#Discord Bot manager

TRABAJO FINAL DE CICLO EN DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Pelayo Palacio Suárez

#curso académico 2024/2025#

#Oviedo, FECHA#

Contenido

[**1. Descripción general del proyecto** 4](#_Toc177584882)

[ Introducción: Resumen de las características del proyecto 4](#_Toc177584883)

[ Presentación de las características 4](#_Toc177584884)

[ Alcance del proyecto: Detalle del problema planteado indicando, punto por punto, todos los detalles que se van a gestionar. 5](#_Toc177584885)

[ Justificación y análisis de la realidad 7](#_Toc177584886)

[ Marco legal 9](#_Toc177584887)

[ Marco teórico 9](#_Toc177584888)

[ Temporalidad 14](#_Toc177584889)

[**2. estudio de la viabilidad del sistema** 15](#_Toc177584890)

[ Análisis económico (DAFO) 15](#_Toc177584891)

[ Plan de marketing 15](#_Toc177584892)

[**3. Descripción del entorno tecnológico** 16](#_Toc177584893)

[ Descripción de todos los perfiles de usuario que habrá en la aplicación 16](#_Toc177584894)

[ Descripción de las tecnologías para cada perfil 17](#_Toc177584895)

[**4. Especificación de requisitos** 18](#_Toc177584896)

[ Requisitos funcionales 18](#_Toc177584897)

[ Requisitos no funcionales 19](#_Toc177584898)

[**5. Modelo entidad / relación** 21](#_Toc177584899)

[ Explicar por qué ha salido ese E/R, de dónde salen las Entidades y de dónde salen las Relaciones. 21](#_Toc177584900)

[ Plasmar el modelo E/R 21](#_Toc177584901)

[ Explicar el origen de los atributos 21](#_Toc177584902)

[ Explicar por qué se han elegido esas claves primarias. 21](#_Toc177584903)

[ Modelo Relacional 21](#_Toc177584904)

[**6. Diagrama de procesos** 21](#_Toc177584905)

[ Casos de Uso 21](#_Toc177584906)

[ Diagrama de clases 21](#_Toc177584907)

[**7. Diseño Interfaz** 21](#_Toc177584908)

[ Diagramación y prototipado del diseño gráfico de la interfaz 21](#_Toc177584909)

[ Guías de estilo 23](#_Toc177584910)

[ Mapa de navegación 24](#_Toc177584911)

[**8 Pruebas** 24](#_Toc177584912)

[ Pruebas unitarias 24](#_Toc177584913)

[ Pruebas de integridad 24](#_Toc177584914)

[**9. Manual de usuario** 24](#_Toc177584915)

[ Manual del cliente 24](#_Toc177584916)

[ Manual de instalación y despliegue 24](#_Toc177584917)

[**10. CONCLUSIONES** 24](#_Toc177584918)

[**11. Anexo** 24](#_Toc177584919)

[ Bibliografía 24](#_Toc177584920)

[ Referencias a texto y/o web de ayuda 24](#_Toc177584921)

[ Referencia de imágenes 24](#_Toc177584922)

# **1. Descripción general del proyecto**

* Introducción: Resumen de las características del proyecto

El proyecto pretende cumplir dos funciones principales. La primera siendo ser una forma de aplicar los contenidos estudiados en la FP de Desarrollo de Aplicaciones Multiplataforma en un caso real de uso sin cerrarse sólo a los propios contenidos proporcionados por los profesores. Y la segunda permitir el control de un Bot de Discord de forma local tanto en aplicación de escritorio como en una versión móvil permitiendo a los usuarios no depender de un host remoto y tener control total sobre los datos.

