

## Tecnologías de Desarrollo de Software

(3º Curso del Grado en Informática)

Curso 2024/2025

18 de septiembre de 2024

---

### Caso Práctico: *AppChat*

Existen algunos servicios o aplicaciones de mensajería para *smartphones* que están muy extendidas como son *Whatsapp* y *Telegram*. Esta especificación de requisitos está inspirada en este tipo de aplicaciones, en particular en la aplicación web de Whatsapp (<https://web.whatsapp.com/>). Sólo se considerarán algunas de las funcionalidades básicas y el objetivo es desarrollar una aplicación de escritorio con Java/Swing. Primero se detallará cada una de las funcionalidades que deberá ofrecer la aplicación. Luego se mostrará la ventana principal de la aplicación y se comentarán el resto de las ventanas que compondrán la interfaz gráfica de usuario. Después se indicará la arquitectura software de la aplicación. Finalmente se proporcionarán detalles sobre la gestión del proyecto, documentación a entregar y la evaluación.

#### Requisitos

---

##### *Login y Registro de usuarios*

Para utilizar los servicios del sistema, los usuarios deben estar registrados y realizar un *login* con su número de teléfono y contraseña. Para registrarse en el sistema, un usuario debe indicar su nombre completo, fecha de nacimiento, email, imagen de perfil, número de teléfono móvil y contraseña, y un mensaje de saludo opcional.

*Nota:* Para introducir la fecha se utilizará el componente Swing *JCalendar* (<https://toedter.com/jcalendar/>). Se debe añadir al proyecto como una dependencia Maven.

*Nota:* Para guardar las imágenes simplemente se almacenará la URL de la imagen. Por tanto, por simplicidad se usarán imágenes disponibles en internet (ver *Anexo*).

##### *Usuarios y Lista de Contactos*

Un usuario tiene una lista de contactos que se identifican con un nombre de contacto. Los contactos pueden ser o bien otros usuarios de la aplicación o bien un grupo de contactos (compuestos a su vez por otros usuarios de la aplicación). *Nota:* un grupo de contacto no puede contener otros grupos de contactos, solo puede contener contactos de usuarios.

Un usuario puede enviar/recibir mensajes a/de otros usuarios. Cuando un usuario recibe un mensaje de otro usuario y este pertenece a su lista de contactos entonces le aparecerá el nombre del contacto. Si no pertenece a la lista de contactos solo le aparecerá el número de teléfono y el sistema le permitirá añadir el usuario como su contacto indicando un nombre de contacto. Lo mismo sucede cuando un usuario envía un mensaje a otro usuario que no es uno de sus contactos. Debe enviarlo usando el número de teléfono del usuario receptor.

El sistema debe permitir también añadir contactos a una lista de contactos del usuario actual indicando un nombre para el contacto y un teléfono. Si el teléfono no existe el sistema debe notificarlo.

El usuario actual debe poder en cualquier momento añadir o cambiar su imagen.

Una vez registrado, el usuario podrá convertir su cuenta en una cuenta “Premium” pagando una cantidad anual. Existen descuentos dirigidos a usuarios registrados en un intervalo de fechas o a usuarios que han enviado más de un cierto número de mensajes en el último mes (el sistema debe estar preparado para incorporar en el futuro nuevos descuentos). Los usuarios Premium tienen disponible la funcionalidad de exportar en un documento PDF los mensajes intercambiados con otro usuario indicado.

### ***Mensajes***

Un mensaje está formado por un texto y puede contener emoticonos como se explica en el documento titulado “Uso de una librería de chat en Swing”, también disponible en los recursos de la asignatura en el AV. Además, un mensaje incluye el número de teléfono del emisor y el del receptor, quien podrá estar o no en su lista de contactos del receptor con un nombre asociado. Si no lo está, el usuario receptor puede añadirlo asociándole un nombre.

Un mensaje también se registrará con la fecha y hora de envío (información necesaria para ordenar los mensajes intercambiados).

Un usuario puede enviar un mensaje a un grupo de contactos. Esto implica que se genere un envío del mensaje (individual) a cada uno de los contactos del grupo. No existe el concepto de conversación grupal. El grupo simplemente facilita el envío de un mensaje a varios contactos a la vez.

