PRACTICA FINAL AppChat

Curso 2019/2020

Nombre: José Antonio García Cartagena

Grupo: 1.\*

DNI: 48700263V

Correo: [joseantonio.garciac@um.es](mailto:joseantonio.garciac@um.es)

**Indice**

1. Diagrama de clases del dominio.

2. Un diagrama de colaboración o secuencia para la operación añadir un nuevo contacto a un grupo creado por el usuario.

3. Una breve explicación de la arquitectura de la aplicación y decisiones de diseño que se consideren de interés para la comprensión del trabajo.

4. Explicación de los patrones de diseño utilizados.

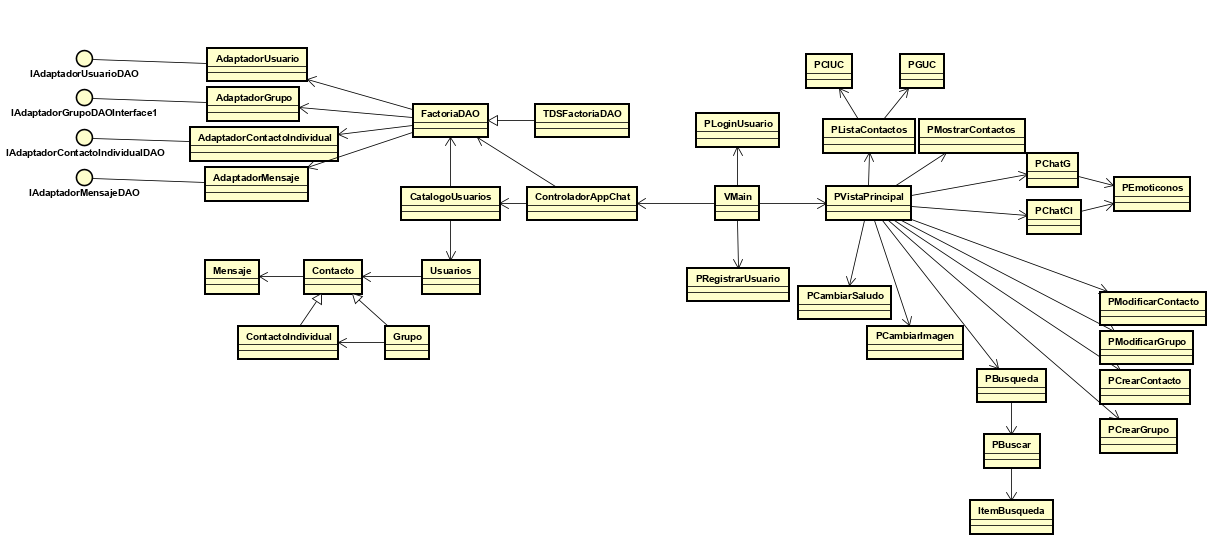
5. Explicación sobre los componentes utilizados.

