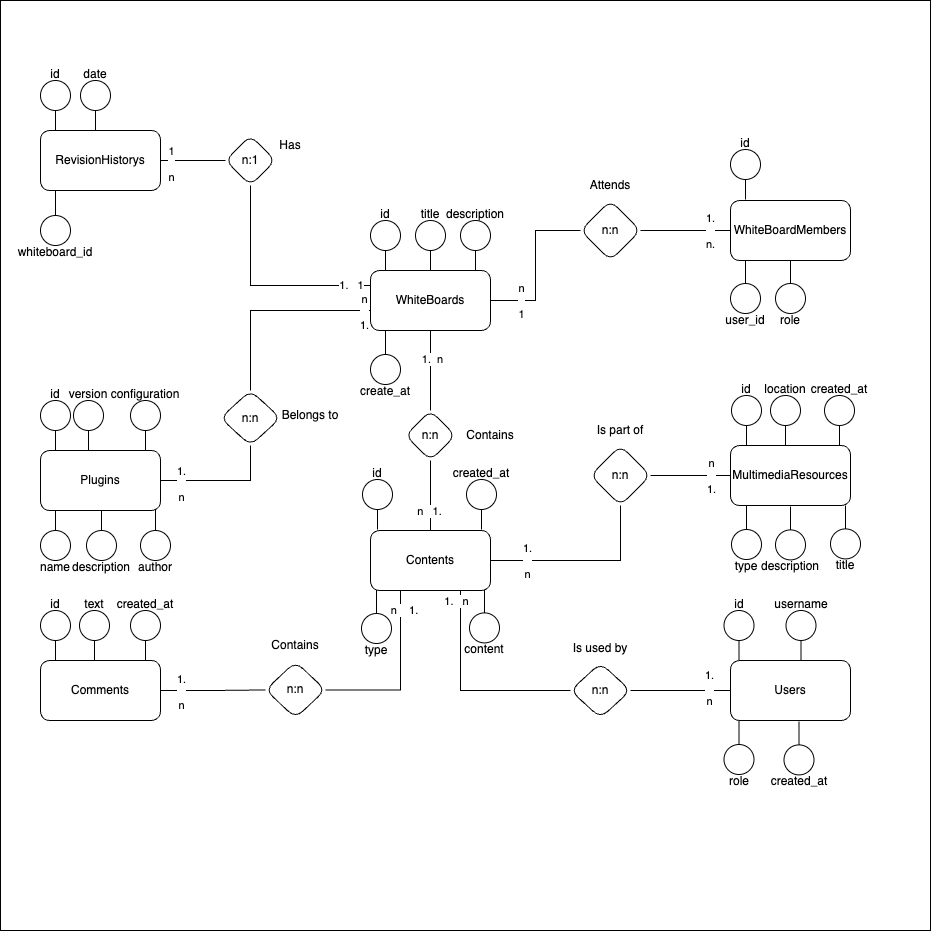
**Distribución de Cargas Equitativas:** Implementar un mecanismo de balanceo de carga que distribuya de manera equitativa las solicitudes de los usuarios entre los servidores o instancias del Chat-bot y los agentes de soporte.

**Notificaciones:** El sistema debe ser capaz de manejar notificaciones para los agentes de soporte, lo que significa que los agentes deben recibir alertas cuando un usuario solicita asistencia.