### ***Grupos***

Los usuarios pueden crear grupos de contactos para facilitar el envío de mensajes. Un grupo se crea añadiendo los contactos que lo forman y asociándole un nombre al grupo y opcionalmente una imagen. Nótese que el grupo es solo propio de un usuario. No existen grupos compartidos entre usuarios, por simplicidad.

### ***Buscar Mensajes***

Un usuario podrá buscar una lista de mensajes de acuerdo con diferentes filtros.

Se pueden buscar mensajes por aparición de un fragmento de texto, por número de teléfono (sea el emisor o el receptor), por nombre de contacto (de nuevo, sea el emisor o receptor), y por una combinación de estas.

### ***Generar un documento PDF***

Se utilizará un API de creación de archivos PDF (por ejemplo, iText) para generar un archivo que incluya los nombres de los usuarios y grupos y sus teléfonos. Para un grupo se detallará el listado de nombres y teléfonos de sus miembros.

<http://soloinformaticavalgomas.blogspot.com.es/2010/12/generar-un-documento-pdf-desde-java.html>

El acceso a la librería se debe obtener mediante una dependencia a través de Maven.

**IMPORTANTE:** Cada grupo tiene libertad para considerar funcionalidad adicional o realizar cambios sobre la funcionalidad aquí explicada, pero deberá consultarlo con el profesor de la asignatura.

### ***Interfaces de usuario***

---

A continuación se muestran bocetos (wireframes) de las ventanas de la aplicación, aunque el alumno tiene libertad para optar por el diseño que considere más apropiado para cada ventana.

**Ventana Login.** Permite entrar en el sistema con un teléfono y la clave asociada. Es necesario que el usuario esté previamente registrado.

# AppChat

Login

telefono

contraseña

Registrar

Cancelar

Aceptar

**Ventana 'Registro'.** La contraseña se repite dos veces y debe coincidir. Se controla que se hayan introducido valores en los campos obligatorios (todos menos la fecha de nacimiento y la imagen), y se comprueba que las dos claves coincidan. Se mostrarán los mensajes de error pertinentes.

nombre:

apellidos:

telefono

contraseña:

contraseña:

fecha:

//

saludo:

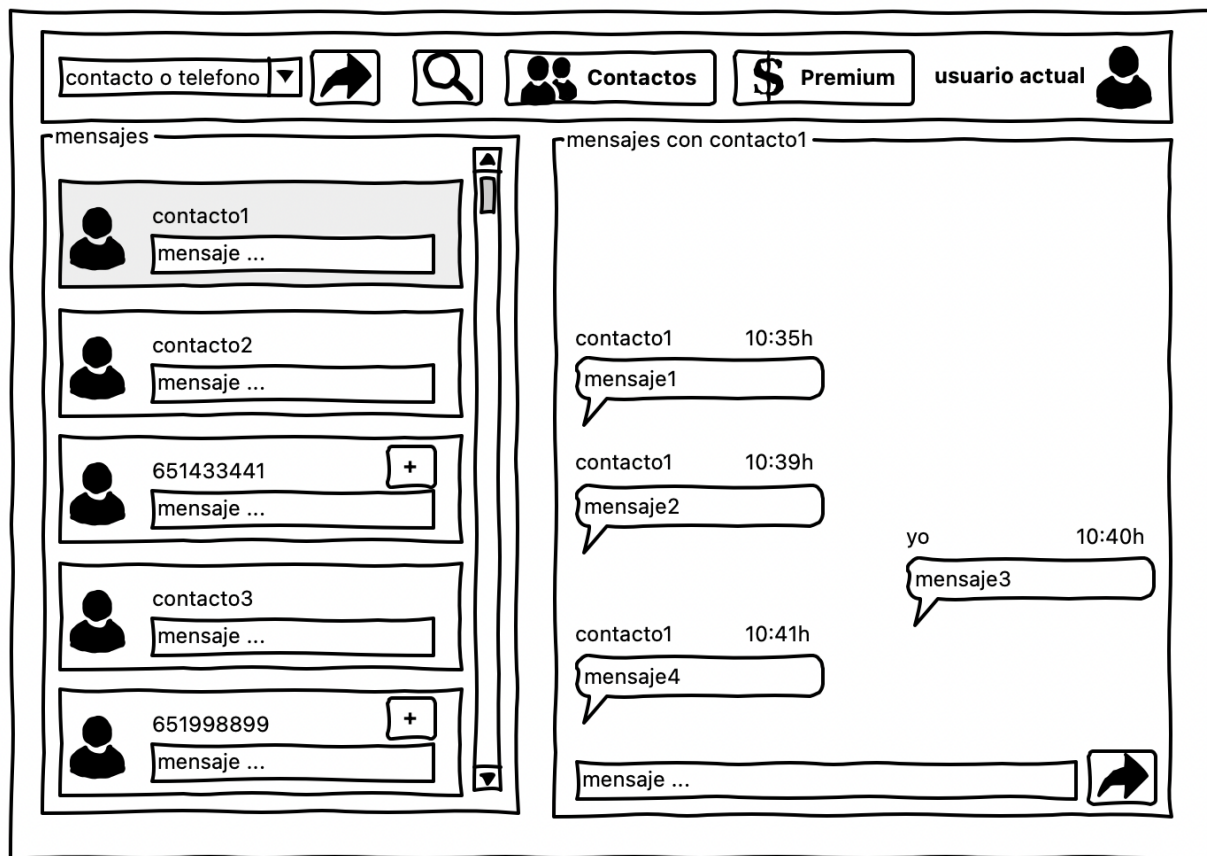
Hi there..