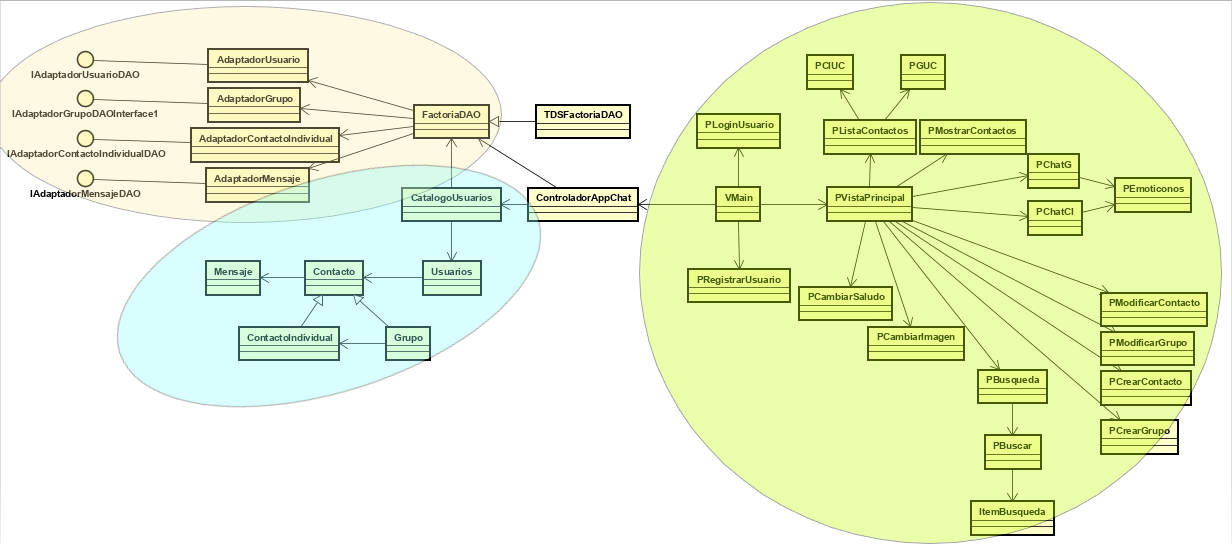
6. Tests unitarios implementados

7. Un pequeño manual de usuario que explique cómo usar la aplicación

8. Observaciones finales (deben incluir una estimación del tiempo dedicado)

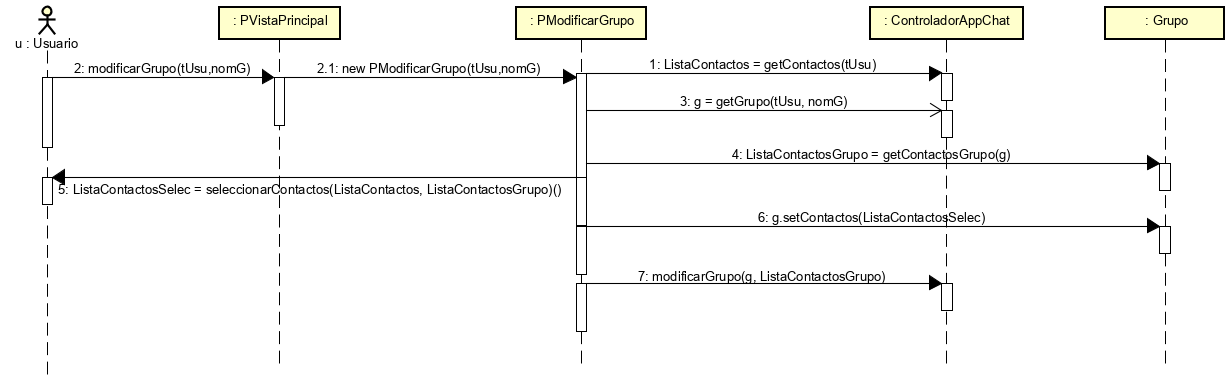
**Diagrama de Clases**



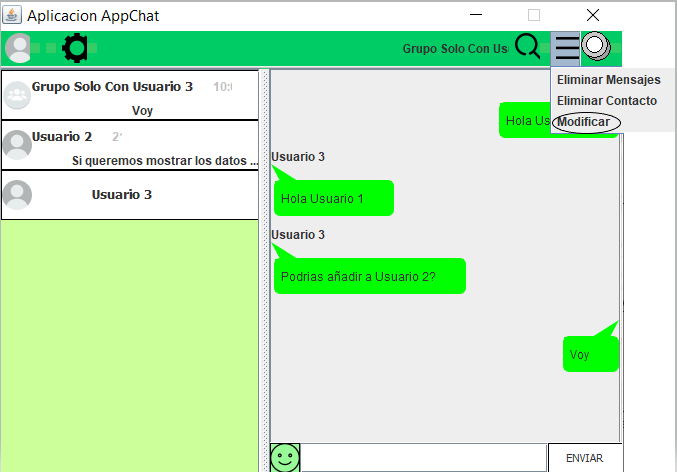


Dentro del Diagrama de Clases podemos ver que El proyecto esta diferenciado en tres capas: Presentación(Verde), Lógica de Negocio(Azul) y Almacenamiento(Naranja).

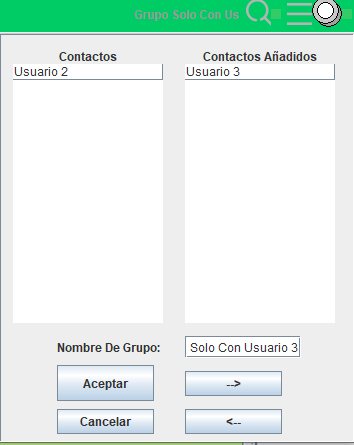
**Diagrama de Secuencia para la operación añadir un nuevo Usuario a un grupo**



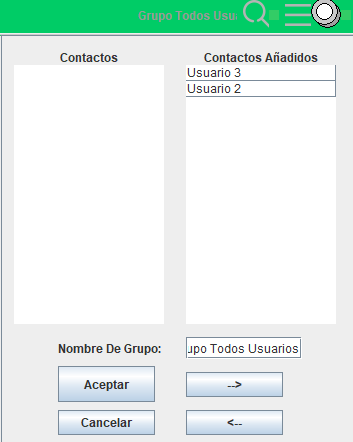
Para la operación de añadir un nuevo Usuario, se pasan como parámetros el móvil del Usuario que está utilizando la aplicación junto al nombre del Grupo que está seleccionado en la capa de presentación, en la clase PVistaPrincipal. Esta clase, a su vez llama a otra clase que gestiona la vista de Modificación de Grupo Mostrando 2 tablas. Una a la izquierda con los contactos que no se encuentran en el grupo (‘ListaContactos’ sin los contactos de ‘ListaContactosGrupo’) y otra a la derecha Con los Contactos del Grupo. El Usuario selecciona los Contactos que quiera añadir o eliminar del grupo y los almacena en una lista (‘ListaContactosSelec ‘), siendo ésta, la nueva lista de contactos del Grupo. Para que los cambios se hagan en la capa de Almacenamiento se llama a la función modificarGrupo con los argumentos del grupo a modificar y su antigua lista de contactos. En este método se gestiona tanto la eliminación como asignación nueva de Contactos y cambio de nombre.



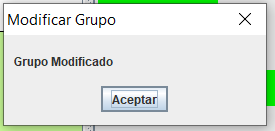
Vista de PVistaPrincipal



Vista de PModificarGrupo



PModificadoGrupo con Usuario 2 añadido



Ventana emergente al haber modificado un grupo

**Breve explicación de la arquitectura de la aplicación y decisiones de diseño que se consideren de interés para la comprensión del trabajo.**

El proyecto está dividio en 3 capas:

**Almacenamiento:**

Se ha utilizado un servicio de Persistencia propuesto en esta asignatura compatible tanto para un motor local de base de datos llamado h2 (que es la opción que se ha elegido) como una versión para servidor MySQL.