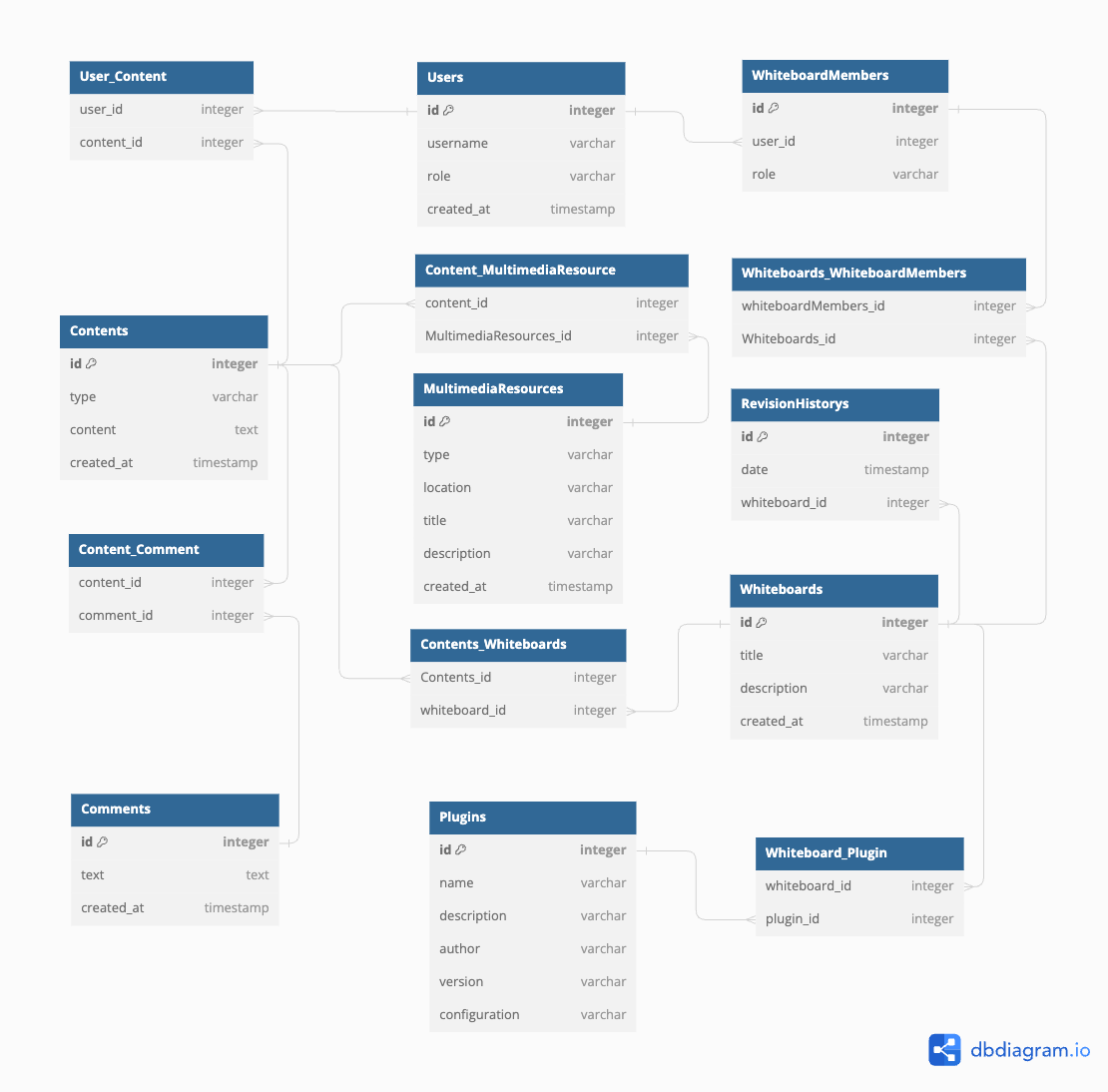
**Integración de IA y Aprendizaje Automático:** Si se utiliza IA o aprendizaje automático en el Chat-bot, asegúrate de que el sistema pueda escalar para manejar el entrenamiento y la inferencia de modelos a medida que la cantidad de datos y usuarios aumente.

# Modelado E/R

## Diagrama de Entidad-Relación



## Diagrama Relacional



## 

## Script de modelo relacional

<https://dbdiagram.io/>

| Table Users {  id integer [primary key]  username varchar  role varchar  created\_at timestamp  }  Table User\_Content {  user\_id integer [ref: > Users.id]  content\_id integer [ref: > Contents.id]  }  Table Content\_Comment {  content\_id integer [ref: > Contents.id]  comment\_id integer [ref: > Comments.id]  }  Table Whiteboards {  id integer [primary key]  title varchar  description varchar  created\_at timestamp  }  Table Whiteboards\_WhiteboardMembers {  whiteboardMembers\_id integer [ref: > WhiteboardMembers.id]  Whiteboards\_id integer [ref: > Whiteboards.id]  }  Table Whiteboard\_Plugin {  whiteboard\_id integer [ref: > Whiteboards.id]  plugin\_id integer [ref: > Plugins.id]  }  Table Contents {  id integer [primary key]  type varchar  content text [note: 'Content of the content']  created\_at timestamp  }  Table Contents\_Whiteboards {  Contents\_id integer [ref: > Contents.id]  whiteboard\_id integer [ref: > Whiteboards.id]  } | Table Content\_MultimediaResource {  content\_id integer [ref: > Contents.id]  MultimediaResources\_id integer [ref: > MultimediaResources.id]  }  Table Comments {  id integer [primary key]  text text  created\_at timestamp  }  Table RevisionHistorys {  id integer [primary key]  date timestamp  whiteboard\_id integer [ref: > Whiteboards.id]  }  Table MultimediaResources {  id integer [primary key]  type varchar  location varchar  title varchar  description varchar  created\_at timestamp  }  Table Plugins {  id integer [primary key]  name varchar  description varchar  author varchar  version varchar  configuration varchar  }  Table WhiteboardMembers {  id integer [primary key]  user\_id integer [ref: > Users.id]  role varchar  } |
| --- | --- |

