UNIVERSIDAD DE MURCIA

AppChat
TECNOLOGÍAS DE DESARROLLO DE SOFTWARE

Autores:
Alberto Zapata Mira
Laura Cañada Rubio

5 de mayo de 2025

Índice general

Historias de Usuario

HU1: Registro de nuevo usuario

Como: usuario nuevo

quiero: poder registrarme con mi número de telefono y una contraseña

para: poder acceder a la aplicación de mensajería

Criterios de aceptación:

- El usuario puede introducir un número valido y una contraseña segura.
- El sistema valida el número y muestra errores claros si es incorrecto.
- El sistema informa si el número ya está registrado.

HU2: Inicio de sesión

Como: usuario registrado

quiero: iniciar sesión con mis credenciales

para: acceder a mis contactos y conversaciones

Criterios de aceptación:

- El usuario puede ingresar su número y contraseña correctamente.
- Si las credenciales son inválidas, el sistema muestra un mensaje de error.
- El acceso es denegado hasta que las credenciales sean válidas.

HU3: Añadir contacto

Como: usuario registrado

quiero: poder añadir nuevos contactos mediante su identificador

para: poder comunicarme con ellos desde la aplicación

Criterios de aceptación:

- El sistema permite buscar usuarios por identificador.
- Se muestra un mensaje si el usuario no existe.
- El contacto se añade a la lista del usuario tras la confirmación.

HU4: Crear grupo

Como: usuario registrado

quiero: poder crear grupos de conversación

para: mantener conversaciones con múltiples contactos al mismo tiempo

Criterios de aceptación:

- El usuario puede nombrar el grupo y seleccionar contactos.
- El grupo se guarda con la información y miembros indicados.
- Se puede ver el grupo creado en la lista de grupos.

HU5: Enviar mensaje

Como: usuario registrado

quiero: poder enviar mensajes de texto a mis contactos o grupos

para: comunicarme con mis contactos.

Criterios de aceptación:

- El usuario puede escribir y enviar mensajes.
- Los mensajes se muestran inmediatamente en el chat.
- Los destinatarios reciben el mensaje cuando se conectan.

HU6: Ver historial de conversaciones

Como: usuario registrado

quiero: poder ver el historial de mensajes de mis conversaciones

para: seguir el contexto de la conversación

Criterios de aceptación:

- El historial de cada chat se carga al abrirlo.
- El usuario puede desplazarse y leer mensajes anteriores.
- Los mensajes aparecen en orden cronológico.

HU7: Gestionar Contactos y Grupos

Como: usuario registrado

quiero: poder gestionar mis contactos y grupos

para: mantener mi lista de contactos y grupos actualizada

Criterios de aceptación:

- El usuario puede acceder a un formulario de edición.
- Los cambios se guardan y se reflejan inmediatamente.
- El sistema valida los campos antes de guardar.

HU8: Cerrar sesión

Como: usuario registrado

quiero: poder cerrar sesión de forma segura

para: proteger el acceso a mi cuenta

Criterios de aceptación:

- Hay una opción accesible para cerrar sesión.
- Al cerrar sesión se redirige al usuario a la pantalla de inicio.
- No se puede acceder a funciones privadas tras cerrar sesión.

HU9 - Convertirse en usuario premium

Como: usuario registrado

quiero: poder convertirme en usuario premium pagando una suscripción

para: acceder a funciones adicionales como la exportación de mensajes

Criterios de verificación:

- El sistema debe permitir al usuario registrarse como premium mediante el pago de una suscripción anual.
- El usuario premium debe tener acceso a funciones adicionales como la exportación de mensajes en PDF.

HU10 - Buscar mensajes

Como: usuario registrado

quiero: buscar mensajes por fragmento de texto, nombre de contacto o número de teléfono

para: encontrar fácilmente los mensajes que necesito

Criterios de verificación:

- El sistema debe permitir buscar mensajes enviados o recibidos por el usuario filtrando por fragmento de texto, nombre del contacto o número de teléfono.
- Los resultados de la búsqueda deben mostrarse en forma de lista.
- El sistema debe permitir combinar varios criterios de búsqueda (por ejemplo, texto y contacto).