Este Servidor de Persistencia se basaba en la creación de entidades y dentro de ella poder añadir y eliminar propiedades, de esta manera era muy sencillo tanto registrar objetos como obtenerlos. Los diferentes tipos de entidades creadas son las mismas que las de la capa de Lógica de negocio.

Se han creado clases de registro y recuperación para cada uno de los objetos, siendo Usuario, ContactoIndividual, Grupo y Mensaje.

**Modelo de Negocio**

Las principales clases son Usuario, ContactoIndividual y Grupo(que heredan de Contacto) y Mensaje. Un Usuario tiene una lista de Contactos. Estos contactos pueden ser tanto ContactosIndividuales como Grupos. Si son ContactosIndividuales, tienen a un usuario asociado, el usuario del ContactoIndividual. Si Son Grupos, tienen una lista de ContactosIndividuales que forman los integrantes del Grupo. Todos los Contactos tienen mensajes, y estos están asociados tanto al Contacto que lo tienen como el Usuario que lo ha escrito.

**Presentación**

Para la inicialización de esta capa se ha utilizado Java Swing y la extensión SceneBuilder de Eclipse.

Se ha creado una clase que extiene de JFrame llamada VMain. Esta función solo crea un JFrame y se encargar del registro y login de un Usuario.

Una vez validado o registrado el Usuario, el JPanel principal cambia por la clase PVistaPrincipal que extiende de JPanel. Este panel está compuesto por tres grandes partes. En la parte superior se Encuentra un JMenuBar desde el que se gestiona el cambio de Saludo e Imagen de Usuario; Configuración del Usuario, como ser Premium, Mostrar Contactos o ser Premium; Información del Contacto seleccionado; Búsqueda por palabras clave; gestión del Contacto, como Eliminar Contacto, Mensajes o Modificar; y el componente creado para la importación de MensajesWhatsapp.

En la Parte central del Panel se encuentra un JSplitPane compuesto por dos paneles:

El Panel de la izquierda se compone la lista de Contactos más recientes, que puede ser cambiada por la lista de Contactos.

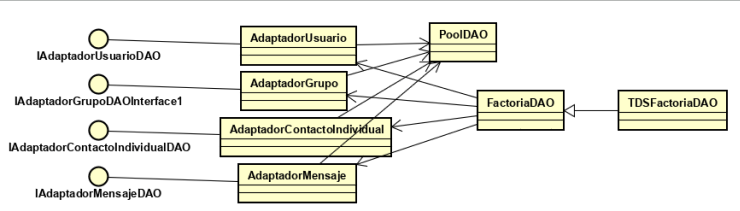
El Panel de la derecha se inicializa vacío, pero es el más versátil. Su panel más utilizado es el del Chat seleccionado del panel de la Izquierda del JSplitPane, pero también se utiliza para mostrar la creación de Contactos, modificación de los mismos o panel de búsquedas.

Todos estos paneles son clases que extienden de JPanel.

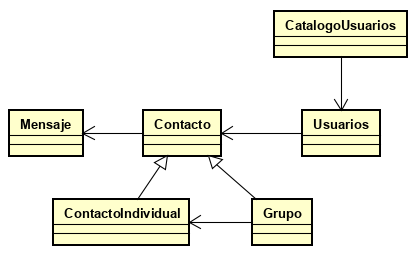
Dentro de los Paneles de Chat(PChatCI/PChatG) podemos encontrar la librería de BubbleText y un panel emergente de emoticonos(PEmoticonos).

**Explicación de los patrones de diseño utilizados**

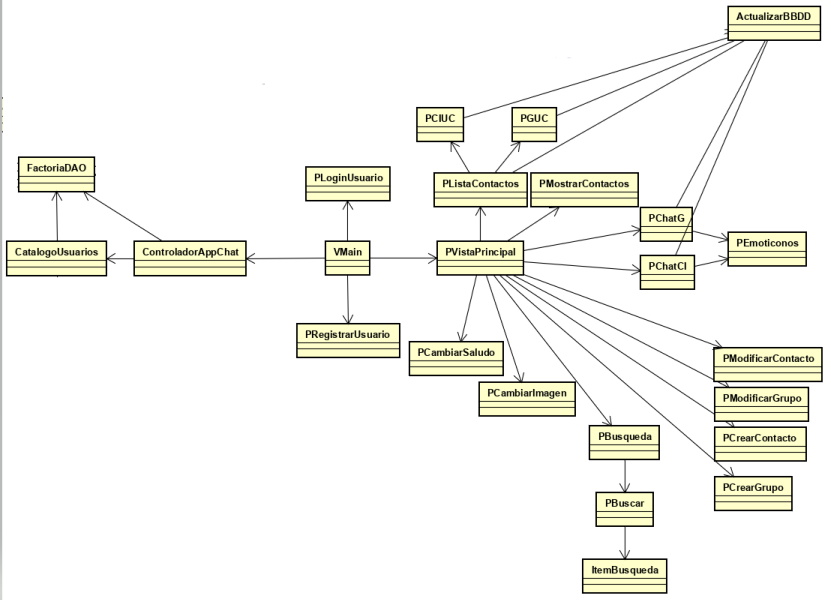
Para la gestión de la **capa de Almacenamiento**, para todos los adaptadores, como existen relaciones entre estas clases, para la recuperación de un objeto de la base de datos se ha implementado una clase Pool que consiste en almacenar dichos objetos sin sus relaciones con los demás objetos. De esta manera no se generan bucles de obtención de objetos relacionados. Todas estas clases de registro tienen opción de obtener cualquier objeto del Pool. Todas las clases implementadas, incluida la clase Pool son de única Instancia o singleton de inicialización perezosa. Gracias a esto la clase Pool siempre tiene los mismos objetos para todas las clases. Para una mayor gestión de esta capa, se ha implementado una clase Factoria Abstracta que inicializa todos las Clases creadas y devuelve instancias de las mismas.