## Descripción de Entidades y Relaciones

Entidades:

1. **User (Usuario):**

Almacena información sobre los usuarios que pueden acceder a la pizarra.

**Atributos:** ID (identificador único), nombre de usuario, rol (como administrador, docente, estudiante), fecha de creación.

**Relaciones:** Cada usuario puede estar asociado con varias pizarras a través de la entidad "WhiteboardMember."

1. **Whiteboard (Pizarra):**

Representa una pizarra en la aplicación de pizarra compartida.

**Atributos:** ID (identificador único), título de la pizarra, descripción, fecha de creación.

**Relaciones:** Cada pizarra puede contener contenido a través de la relación con la entidad "Content" y puede tener plugins asociados a través de la relación con la entidad "Plugin."

1. **Content (Contenido):**

Almacena contenido que se puede agregar a las pizarras, como texto, imágenes, videos, documentos, etc.

**Atributos:** ID (identificador único), tipo de contenido, contenido en sí, fecha de creación.

**Relaciones:** El contenido se asocia con un usuario a través de la relación con la entidad "User" y puede estar relacionado con comentarios.

1. **Comment (Comentario):**

Almacena comentarios realizados por los usuarios en relación con el contenido de la pizarra.

**Atributos:** ID (identificador único), texto del comentario, fecha de creación.

**Relaciones:** Cada comentario se relaciona con el contenido específico en la entidad "Content."

1. **RevisionHistory (Historial de Revisiones):**

Registra el historial de revisiones y cambios realizados en las pizarras.

**Atributos:** ID (identificador único), fecha de la revisión.

**Relaciones:** Cada entrada de historial se relaciona con una pizarra específica en la entidad "Whiteboard."

1. **MultimediaResource (Recurso Multimedia):**

Almacena recursos multimedia, como imágenes, videos, documentos, etc.

**Atributos:** ID (identificador único), tipo de recurso, ubicación o URL del recurso, título, descripción, fecha de carga.

**Relaciones:** Cada entrada de Recurso multimedia se relaciona con una un contenido específico en la entidad " Content."

1. **Plugin:**

Almacena información sobre los plugins que pueden proporcionar funcionalidades personalizadas en las pizarras.

**Atributos:** ID (identificador único), nombre del plugin, descripción, autor, versión, configuración.

**Relaciones:** Cada pizarra puede tener asociado uno o varios plugins a través de la entidad "Whiteboard."

Relaciones:

* "User" se relaciona con "WhiteboardMember" para indicar la asociación de los usuarios con las pizarras.
* "Content" se relaciona con "User" para registrar los usuarios que pueden interactuar con el contenido.
* "Content" se relaciona con "Comment" para permitir comentarios en los contenidos.
* "Whiteboard" se relaciona con "Content" para indicar que una pizarra puede contener contenido.
* "Whiteboard" se relaciona con "Plugin" para permitir la asociación de plugins con las pizarras.
* "WhiteboardMember" se relaciona con "User" y "Whiteboard" para indicar la asociación de usuarios con pizarras y sus roles.
* "RevisionHistory" se relaciona con "Whiteboard" para registrar revisiones en las pizarras.
* "MultimediaResource " se relaciona con "Content" para registrar recursos asociados en el contenido.