HU11 - Exportar mensajes a PDF (solo premium)

Como: usuario premium

quiero: exportar mis conversaciones a un archivo PDF

para: tener un registro de los mensajes intercambiados

Criterios de verificación:

- El sistema debe permitir al usuario exportar sus conversaciones a PDF.
- El archivo PDF debe incluir los nombres de los participantes, el contenido del mensaje, y la fecha y hora de cada mensaje.
- El archivo PDF debe generarse correctamente y descargarse en el dispositivo del usuario.

Diagrama de Secuencia: Añadir Contacto a Grupo

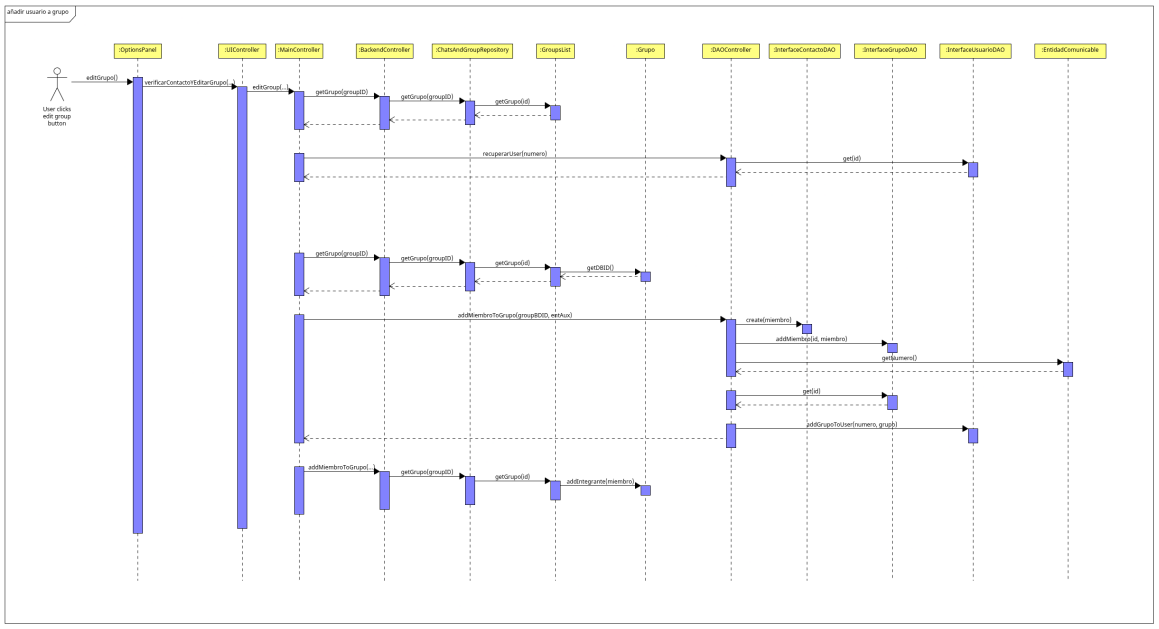


Figura 1: Secuencia para añadir un contacto a un grupo