Para la el **Modelado de Negocio**, como se puede ver, desde un Usuario se puede ver los Contactos, y de ellos los mensajes. Es por ello, que se ha creado una clase CatalogoUsuario singleton desde la que se puede acceder a cualquier Usuario. Esta clase también tiene una instancia a la clase Factoria de la capa de Almacenamiento para construir todos los objetos.



En cuanto a la **Presentación**, tanto para registrar Objetos(Factoria) como para obtenerlos(CatalogoUsuarios) se ha creado una clase Maestra llamada ControladorAppChat siendo ésta singleton. Esta clase contiene todas las funciones de modificación y obtención de todos los Objetos del proyecto. Como este programa es asíncrono, se ha implementado una clase que funciona como mediador, llamada ActualizarBBDD singleton que implementa un Timer de 10 segundos. Cuando el timer se activa, actualiza los objetos PoolDAO, el catálogo y los paneles de Chat, Contactos y la información de cada uno de los Contactos.



**Explicación sobre los componentes utilizados**

Se ha creado un componente llamado Luz. Este componente tiene la función de obtener un archivo .txt mediante un JFileChooser que es pasado al ControladorAppChat y este al ControladorCargarMensajes. Este texto deberá contener mensajes escritos. Si se han cargado mensajes nuevos ControladorAppChat, al implementar un listener, registra los nuevos mensajes y llama al Panel Chat Respectivo para actualizarlo. Es decir, el componente tiene la funcionalidad de importar Mensajes.

**Tests unitarios implementados**

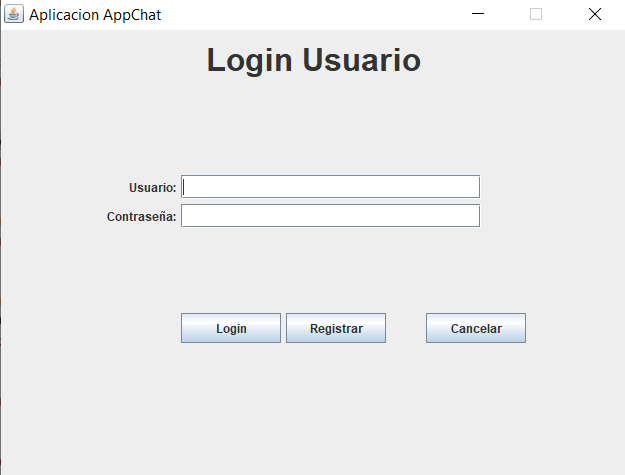
Los test unitarios son implementados por JUnit y son una forma de comprobar el correcto funcionamiento de una unidad de código.

Se ha creado una clase TestControladorAppChat que ha servido para crear Test de los métodos más característicos de la clase ControladorAppChat.

Para una mayor agilidad se ha implementado unas clases que sirven para una ordenación al ejecutar los tests. Se ha hecho de esta manera porque lo que se ha comprobado es la funcionalidad de escritura en la base de datos y el tratamiento de los objetos.

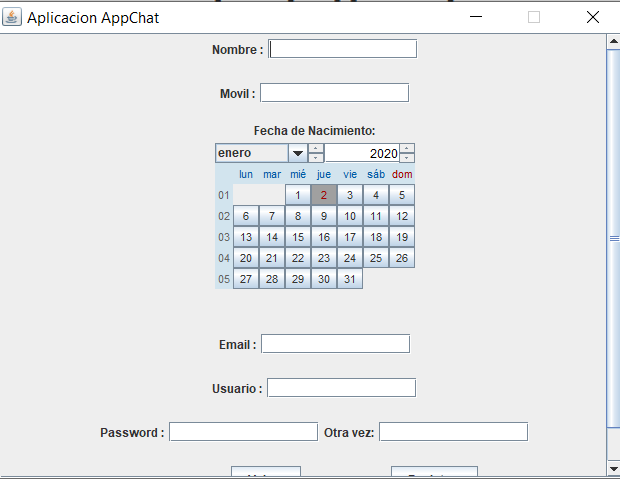
**Manual de Usuario**

Al **Iniciar el Programa** se mostrará una ventana de Inicio de Sesión.

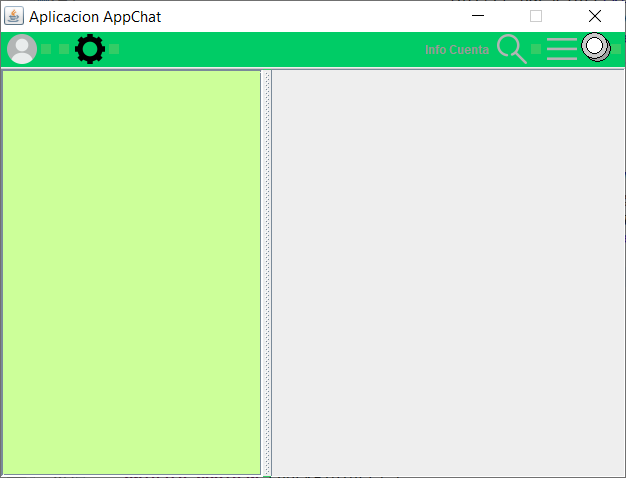


Se puede, o registrar como usuario nuevo o recuperar un usuario previamente registrado.