Dentro de las funciones disponibles se encontrará los comandos básicos de moderación, log de dichas acciones, manipulación de servidores, etc.…

* Presentación de las características

Aplicación de escritorio:

La aplicación de escritorio permitirá al usuario ejecutar el Bot, tener acceso al sistema de módulos de este permitiendo desactivar ciertas funciones en ejecución o previo a su lanzamiento mediante el uso de un archivo de texto para su configuración. Se pretende que la versión de escritorio sea la que permita la modificación de funcionalidades al mismo tiempo que host para la versión de móvil en el caso de que se busque editar un Bot en lugar de hostearlo desde el dispositivo móvil. Deberá ser capaz de mostrar información del servidor y sus usuarios mediante la id\_server o id\_user, así como las acciones de los moderadores quedando todo ello registrado en una base de datos.

Aplicación móvil:

La aplicación móvil tendrá dos funciones principales, hostear un Bot no modificable para dar un uso a dispositivos viejos o en desuso y ser capaz de acceder a las modificaciones del Bot hosteado por la versión de escritorio y a los datos que la misma proporcione.

Bot de Discord:

El Bot deberá ser capaz de realizar funciones básicas de moderación, administración de servidores, modificación de estos, control de usuarios, obtención de información de ellos, …

* Alcance del proyecto: Detalle del problema planteado indicando, punto por punto, todos los detalles que se van a gestionar.

Alcance del proyecto:

El proyecto tiene como objetivo principal proporcionar una solución accesible y personalizable para la gestión de bots de Discord a través de una aplicación multiplataforma. Esta solución aborda varios problemas que enfrentan los usuarios al administrar y personalizar bots, como la dependencia de servicios de hosting remotos, la complejidad de configurar funcionalidades avanzadas, y la falta de control directo sobre los datos y el rendimiento del Bot.

Problemas planteados:

* + Dependencia de un hosting remoto: Los bots de Discord suelen necesitar servidores externos para ejecutarse, lo que implica costos adicionales, falta de control sobre el entorno y riesgos de seguridad.
  + Falta de accesibilidad a funciones avanzadas: Muchas requieren de servicios premium o gran conocimiento técnico para implementarlos.
  + Carencia de aplicaciones locales para el manejo de los bots: La mayoría de bots de Discord tienen una dashboard online a través de la cual se controlan.
  + Seguridad y privacidad: El hosting local permite evitar problemas de seguridad al no tener control sobre los protocolos con los que se tratan los datos.
  + Facilidad de uso: La mayoría de bots son de fácil uso, aunque no disponen de una estructura estándar de desarrollo.

Detalles que gestionar:

* + Creación de una aplicación de escritorio
    - Desarrollo de una interfaz que permita al usuario:
      * Configurar módulos y comandos del Bot.
      * Ejecutar el Bot localmente y actuar como host para otras aplicaciones.
      * Visualizar información relevante de servidores y usuarios (ID, acciones, registros, etc.).
    - Implementación de una base de datos para registrar actividades y datos del Bot.
  + Desarrollo de una aplicación móvil
    - Función principal como host para bots en dispositivos móviles.
    - Acceso a la configuración y registros proporcionados por la aplicación de escritorio.
    - Soporte para dispositivos antiguos o en desuso, maximizando el aprovechamiento del hardware existente.
  + Diseño y funcionalidad del Bot de Discord
    - Comandos básicos para moderación (gestión de usuarios, permisos, logs).
    - Herramientas de administración de servidores, como configuraciones automáticas y reportes.
    - Opciones de personalización para adaptarse a necesidades específicas de servidores de Discord.
  + Gestión de seguridad y datos
    - Hay que asegurar que toda la información manejada esté protegida contra accesos no autorizados.
    - Cumplir con las normativas de privacidad de datos aplicables a Discord y a los usuarios finales.
  + Soporte multiplataforma y escalabilidad
    - Garantizar una experiencia fluida y consistente entre las versiones de escritorio y móvil.
    - Facilitar actualizaciones y mejoras para mantener la compatibilidad con la API de Discord.
  + Facilidad de uso y documentación
    - Desarrollo de manuales detallados para instalación, configuración y uso.
    - Diseño de una interfaz intuitiva que reduzca la curva de aprendizaje, permitiendo a usuarios sin experiencia técnica aprovechar el producto.
* Justificación y análisis de la realidad

Justificación del proyecto:

* + Demanda creciente de bots personalizados:
    - Desarrollo A medida que las comunidades de Discord crecen en tamaño y complejidad, existe una necesidad cada vez mayor de bots que puedan adaptarse a las necesidades específicas de cada servidor. Sin embargo, muchas soluciones disponibles son genéricas o difíciles de personalizar sin conocimientos técnicos avanzados.
  + Control total del entorno y los datos:
    - La mayoría de los bots de Discord dependen de servicios de hosting remoto, lo que implica riesgos relacionados con la privacidad, la seguridad y la dependencia de terceros. Este proyecto permite a los usuarios gestionar el Bot localmente, ofreciendo control total sobre sus datos y funcionalidades.
  + Accesibilidad y facilidad de uso:
    - La posibilidad de usar el Bot tanto en aplicaciones de escritorio como móviles abre la puerta a una mayor flexibilidad. Además, el diseño multiplataforma facilita el acceso y uso de esta herramienta, incluso para aquellos con recursos limitados, como dispositivos antiguos o en desuso.
  + Aplicación práctica de conocimientos técnicos:
    - Este proyecto no solo resuelve un problema práctico, sino que también sirve como ejercicio para aplicar los conocimientos adquiridos en el ciclo formativo de Desarrollo de Aplicaciones Multiplataforma, integrando múltiples tecnologías y principios de diseño en un caso real de uso.

Análisis de la realidad:

* + Popularidad de Discord:
    - En los últimos años, Discord se ha posicionado como una de las plataformas de comunicación más utilizadas, con más de 300 millones de usuarios registrados y comunidades activas en todo el mundo. Esta popularidad genera un ecosistema ideal para el desarrollo de herramientas avanzadas como este proyecto.
  + Competencia y mercado actual:
    - Actualmente, existen varios bots con funcionalidades avanzadas (como MEE6, Dyno, o Carl-Bot), pero muchos de ellos operan bajo modelos de negocio freemium que limitan las funciones más atractivas a usuarios pagos. Este proyecto busca posicionarse como una alternativa flexible, de bajo costo y con posibilidades de personalización.
  + Avances tecnológicos y accesibilidad:
    - Con el auge de dispositivos móviles más potentes y el acceso a tecnologías multiplataforma, los usuarios esperan soluciones que puedan ser gestionadas desde cualquier lugar. Este proyecto se alinea con esa expectativa al ofrecer una integración completa entre escritorio y móvil.
  + Retos legales y éticos:
    - La gestión de datos personales y las actividades de moderación requieren un enfoque cuidadoso para cumplir con las normativas internacionales de privacidad, como el Reglamento General de Protección de Datos (GDPR) en Europa. El proyecto incluye mecanismos para garantizar que los datos se gestionen de manera segura y ética.
  + Crecimiento y evolución de la API de Discord:
    - Discord actualiza continuamente su API para incluir nuevas funciones y mejorar la experiencia del usuario. Este proyecto está diseñado con un enfoque escalable que facilita su mantenimiento y actualización frente a estos cambios.
* Marco legal
  + Reglamento General de Protección de Datos (GDPR):
  + Reglamento (UE) 2016/679 del Parlamento Europeo y del Consejo, de 27 de abril de 2016, relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos. Este reglamento establece las bases para la recopilación, almacenamiento y procesamiento de datos personales en la Unión Europea.
  + Ley Orgánica de Protección de Datos y Garantía de Derechos Digitales (LOPDGDD):
  + Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales, que adapta el GDPR al marco legal español y regula el tratamiento de datos personales en España.
  + Normas de uso de la API de Discord:
  + Discord API Terms of Service (Discord, Inc.). Este documento establece las condiciones legales para el uso de la API de Discord, incluyendo las limitaciones sobre el uso de datos, el respeto a los derechos de los usuarios y las políticas de privacidad.
  + Ley de Servicios de la Sociedad de la Información y de Comercio Electrónico (LSSI-CE):
  + Ley 34/2002, de 11 de julio, que regula las actividades relacionadas con los servicios de la sociedad de la información en España, incluyendo la provisión de servicios digitales y el comercio electrónico.
* Marco teórico

El marco teórico de este proyecto se centra en el uso de Java como lenguaje de desarrollo para todas las aplicaciones y la integración de una base de datos SQL como núcleo para la gestión de datos del Bot de Discord. Este enfoque combina la potencia de Java para aplicaciones multiplataforma con la robustez y escalabilidad de bases de datos relacionales.