Diagrama de Dominio UML

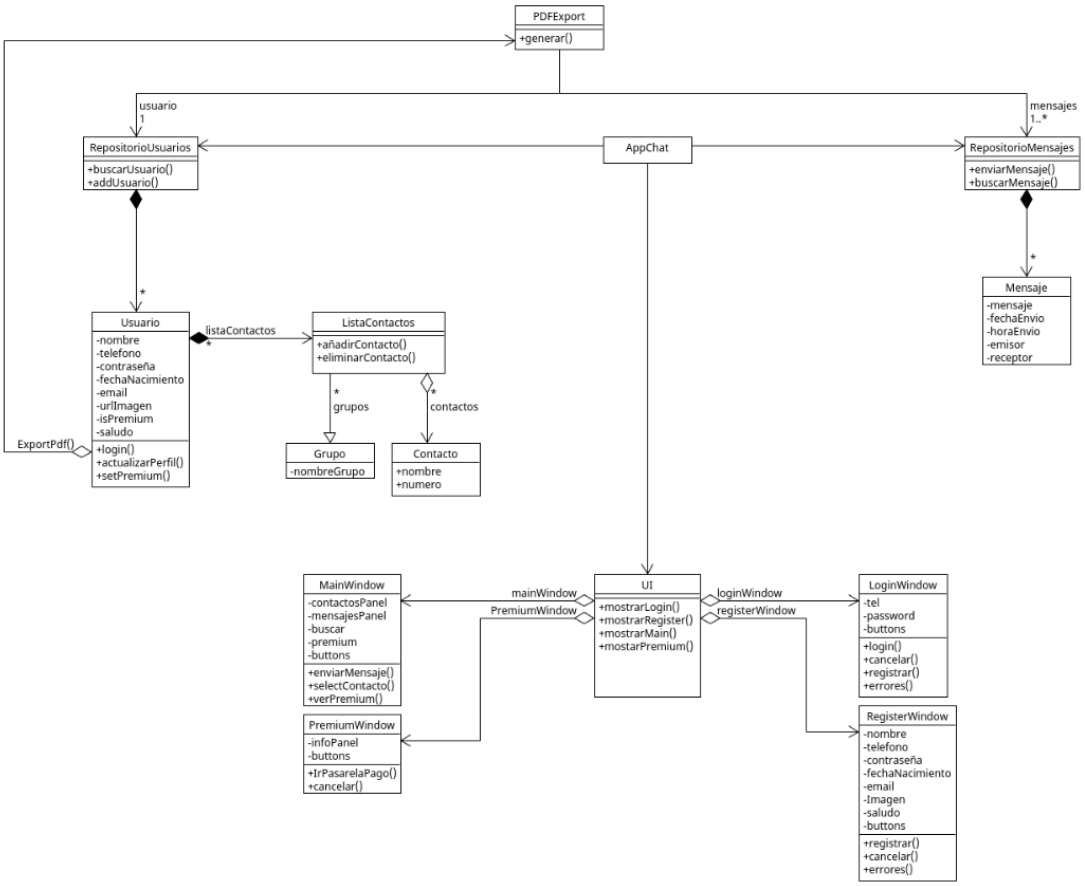


Diagrama de dominio - versión inicial

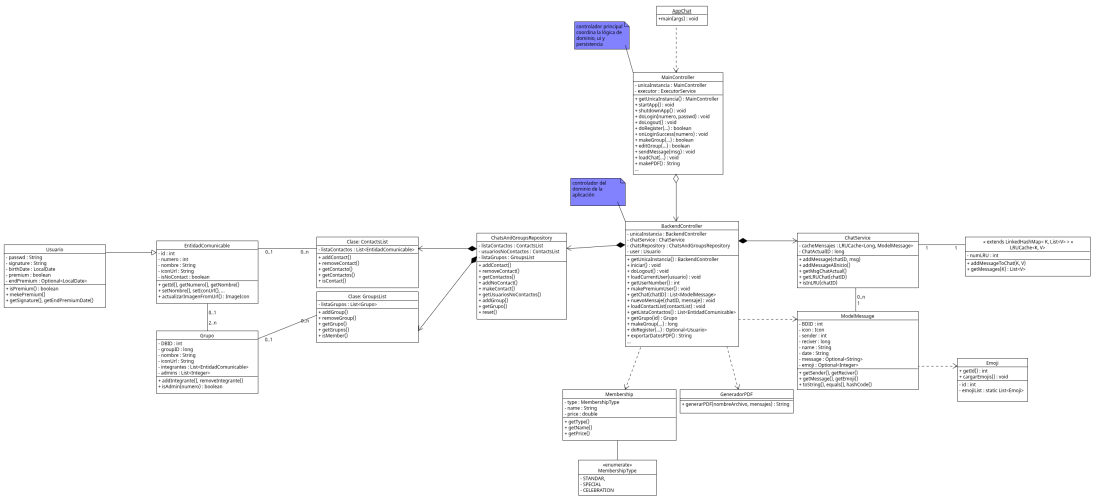


Diagrama de dominio - versión final

Arquitectura de la Aplicación

1. Introducción

El objetivo del sistema es permitir el envío de mensajes entre usuarios.
Para ello, la aplicación se organiza en distintas capas funcionales que separan la presentación, la lógica de negocio y persistencia, las cuales se gestionan a través de controladores específicos. Esta estructura responde a una necesidad de organización del código y de distribución de responsabilidades entre componentes.