imagen:

Cancelar

Aceptar

**Ventana Principal.** Como se observa abajo está formada por dos paneles a la izquierda y derecha, y un panel arriba. El panel de la izquierda muestra la lista de los últimos mensajes recibidos/enviado por contacto. Se muestra el nombre del contacto o el teléfono si el mensaje es con un usuario que no está en sus contactos. El panel de la derecha muestra los mensajes intercambiados con el contacto seleccionado en el panel de la izquierda. Además, permite enviar nuevos mensajes. En el panel superior, de izquierda a derecha: (1) se puede seleccionar un contacto o introducir un teléfono al que enviar un mensaje para iniciar una conversación (al pulsar el botón con el icono de enviar se carga en el panel de la derecha el espacio para intercambiar mensajes); (2) se puede abrir la ventana de búsqueda (3) se puede abrir la ventana de gestión de contactos; (4) se puede convertir al usuario en usuario Premium; (5) se muestra el nombre del usuario actual y haciendo clic en la imagen se puede cambiar dicha imagen.



El **botón** + junto a los números de teléfono permite añadir al usuario con dicho teléfono como contacto (lista de contactos) del usuario actual dándole un nombre (el campo del teléfono debe aparecer autocompletado). El botón superior de **Contactos** permite la gestión de la lista de contactos.