La ventana **Registrar** tiene la siguiente forma



Una vez rellenado los datos, si se pulsa el Botón *Cancelar* se vuelve a la Ventana de login de Usuario, si se pulsa el Botón de *Aceptar* accedemos a la vista Principal de la aplicación.

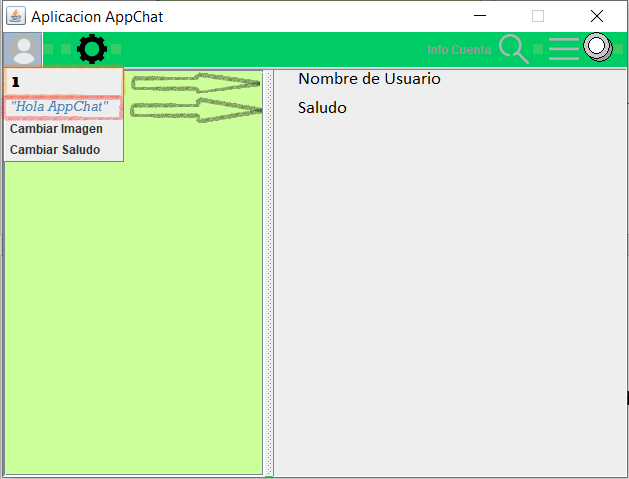


La **Vista Principal** está Compuesto por un barra superior de 6 botones en la parte superior de la ventana y 2 pantallas en el centro.

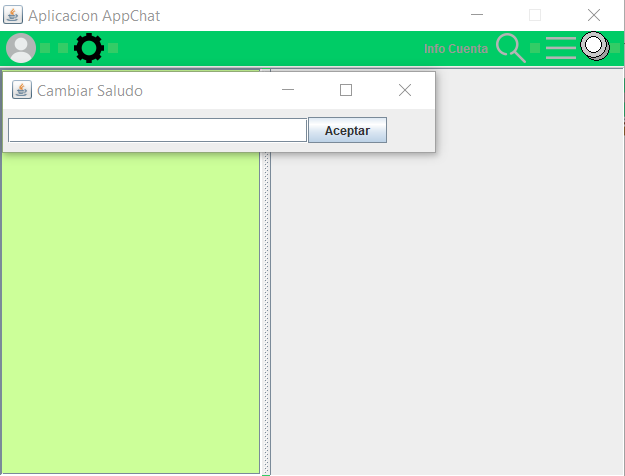
Empezando con la barra superior:

Al seleccionar la **imagen:**

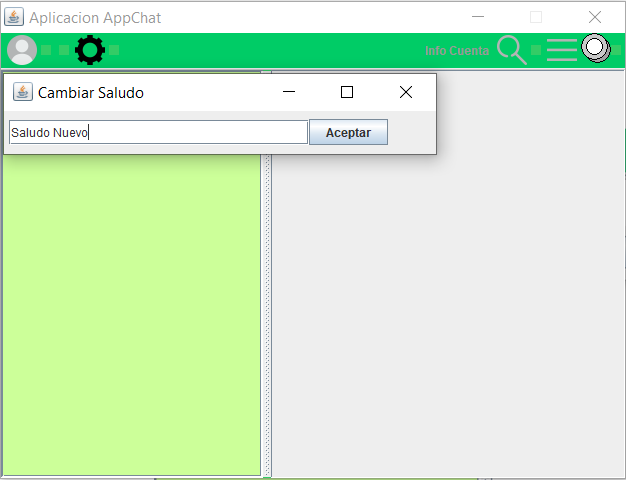
Encontramos más opciones:



Las dos primera columnas son tanto el nombre del Usuario como el Saludo del Usuario. Para modificar el saludo, pulsaremos el botón de *Cambiar Saludo*.



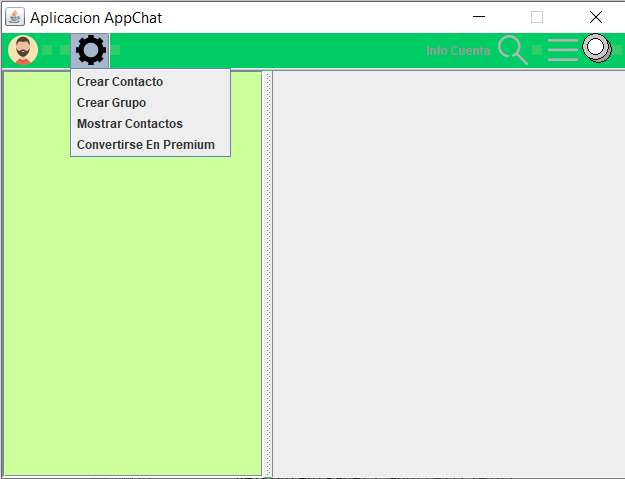
Aparecerá una ventana emergente en forma de barra para *Cambiar Saludo*. Ahí escribiremos el nuevo saludo.



Al pulsar el botón de *Aceptar* nuestro nuevo saludo estará actualizado.