2. Arquitectura

La aplicación implementa una arquitectura en capas estructurada según el patrón Modelo-Vista-Controlador (MVC). Esta combinación permite una separación clara de responsabilidades y facilita el mantenimiento y la escalabilidad del sistema.

- **Vista:** La interfaz de usuario es gestionada por la clase `UIController`, que captura las interacciones del usuario.
- **Controladores:** Las clase `BackendController` se encargan de coordinar la lógica del sistema.
- **Modelo y persistencia:** `DAOController` actúa como intermediario entre la lógica de negocio y los datos almacenados.
- **Coordinación:** La clase `MainController` orquesta la comunicación entre la vista, la lógica de negocio y la persistencia.

A nivel estructural, la aplicación respeta la arquitectura en capas tradicional:

1. **Capa de presentación:** interacción con el usuario.
2. **Capa de lógica:** procesamiento de acciones y flujo de trabajo.
3. **Capa de persistencia:** gestión de datos y almacenamiento.

Esta combinación proporciona modularidad y una clara organización del código fuente.

3. Componentes principales

Interfaz de usuario (UI): Componente que gestiona la presentación y la interacción con el usuario, desarrollado con Java Swing. Las entradas del usuario son capturadas y enviadas al controlador principal para su procesamiento.

Lógica de negocio (Dominio): La gestión del dominio se realiza en su mayor parte a través del `BackendController`, que centraliza la lógica asociada a las reglas del sistema, el procesamiento de operaciones sobre entidades y la coordinación de servicios internos. Esta capa encapsula la lógica de negocio sin depender de detalles de interfaz o persistencia, manteniendo la coherencia y autonomía del modelo. La lógica del dominio se desarrolla siguiendo principios de encapsulamiento y separación de responsabilidades, permitiendo que la aplicación evolucione sin comprometer su estructura.

Persistencia de datos: Gestionada mediante `DAOController`, responsable del acceso y almacenamiento de datos.

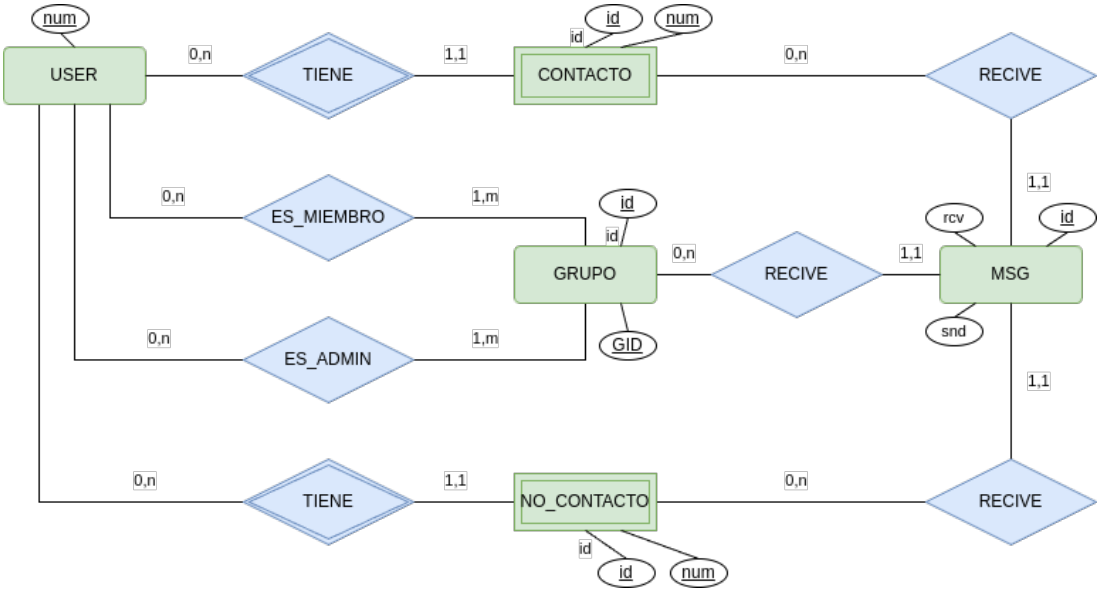
MainController: Responsable de la coordinación general del sistema. Su función es actuar como punto de entrada desde la interfaz, delegando las operaciones correspondientes a otras capas (como la lógica de negocio y la persistencia). Aunque participa en el flujo de datos y en la toma de decisiones, su propósito es más bien estructural, no lógico.