## Reglas de Integridad Referencial

1. **Integridad Referencial entre "User\_Content" y "Users"**: Cada registro en la tabla "User\_Content" debe estar asociado con un usuario existente en la tabla "Users" a través de la clave foránea "user\_id."
2. **Integridad Referencial entre "User\_Content" y "Contents"**: Cada registro en la tabla "User\_Content" debe estar asociado con un contenido existente en la tabla "Contents" a través de la clave foránea "content\_id."
3. **Integridad Referencial entre "Content\_Comment" y "Contents"**: Cada registro en la tabla "Content\_Comment" debe estar asociado con un contenido existente en la tabla "Contents" a través de la clave foránea "content\_id."
4. **Integridad Referencial entre "Content\_Comment" y "Comments"**: Cada registro en la tabla "Content\_Comment" debe estar asociado con un comentario existente en la tabla "Comments" a través de la clave foránea "comment\_id."
5. **Integridad Referencial entre "Whiteboards\_WhiteboardMembers" y "Whiteboards"**: Cada registro en la tabla "Whiteboards\_WhiteboardMembers" debe estar asociado con una pizarra existente en la tabla "Whiteboards" a través de la clave foránea "Whiteboards\_id."
6. **Integridad Referencial entre "Whiteboards\_WhiteboardMembers" y "WhiteboardMembers"**: Cada registro en la tabla "Whiteboards\_WhiteboardMembers" debe estar asociado con un miembro de la pizarra existente en la tabla "WhiteboardMembers" a través de la clave foránea "whiteboardMembers\_id."
7. **Integridad Referencial entre "Whiteboard\_Plugin" y "Whiteboards"**: Cada registro en la tabla "Whiteboard\_Plugin" debe estar asociado con una pizarra existente en la tabla "Whiteboards" a través de la clave foránea "whiteboard\_id."
8. **Integridad Referencial entre "Whiteboard\_Plugin" y "Plugins"**: Cada registro en la tabla "Whiteboard\_Plugin" debe estar asociado con un plugin existente en la tabla "Plugins" a través de la clave foránea "plugin\_id."
9. **Integridad Referencial entre "Contents\_Whiteboards" y "Contents"**: Cada registro en la tabla "Contents\_Whiteboards" debe estar asociado con un contenido existente en la tabla "Contents" a través de la clave foránea "Contents\_id."
10. **Integridad Referencial entre "Contents\_Whiteboards" y "Whiteboards"**: Cada registro en la tabla "Contents\_Whiteboards" debe estar asociado con una pizarra existente en la tabla "Whiteboards" a través de la clave foránea "whiteboard\_id."
11. **Integridad Referencial entre "Content\_MultimediaResource" y "Contents"**: Cada registro en la tabla "Content\_MultimediaResource" debe estar asociado con un contenido existente en la tabla "Contents" a través de la clave foránea "content\_id."
12. **Integridad Referencial entre "Content\_MultimediaResource" y "MultimediaResources"**: Cada registro en la tabla "Content\_MultimediaResource" debe estar asociado con un recurso multimedia existente en la tabla "MultimediaResources" a través de la clave foránea "MultimediaResources\_id."
13. **Integridad Referencial entre "RevisionHistorys" y "Whiteboards"**: Cada registro en la tabla "RevisionHistorys" debe estar asociado con una pizarra existente en la tabla "Whiteboards" a través de la clave foránea "whiteboard\_id."
14. **Integridad Referencial entre "WhiteboardMembers" y "Users"**: Cada registro en la tabla "WhiteboardMembers" debe estar asociado con un usuario existente en la tabla "Users" a través de la clave foránea "user\_id."

## Colecciones (NoSLQ)

| **Users**: {  \_id: ObjectId,  username: String,  role: String,  created\_at: date,  contents\_id: [ObjectId],  }  **Whiteboards**: {  \_id: ObjectId,  title: String,  description: String,  created\_at: Date,  contents\_id: [ObjectId],  whiteboardMembers\_id: [ObjectId],  plugin\_id: [ObjectId],  }  **Contents**: {  \_id: ObjectId,  type: String,  content: String,  created\_at: Date,  user\_id: [ObjectId],  comment\_id: [ObjectId],  whiteboard\_id: [ObjectId],  multimediaResources\_id: [ObjectId],  }  **WhiteboardMembers**: {  \_id: ObjectId,  user\_id: ObjectId,  role: String,  whiteboards\_id: [ObjectId],  } | **Comments**: {  \_id: ObjectId,  text: String,  created\_at: ObjectId,  contents\_id: [ObjectId],  }  **RevisionHistorys**: {  \_id: ObjectId,  Date: Date,  whiteboard\_id: [ObjectId],  }  **MultimediaResources**: {  \_id: ObjectId,  Type: String,  Location: String,  Title: String,  Description: String,  created\_at: Date,  content\_id: [ObjectId],  }  **Plugins**: {  \_id: ObjectId,  Name: String,  Description: String,  Author: ObjectId,  Version: String,  Configuration: Object,  whiteboard\_id: [ObjectId],  } |
| --- | --- |
|  |  |