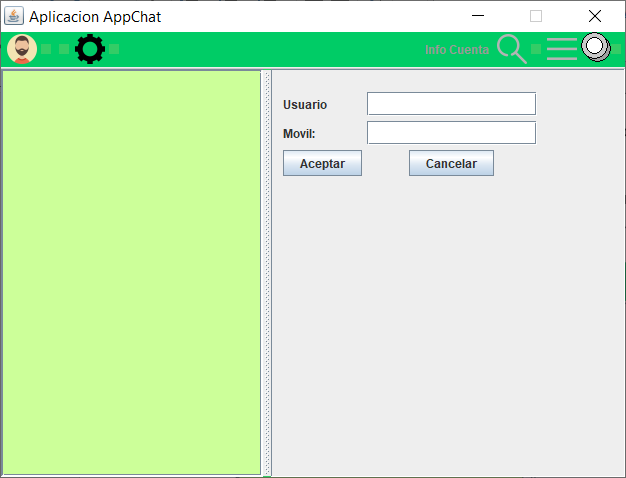
Al pulsar *Cambiar Imagen* aparecerá un seleccionador de archivos para obtener la imagen que deseemos seleccionar.

Al Pulsar el Botón con forma de engranaje encontraremos las siguientes opciones:

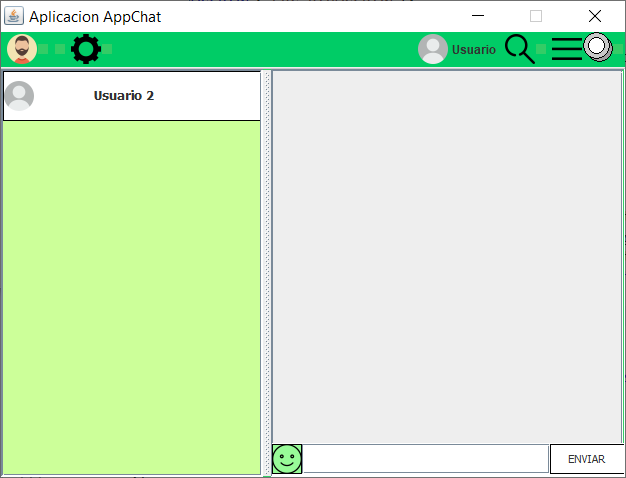


* *Crear Contacto:*

Al pulsar a este botón, en la parte derecha de la ventana aparecerán dos apartados a registrar, el nombre que le queremos dar al contacto y su número de teléfono.



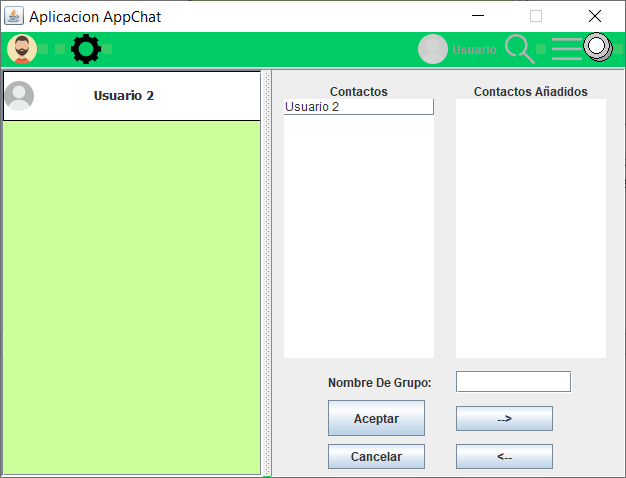
Una vez pulsado el botón *Aceptar* nos aparecerá en la ventana de la izquierda el nuevo Usuario y tendremos en el lado derecho el chat del Contacto.



Como vemos todas las opciones de la barra de la derecha ya no está en color gris.

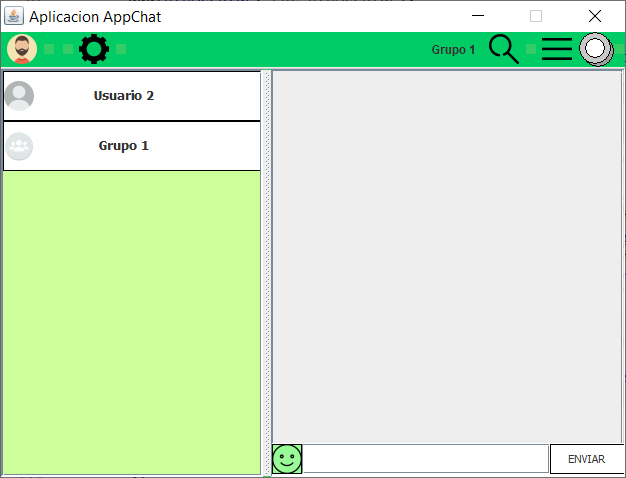
* *Crear Grupo:*

Al pulsar ese botón encontramos esta vista

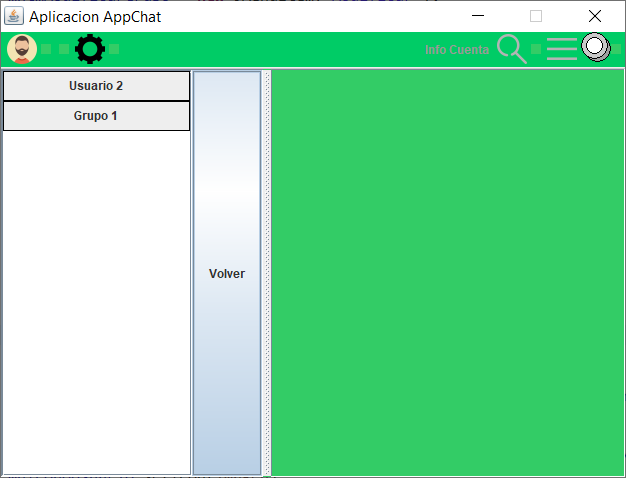


En la tabla de *Contactos* encontramos los contactos de nuestro usuario, en la tabla *Contactos Añadidos* tendremos los Contactos que queremos añadir al Grupo. Para añadir un Usuario, seleccionamos al Contacto que queramos de la tabla de *Contactos* y pulsamos el botón *->.*  Si lo queremos eliminar de los contactos de Grupo, seleccionamos el contacto seleccionado de *Contactos Añadidos* y pulsamos *<-.*