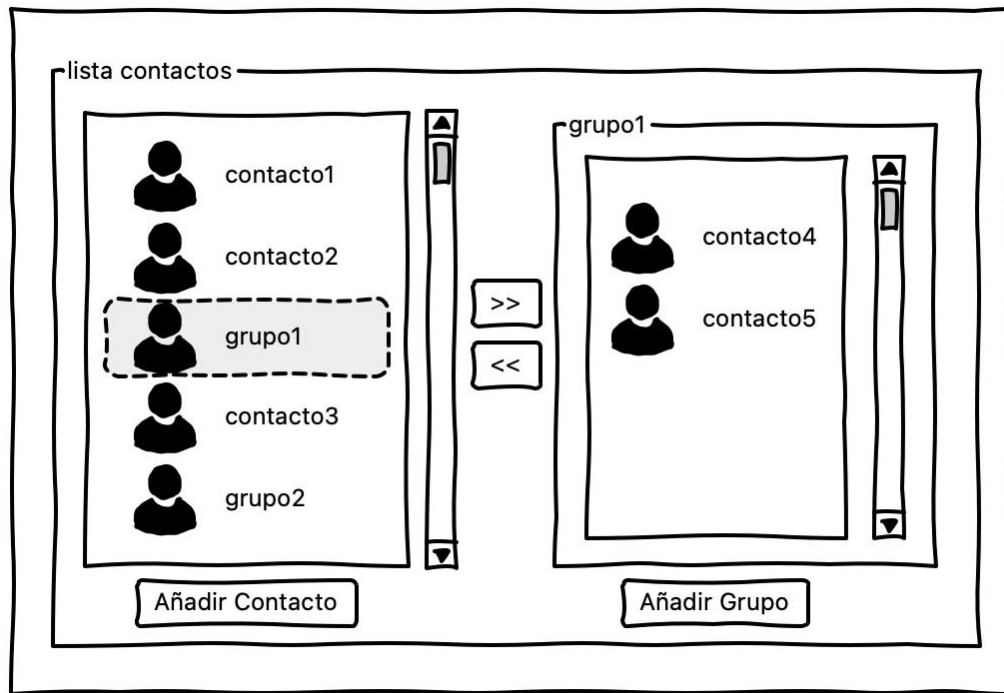
Alert

Introduzca el nombre del contacto y su teléfono


Aceptar


Cancelar

**Ventana Contactos.** Permite gestionar la lista de contactos, añadiendo contactos y creando y gestionando grupos de contactos.