* + Discord y su ecosistema de bots

Discord es una plataforma de comunicación en línea que permite a las comunidades gestionar servidores y automatizar procesos mediante bots. Estos bots interactúan con la plataforma a través de la API de Discord, que ofrece una amplia gama de funcionalidades.

* + - API de Discord:

La API de Discord proporciona un medio para crear bots personalizados utilizando bibliotecas como JDA (Java Discord API), que simplifica la integración con funciones como:

* + - * Gestión de mensajes, roles y usuarios.
      * Registro y respuesta a eventos en tiempo real.
      * Automatización de tareas como moderación, administración y envío de notificaciones.
    - Funciones principales del Bot:
      * Moderación: Bloqueo, silenciado y eliminación de usuarios o mensajes según las reglas del servidor.
      * Gestión de servidores: Cambios en la configuración, creación de roles y canales.
      * Registro de eventos: Almacenamiento de acciones y eventos en la base de datos SQL para auditorías y reportes.
      * Interactividad: Personalización de comandos y respuestas basados en las necesidades del usuario.
  + Desarrollo multiplataforma con Java y SQL

Java es un lenguaje ampliamente utilizado por su portabilidad, lo que lo hace ideal para desarrollar aplicaciones multiplataforma. La base de datos SQL permitirá una gestión centralizada y eficiente de los datos generados por el Bot.

* + - Herramientas principales
      * JavaFX (Aplicación de escritorio):

Framework para crear interfaces gráficas interactivas y modernas, combinando Java con XML (FXML) y CSS.

* + - * Android con Java (Aplicación móvil):

Desarrollo nativo de la aplicación móvil para permitir la interacción remota con el Bot y la base de datos.

* + - * Bases de datos SQL:
        1. MySQL o PostgreSQL: Bases de datos relacionales ideales para un entorno de producción.
        2. SQLite: Una opción ligera para entornos locales o móviles, permitiendo almacenamiento sin un servidor central.
      * Librerías de acceso a bases de datos:
        1. JDBC (Java Database Connectivity): Conector estándar para interactuar con bases de datos SQL desde Java.
        2. Herramientas ORM como Hibernate, si se requiere una capa adicional de abstracción para el manejo de datos.
    - Principios de desarrollo multiplataforma
      * Compatibilidad: Garantizar que la aplicación funcione en sistemas Windows, macOS y Linux para escritorio, y Android para móviles.
      * Conexión a la base de datos SQL:
        1. La aplicación de escritorio actuará como un servidor central, conectado a la base de datos SQL para gestionar los datos.
        2. La aplicación móvil se conectará al servidor utilizando APIs RESTful para interactuar de manera segura con los datos.
  + Base de datos SQL

La base de datos SQL será el núcleo del sistema, permitiendo almacenar y recuperar información de manera estructurada, eficiente y segura.

* + - Modelo relacional
      * Entidades principales:
        1. Usuarios: Datos básicos de los miembros del servidor, como ID, nombre y roles.
        2. Servidores: Información de cada servidor donde opera el Bot.
        3. Acciones y logs: Historial de eventos como baneos, cambios de roles y comandos ejecutados.
        4. Configuraciones: Preferencias del Bot para cada servidor.
    - Funciones de la base de datos
      * Persistencia de datos:
        1. Almacenamiento de logs de moderación para auditorías.
        2. Registro de comandos personalizados y configuraciones de los servidores.
      * Consultas optimizadas:
        1. Uso de índices para agilizar la búsqueda de datos en grandes volúmenes.
      * Seguridad de datos:
        1. Uso de roles y permisos en SQL para garantizar que solo usuarios autorizados puedan acceder o modificar la información.
    - Interacción con Java
      * Uso de JDBC para ejecutar consultas SQL desde las aplicaciones Java.
      * Posibilidad de usar herramientas ORM como Hibernate para simplificar la manipulación de datos y garantizar escalabilidad.
  + Arquitectura cliente-servidor

El proyecto se basa en una arquitectura cliente-servidor, en la que la aplicación de escritorio actúa como servidor y la base de datos SQL como repositorio central de datos.