4. Flujo general de datos

- El usuario realiza una acción en la interfaz (`UIController`).
- Esta acción es enviada a `MainController`, que determina la lógica a aplicar.
- La lógica de negocio se delega a `BackendController`, donde puede incluirse validación y procesamiento.
- Si es necesario acceder a los datos, la solicitud se redirige a `DAOController`.
- El resultado se devuelve a `MainController` y se refleja en la interfaz.

5. Consideraciones técnicas y decisiones

- Los grupos utilizan un identificador de 10 dígitos en el dominio. Sin embargo, dado que el servicio de persistencia genera su propio identificador, se optó por utilizar este último para la persistencia, manteniendo al mismo tiempo el identificador de 10 dígitos en el dominio.
- La persistencia se modeló siguiendo el siguiente diseño conceptual:



Diseño conceptual de la persistencia

No obstante, debido a las particularidades del `Servicio de Persistencia H2`, fue necesario realizar ciertas adaptaciones, especialmente en las relaciones `n,m`, que se implementaron como relaciones `1,n` desde cada una de las entidades participantes.

- No se implementó una caché DAO, ya que objetos como la lista de grupos y contactos se cargan al iniciar sesión, y los mensajes disponen de su propia caché, implementada mediante un algoritmo LRU. Este enfoque permite:
  - Asignar más recursos a un chat cuanto más tiempo se permanezca en él.
  - Liberar recursos de chats que no se han utilizado recientemente.
- Con el objetivo de facilitar el desarrollo, la aplicación incorpora un sistema de registro de eventos (logger), cuya verbosidad puede configurarse mediante un parámetro que se pasa al ejecutar la aplicación. Este admite los siguientes niveles:
  - `DEBUG`
  - `INFO`
  - `WARN`
  - `ERROR`
  - `OFF` (nivel por defecto si no se especifica ninguno)
  - `TRACE`

  Los mensajes generados se muestran por terminal y, además, se almacenan en el directorio `logs/`.

Patrones de diseño utilizados

A continuación se describen los patrones de diseño identificados en la implementación del proyecto.

1. MVC (Modelo-Vista-Controlador)

La aplicación combina una estructura en capas con el patrón Modelo-Vista-Controlador (MVC). Esta integración permite organizar el sistema de forma comprensible, manteniendo la separación entre los distintos tipos de responsabilidades.

- **Vista:** `UIController` gestiona la interfaz de usuario y captura las interacciones del usuario.
- **Controladores:** `BackendController` implementa la lógica del dominio, y `MainController` coordina la comunicación entre componentes.
- **Modelo y persistencia:** `DAOController` proporciona acceso estructurado a los datos persistentes del sistema.

Este patrón se encuentra implementado dentro de una arquitectura por capas clásica (presentación, lógica, persistencia).

2. Abstract Factory

Este patrón creacional permite crear familias de objetos relacionados sin especificar sus clases concretas. En el proyecto, `AbstractFactoriaDAO` define métodos abstractos para obtener diferentes DAOs, y `FactoriaDAO` proporciona su implementación.

3. Factory Method

Utilizado para delegar la creación de objetos a subclases. En el proyecto, `FactoriaDAO` implementa métodos como `getUsuarioDAO()` o `getGrupoDAO()` para instanciar objetos concretos de acceso a datos.

4. DAO (Data Access Object)

Este patrón encapsula el acceso a los datos y separa la lógica de persistencia del resto de la aplicación. Se encuentra implementado en `DAOController`, `UsuarioDAO`, entre otros.

5. Facade

Este patrón estructural proporciona una interfaz unificada para un conjunto de interfaces en un subsistema. En el proyecto, `MainController` actúa como fachada, centralizando la comunicación entre la vista, la lógica de negocio y la persistencia.