Una vez Escrito el *Nombre del Grupo*, pulsamos el botón *Aceptar.* El Grupo nos aparecerá en el lado izquierdo de la ventana y se nos abrirá el chat del Grupo en el derecho.



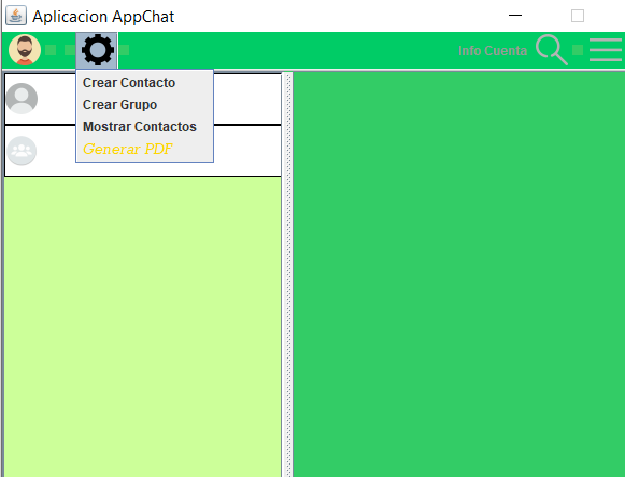
* *Mostrar Contactos:* Nos aparecerá en el lado izquierdo de la ventana La lista de Contactos. Esta lista es sólo informativa, no se podrá seleccionar ningún contacto para abrir ningún Chat.



Siempre podemos volver a la lista de Contactos recientes pulsado el botón *volver.*

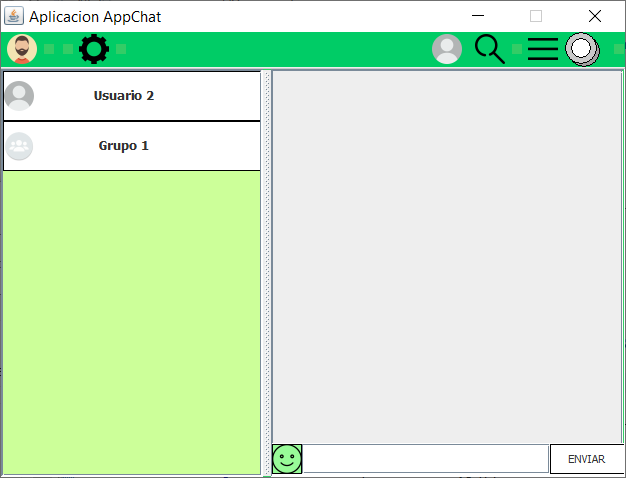
* *Convertirse en Premium:*

Una vez pulsado este botón aparecerá una nueva opción en amarillo:

**

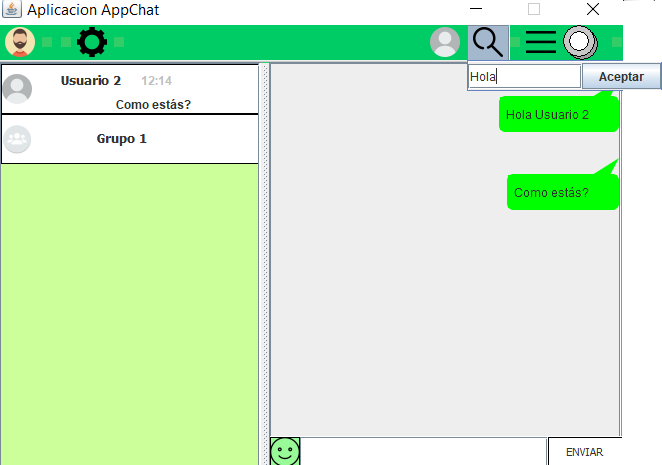
*Generar PDF*, esta opción genera un PDF con los Contacto del Usuario

Como hemos visto previamente, una vez seleccionado una Contacto al registrarlo o directamente seleccionado desde la lista de Contactos que tenemos en el lado izquierdo de la ventana, aparte de encontrar el Chat del Contacto, vemos que los demás botones de la barra se pueden seleccionar. Vamos a ver el ejemplo al seleccionar el contacto Usuario 2:

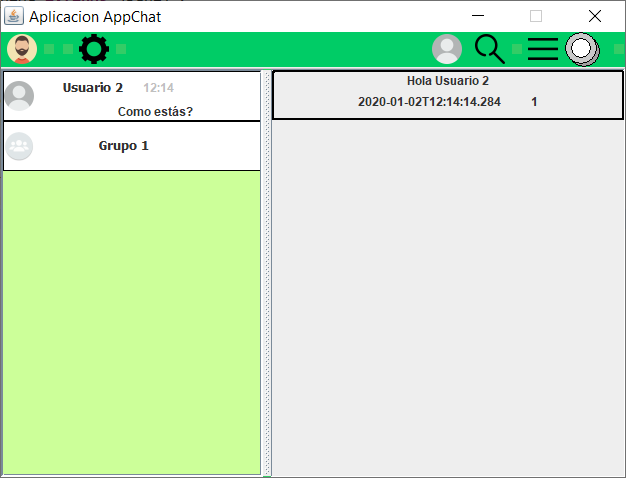


Al pulsar la imagen de la derecha, que es la imagen del Usuario, encontraremos el nombre del Usuario y el nombre. Si es un grupo, encontraremos solo el nombre del grupo.