* + - Componentes principales:
      * Servidor:

La aplicación de escritorio ejecutará el Bot, gestionará las configuraciones y registrará los eventos en la base de datos SQL.

* + - * Cliente:

La aplicación móvil permitirá acceder a los datos almacenados en el servidor, mostrando logs y configuraciones, y permitiendo algunas interacciones básicas.

* + - Comunicación entre cliente y servidor
      * APIs RESTful: Utilizando frameworks como SpringBoot para gestionar las solicitudes del cliente móvil hacia la base de datos.
      * Sockets TCP/IP: Para funcionalidades en tiempo real entre el servidor y la aplicación móvil.
  + Seguridad y escalabilidad
    - Seguridad de datos
      * Protección de la base de datos:
        1. Implementación de cifrado para datos sensibles, como contraseñas o tokens de Discord.
        2. Uso de conexiones seguras (SSL/TLS) para proteger las comunicaciones.
      * Gestión de usuarios:
        1. Control de acceso a las configuraciones y datos mediante autenticación basada en roles.
    - Escalabilidad
      * Optimización de la base de datos SQL
      * Despliegue en la nube:
        1. Posibilidad de migrar el servidor de escritorio y la base de datos SQL a un entorno en la nube, como AWS o Google Cloud, para soportar múltiples clientes móviles.
* Temporalidad

# **2. estudio de la viabilidad del sistema**

* Análisis económico (DAFO)
* Debilidades
  + El éxito de la aplicación depende de que la plataforma Discord se mantenga estable.
  + Tener aplicación de escritorio y de móvil en lugar de web puede suponer mayores costes de mantenimiento y desarrollo.
  + Curva de aprendizaje alta en caso de querer modificar el código de forma manual.
* Amenazas
  + Competencia: Existen otros bots de Discord con funcionalidades similares que pueden captar a los usuarios.
  + Seguridad: La gestión de datos personales de los usuarios debe ser especialmente cuidada.
  + Evolución tecnológica: La api utilizada y el propio Discord puede ir requiriendo actualizaciones periódicas completas de la aplicación
* Fortalezas
  + Ofrecer de forma condensada gran cantidad de funcionalidades puede atraer a muchos usuarios.
  + La existencia de aplicación móvil puede dar flexibilidad a los usuarios por poder editar funciones en cualquier lugar.
  + El ser un Bot de Discord permite que gran cantidad de usuarios puedan acceder al producto al estar ya dentro de la plataforma.
  + Gran cantidad de información comunitaria existente.
* Oportunidades
  + Crecimiento de la plataforma Discord.
  + Personalización de las funciones según las necesidades del cliente.
  + Posibles alianzas con servidores que promocionen el producto.
  + Monetización en funcionalidades premium.
* Plan de marketing