6. Singleton

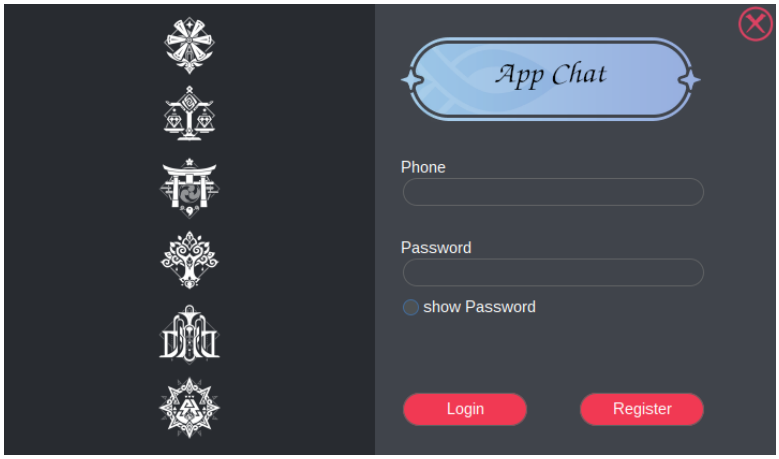
Asegura que una clase tenga una única instancia accesible globalmente. Se aplica en clases clave como `MainController`, `BackendController`, `DAOController` y `UIController`, así como en los adaptadores de los DAO, como `ContactoDAO`, mediante un constructor privado y un método estático `getUnicaInstancia()`.

Manual de Usuario

Este manual proporciona una guía básica para el uso de la aplicación, explicando las funcionalidades disponibles y cómo acceder a ellas.

1. Inicio de la aplicación

Al ejecutar la aplicación, se muestra una pantalla de inicio de sesión con opciones para iniciar sesión o registrarse.

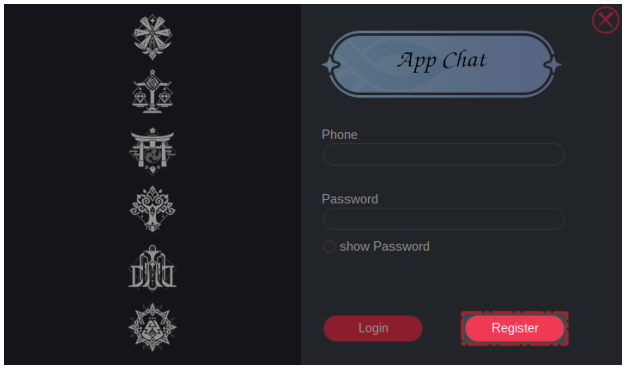


Pantalla de inicio de sesión

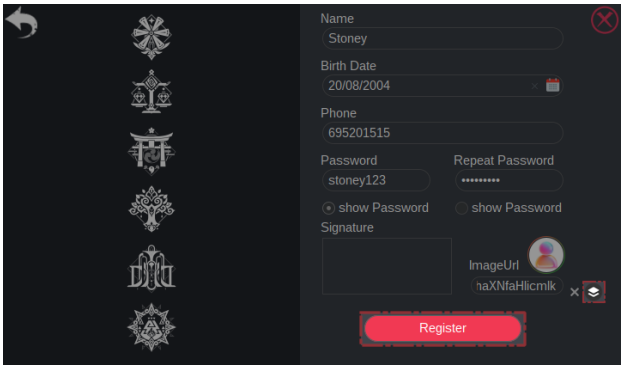
2. Registro de un nuevo usuario

Para crear una cuenta, el usuario debe seleccionar la opción `Register` y completar el formulario.

- Introducir el número de teléfono, nombre de usuario y contraseña.
- Para modificar la imagen de perfil, introducir la url de la imagen deseada, y pulsar el botón de la derecha.
- Pulsar el botón `register`.