La *lupa* sirve para la búsqueda de mensajes con palabras clave:

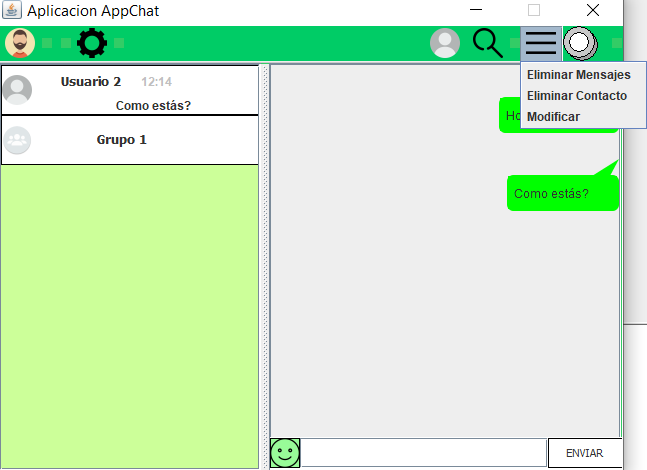


Al pulsar aceptar, nos devolverá una lista donde se encuentre las palabras clave:



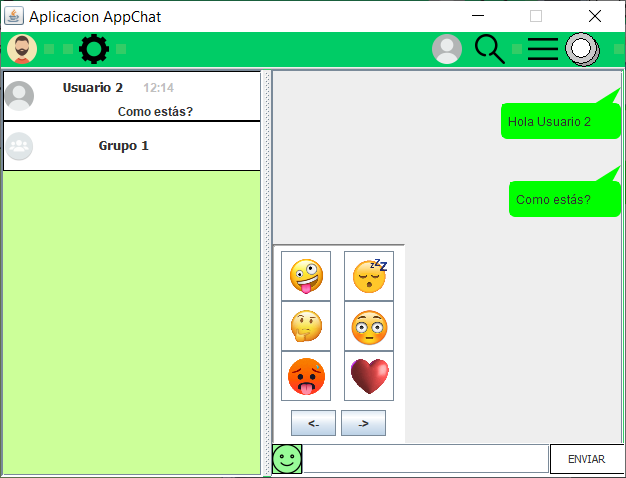
Encontramos el Texto, el Contacto donde se encontró la fecha y quién lo escribió.

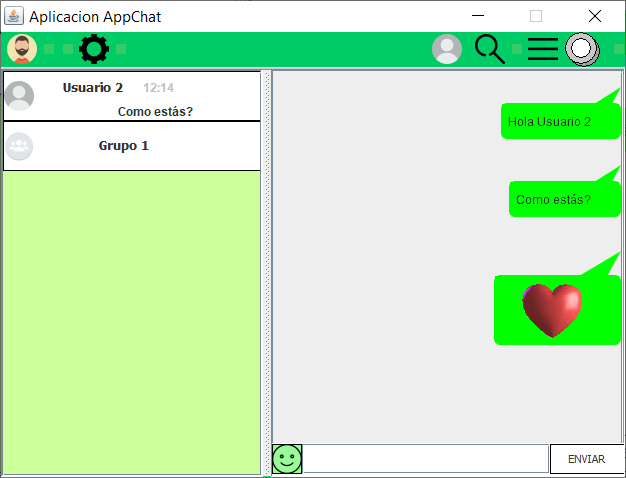
Al pulsar el botón de las 3 rayas horizontales encontramos las opciones del Contacto del Chat que se encuentra abierto:



Podemos eliminar los Mensajes del Chat, eliminar el mismo o Modificarlo. El botón *Modificar* encontramos las mismas opciones que a las de Crea un Contacto o Grupo, por lo que tiene la misma funcionalidad. Cuando se eliminan los mensajes o el Contacto, en el caso de un Grupo solo se puede hacer por el administrador y los cambios se ven en todos los Usuarios. En el Caso de un chat privado entre dos Usuarios, sólo se muestra para la persona que lo ha borrado.

En Cuanto al Chat se pueden enviar Mensajes escritos al pulsar ‘ENTER’ o el botón *Enviar*, o emoticonos al pulsar el botón de emoticonos:





Observaciones finales

La principal dificultad de este proyecto, ha sido:

* La asincronía entre diferentes programas ejecutándose a la vez, esto se ha resuelto con la clase ActualizarBBDD que actualiza los paneles de cada uno de los Chats, cada uno de los paneles que muestra la lista de contactos, la lista de contactos en sí y la información del Contacto seleccionado.

No obstante, este proyecto aborda conceptos muy extensos y completo, como la organización del mismo, su implementación o el uso de Componentes y librerías externas

Desde mi punto de vista me hubiera gustado haber tenido mucho más tiempo para hace un programa más pulido y extenso en cuanto al uso de Componentes y funcionalidad.

Estimación del tiempo dedicado: 80 horas.