# **3. Descripción del entorno tecnológico**

* Descripción de todos los perfiles de usuario que habrá en la aplicación
  + La aplicación contará con los siguientes perfiles de usuario, cada uno con roles y funciones específicas:
    - Administrador del sistema:
    - Perfil destinado a los propietarios del servidor de Discord o usuarios avanzados.
    - Funciones principales:
      * Configuración avanzada del Bot (módulos, comandos personalizados, permisos).
      * Gestión de los datos almacenados en la base de datos.
      * Monitorización y auditoría de los registros de acciones del Bot.
  + Moderador del servidor:
    - Usuarios encargados de la moderación dentro de los servidores de Discord.
    - Funciones principales:
      * Uso de comandos de moderación (ban, kick, mute, etc.).
      * Acceso limitado a los registros de moderación.
      * Generación de reportes básicos sobre actividades en el servidor.
  + Usuario final:
    - Miembros regulares del servidor de Discord que interactúan con el Bot.
    - Funciones principales:
    - Uso de comandos básicos definidos por los administradores (consultas, juegos, encuestas).
    - Interacción con funciones no administrativas del Bot.
* Descripción de las tecnologías para cada perfil
  + Administrador del sistema
    - Aplicación de escritorio (JavaFX):
    - Herramienta principal para configurar y personalizar el Bot.
    - Interfaz gráfica diseñada para acceder a funciones avanzadas como:
      * Gestión de módulos y comandos.
      * Visualización de registros almacenados en la base de datos SQL.
      * Control del estado del Bot en tiempo real.
    - Base de datos SQL (MySQL o PostgreSQL):
      * Almacenamiento seguro de configuraciones, registros y datos de usuarios.
      * Soporte para consultas avanzadas y reportes.
  + Moderador del servidor
    - Aplicación móvil (Android en Java):
      * Proporciona acceso rápido y remoto a las funcionalidades del Bot.
      * Funciones clave:
        + Consultar registros básicos de moderación.
        + Ejecutar comandos de moderación desde dispositivos móviles.
    - Conexión al servidor mediante API RESTful:
    - Uso de endpoints seguros para consultar información o ejecutar comandos en el Bot.
  + Usuario final
    - Bot de Discord (JDA - Java Discord API):
      * Interacción directa con los usuarios dentro de Discord.
      * Comandos personalizados accesibles desde la interfaz de Discord (por ejemplo, !ayuda, !stats).
      * Respuestas inmediatas mediante WebSockets.
    - Servidor local o en la nube:
      * Ejecuta el Bot y procesa las solicitudes de los usuarios finales.
      * Escalable para soportar múltiples usuarios y peticiones concurrentes.

# **4. Especificación de requisitos**

* Requisitos funcionales

Los requisitos funcionales definen las funcionalidades que el sistema debe ser capaz de realizar. Estos requisitos están directamente relacionados con las operaciones y tareas que el Bot y las aplicaciones asociadas deben ejecutar.

* Gestión del Bot
  + El sistema debe permitir ejecutar el Bot de Discord tanto en una aplicación de escritorio como en una móvil.
  + El Bot debe ser capaz de realizar acciones de moderación, como banear, silenciar o kickear usuarios.
  + Debe ser posible agregar, editar y eliminar comandos personalizados del Bot.
  + El Bot debe registrar todas las acciones de moderación (logs de acciones) en una base de datos SQL.
* Interfaz de usuario
  + La aplicación de escritorio debe tener una interfaz gráfica para permitir la configuración del Bot, incluida la personalización de comandos y módulos.
  + Los administradores deben poder ver un registro de eventos del servidor de Discord y acceder a los detalles de las acciones realizadas por los moderadores.
  + La aplicación móvil debe permitir a los usuarios conectarse al Bot y ver las configuraciones y registros almacenados en la versión de escritorio.
* Base de datos
  + El sistema debe almacenar los datos del Bot (configuraciones, logs, registros de usuarios) en una base de datos SQL.
  + Los administradores deben poder consultar y actualizar las configuraciones del Bot a través de la aplicación de escritorio.
  + La base de datos debe ser capaz de manejar consultas optimizadas y mantener la integridad de los datos.
* Comunicaciones en tiempo real
  + El sistema debe permitir la comunicación en tiempo real entre el Bot y los usuarios a través de WebSockets.
  + La aplicación móvil debe conectarse al servidor para ejecutar comandos de moderación y obtener información relevante.
* Seguridad y privacidad
  + El sistema debe garantizar la seguridad de los datos almacenados y transmitidos mediante el uso de cifrado SSL/TLS.
  + Solo los administradores y moderadores deben tener acceso a funciones avanzadas del Bot.
* Multiplataforma
  + La aplicación de escritorio debe ser compatible con Windows, macOS y Linux.
  + La aplicación móvil debe ser compatible con dispositivos Android.
  + El sistema debe permitir la modificación y personalización de las funciones del Bot a través de ambas plataformas (escritorio y móvil).
* Requisitos no funcionales

Los requisitos no funcionales especifican las características del sistema que no están relacionadas con funciones específicas, sino con el rendimiento, la seguridad y la usabilidad. Estos requisitos son fundamentales para garantizar la calidad y la eficiencia del sistema.