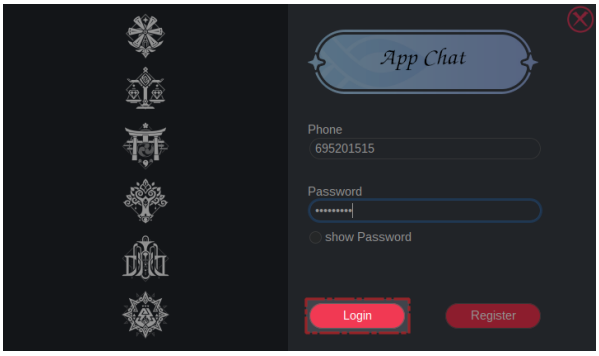
Formulario de registro - Parte 1



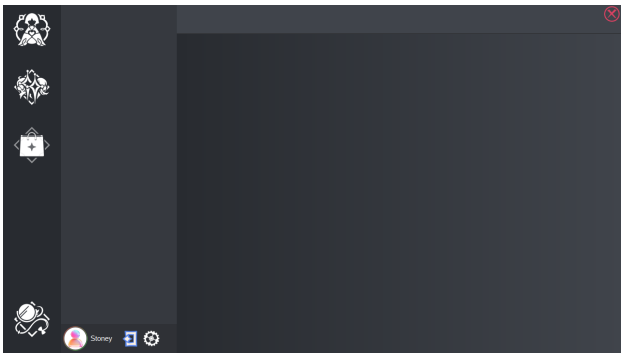
Formulario de registro - Parte 2

3. Inicio de sesión

Para iniciar sesión, el usuario debe introducir sus credenciales en la pantalla de inicio de sesión y pulsar el botón `Login`.



Pantalla de inicio de sesión



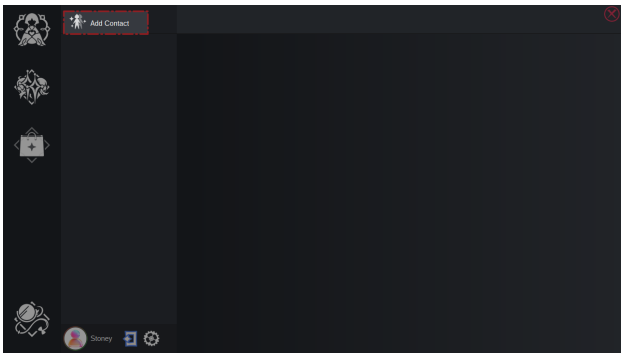
Pantalla principal

Tras iniciar sesión, el usuario accede a la pantalla principal de la aplicación, desde donde puede ver sus contactos y grupos, así como realizar otras operaciones.

4. Añadir un nuevo contacto

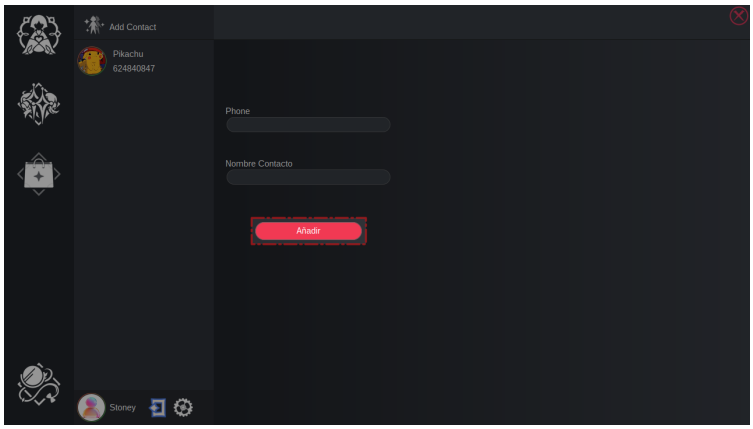
Para añadir un nuevo contacto, el usuario debe pulsar el botón para acceder a la lista de contactos y luego seleccionar la opción `Add Contact`.

Pantalla principal



Lista de usuarios

A continuación, el usuario debe introducir el número de teléfono del contacto, el nombre que desea asignarle y pulsar el botón `Añadir`.



Formulario para añadir contacto

Observaciones Finales

Aspectos relevantes, dificultades enfrentadas, mejoras futuras, etc.

Estimación de tiempo dedicado

- Alumno A: 20 horas
- Alumno B: 18 horas
- ...