# Anexos

## Diagramas Adicionales

## Referencias

# Etapa 2: Persistencia de Datos con Backend

# Introducción

## Propósito de la Etapa

## Alcance de la Etapa

## Definiciones y Acrónimos

# Diseño de la Arquitectura de Backend

## Descripción de la Arquitectura Propuesta

## Componentes del Backend

## Diagramas de Arquitectura

# Elección de la Base de Datos

## Evaluación de Opciones (SQL o NoSQL)

## Justificación de la Elección

## Diseño de Esquema de Base de Datos

# Implementación del Backend

## Elección del Lenguaje de Programación

## Creación de la Lógica de Negocio

## Desarrollo de Endpoints y APIs

## Autenticación y Autorización

# Conexión a la Base de Datos

## Configuración de la Conexión

## Desarrollo de Operaciones CRUD

## Manejo de Transacciones

# Pruebas del Backend

## Diseño de Casos de Prueba

## Ejecución de Pruebas Unitarias y de Integración

## Manejo de Errores y Excepciones

# Etapa 3: Consumo de Datos y Desarrollo Frontend

# Introducción

## Propósito de la Etapa

## Alcance de la Etapa

## Definiciones y Acrónimos

# Creación de la Interfaz de Usuario (UI)

## Diseño de la Interfaz de Usuario (UI) con HTML y CSS

## Consideraciones de Usabilidad

## Maquetación Responsiva

# Programación Frontend con JavaScript (JS)

## Desarrollo de la Lógica del Frontend

## Manejo de Eventos y Comportamientos Dinámicos

## Uso de Bibliotecas y Frameworks (si aplicable)

# Consumo de Datos desde el Backend

## Configuración de Conexiones al Backend

## Obtención y Presentación de Datos

## Actualización en Tiempo Real (si aplicable)

# Interacción Usuario-Interfaz

## Manejo de Formularios y Validación de Datos

## Implementación de Funcionalidades Interactivas

## Mejoras en la Experiencia del Usuario

# Pruebas y Depuración del Frontend

## Diseño de Casos de Prueba de Frontend

## Pruebas de Usabilidad

## Depuración de Errores y Optimización del Código

# Implementación de la Lógica de Negocio en el Frontend

## Migración de la Lógica de Negocio desde el Backend (si necesario)

## Validación de Datos y Reglas de Negocio en el Frontend

# Integración con el Backend

## Verificación de la Comunicación Efectiva con el Backend

## Pruebas de Integración Frontend-Backend

## ANEXOS

Diagramas UML

* **Diagrama de Casos de Uso (Use Case Diagram):** Este diagrama muestra las interacciones entre los actores (usuarios) y el sistema. Puede ayudar a identificar las funcionalidades clave y los actores involucrados.
* **Diagrama de Secuencia (Sequence Diagram):** Estos diagramas muestran la interacción entre objetos y actores a lo largo del tiempo. Puedes utilizarlos para representar cómo los usuarios interactúan con la pizarra en un flujo de trabajo específico.
* **Diagrama de Clases (Class Diagram):** Puedes utilizar este diagrama para modelar las clases y estructuras de datos subyacentes en el sistema, como usuarios, pizarras, comentarios, revisiones, etc.
* **Diagrama de Estados (State Diagram):** Este diagrama puede ser útil para modelar el comportamiento de la pizarra en diferentes estados, como "edición", "visualización", "comentario", etc.
* **Diagrama de Despliegue (Deployment Diagram):** Puedes utilizar este diagrama para representar cómo se despliega la aplicación en servidores y cómo interactúa con otros componentes del sistema, como el CMS.
* **Diagrama de Componentes (Component Diagram):** Este diagrama puede ayudar a representar la estructura de componentes del software, como la interfaz de usuario, la lógica de negocio, las bibliotecas y los servicios utilizados.
* **Diagrama de Actividad (Activity Diagram):** Puedes usar este diagrama para modelar flujos de trabajo o procesos específicos, como el flujo de trabajo de creación y edición de contenido en la pizarra.
* **Diagrama de Comunicación (Communication Diagram):** Similar a los diagramas de secuencia, estos diagramas muestran interacciones entre objetos y actores, pero pueden ser más simples y enfocados en la comunicación.
* **Diagrama de Paquetes (Package Diagram):** Este diagrama puede ayudar a organizar y visualizar los paquetes y módulos del software, lo que es útil para el diseño modular.
* **Diagrama de Objetos (Object Diagram):** Puedes utilizar este diagrama para representar instancias de clases y cómo interactúan en un escenario específico.