* Rendimiento
  + El sistema debe ser capaz de manejar múltiples solicitudes simultáneas sin afectar el rendimiento.
  + La aplicación debe ser capaz de procesar y almacenar eventos en tiempo real sin demoras notables en la respuesta del Bot.
* Escalabilidad
  + La aplicación debe ser escalable para permitir la adición de nuevas funcionalidades sin comprometer el rendimiento o la estabilidad.
  + La base de datos SQL debe ser capaz de manejar grandes volúmenes de datos a medida que el número de usuarios y servidores gestionados crece.
* Usabilidad
  + La aplicación debe ser fácil de usar, con interfaces intuitivas que permitan a los administradores configurar y gestionar el Bot sin necesidad de conocimientos técnicos avanzados.
  + El diseño de la interfaz debe ser coherente y proporcionar retroalimentación clara al usuario sobre las acciones realizadas.
* Seguridad
  + Los datos sensibles, como contraseñas y tokens de Discord, deben ser almacenados de forma segura mediante cifrado.
  + El sistema debe seguir las mejores prácticas de seguridad para proteger los datos personales de los usuarios conforme a las normativas de privacidad (GDPR, LOPDGDD).
* Mantenimiento y soporte
  + El sistema debe ser fácilmente mantenible, con documentación detallada para la instalación, configuración y actualización del Bot y las aplicaciones asociadas.
  + Se deben ofrecer actualizaciones periódicas para mantener la compatibilidad con las últimas versiones de la API de Discord y mejorar las funcionalidades del Bot.
* Compatibilidad
  + El sistema debe ser compatible con versiones recientes de los sistemas operativos de escritorio (Windows, macOS, Linux) y dispositivos móviles (Android).
  + Las aplicaciones deben ser compatibles con versiones de Java recientes y las bibliotecas necesarias para el funcionamiento del Bot y la base de datos SQL.

# **5. Modelo entidad / relación (pendiente de revisión de la versión más reciente de la API)**

* Explicar por qué ha salido ese E/R, de dónde salen las Entidades y de dónde salen las Relaciones.
* Plasmar el modelo E/R
* Explicar el origen de los atributos
* Explicar por qué se han elegido esas claves primarias.
* Modelo Relacional

# **6. Diagrama de procesos (pendiente de implementación)**

* Casos de Uso
* Diagrama de clases

# **7. Diseño Interfaz (modificable en la fase de implementación)**

* Diagramación y prototipado del diseño gráfico de la interfaz(pendiente de decisión final sobre tecnologías de visualización a usar)

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Captura de pantalla de un celular

Descripción generada automáticamente

* Guías de estilo (pendiente de agregar códigos de colores, etc.)

Teniendo en cuenta que es una aplicación dedicada al manejo de bots utilizados en la red social Discord se intentará en la medida de lo posible imitar los colores de su propia interfaz, es decir, una escala de grises en su modo oscuro con colores de los botones destacados en rojo para las opciones importantes como eliminar o borrar y azul para las que se puedan revertir o sean de movimiento entre ventanas.

Captura de pantalla de computadora

Descripción generada automáticamente

* Mapa de navegación

# **8. Pruebas (pendiente de implementación)**

* Pruebas unitarias
* Pruebas de integridad

# **9. Manual de usuario (pendiente de implementación)**

* Manual del cliente
* Manual de instalación y despliegue

# **10. CONCLUSIONES**

# **11. Anexo**

* Bibliografía
* Referencias a texto y/o web de ayuda
* Referencia de imágenes