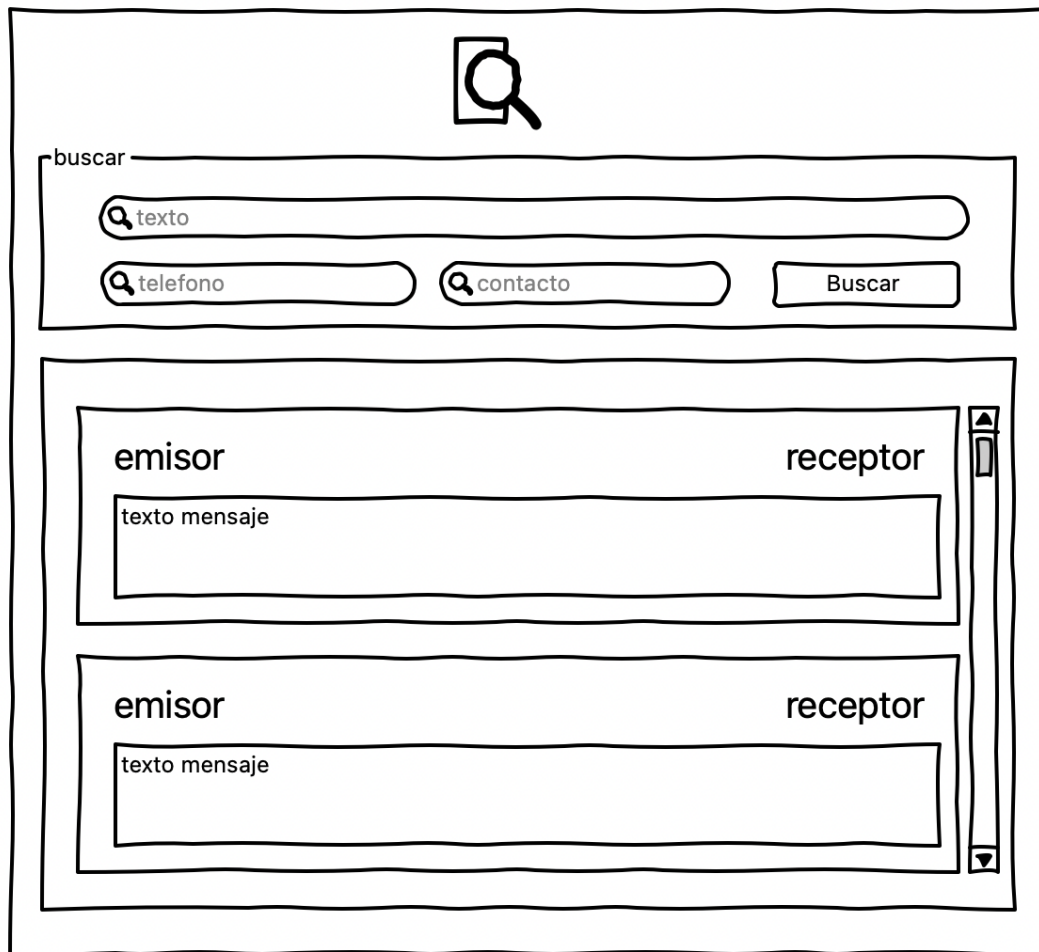


Al pulsar Añadir Contacto se preguntará por un nuevo contacto (si el teléfono no existe se avisará del error):

Alert
<div> Introduzca el nombre del contacto y su teléfono</div> <div><input type="text" value="nombre"/></div> <div><input type="text" value="telefono"/></div> <div><input type="button" value="Aceptar"/> <input type="button" value="Cancelar"/></div>

Alert
<div> El teléfono indicado no existe.</div> <div><input type="button" value="Aceptar"/></div>

[Ventana Buscar.](#)



## Arquitectura de la aplicación

---

Se organizará la aplicación según un modelo de tres capas (Presentación, Lógica de Negocio y Almacenamiento) que se describirá en un seminario de prácticas (en el primer tema teórico se han presentado los aspectos básicos). Se entregará un proyecto de ejemplo “Tienda” para ilustrar cómo aplicar esta arquitectura, así como un ejemplo típico de ventana de login/registro.

Se utilizará la tecnología Java Swing para la implementación de la interfaz de usuario y un servicio de persistencia proporcionado desde la asignatura para el almacenamiento de los datos de la aplicación. Se entregará documentación sobre el uso de estas tecnologías y se explicarán además en seminarios de prácticas.

Se utilizará el patrón DAO para desacoplar la capa de almacenamiento del resto de la aplicación, de acuerdo con las explicaciones proporcionadas en las clases de teoría y prácticas. Cuando se entregue a los alumnos el diagrama de clases de la aplicación se indicará cuáles son las clases persistentes.

Todas las dependencias de librerías externas deben manejarse con Maven.

## Grupos de prácticas

---

Los alumnos deberán formar **grupos de dos** que deberán ser del mismo grupo de teoría salvo casos excepcionales. Se recomienda organizar el trabajo de forma que los dos miembros de un grupo puedan trabajar en paralelo. Por ejemplo, un alumno puede implementar las clases de la interfaz de usuario mientras otro se encarga de las clases relativas a la persistencia de los datos. Los dos alumnos deben tener un conocimiento de todos los aspectos de la práctica y deberían haber programado en cada capa de la arquitectura.

## Gestión del proyecto

---

Cada grupo deberá crear un proyecto **Maven** que debe ser almacenado en un repositorio privado **GIT** (es posible elegir entre GitHub y GitLab). Las dependencias a las librerías externas usadas se manejarán con Maven. Los alumnos deben invitar a su profesor de prácticas al repositorio creado para que pueda acceder a su código en cualquier momento. Se dedicará un seminario de prácticas a explicar el uso de Maven y otro el de repositorios Git.

## Código

---

1. El código debería ser escrito utilizando **expresiones lambda** y **streams** en aquellos puntos en los que se considere apropiado.
2. El alumno no debe cometer errores relacionados con los patrones **GRASP** y principio **separación modelo-vista**.
3. Los conocimientos sobre patrones de diseño se valoran principalmente en el examen de teoría de la asignatura. No obstante, la realización de esta práctica conlleva el uso de algunos patrones aplicados directamente por el alumno o indirectamente al utilizar alguna construcción o funcionalidad de Java o de los servicios proporcionados por los profesores. El alumno deberá indicar en la documentación cuáles son estos patrones y comentar brevemente su uso.

## Hitos

---

La práctica se organizará en las siguientes fases:

1. Realización del **modelado de requisitos** y **diagrama de clases inicial**. La **fecha de esta primera entrega** será el **20 de octubre**. Se creará una tarea en el Aula Virtual para que cada grupo pueda enviar su diagrama de clases junto a una explicación de los detalles que considere oportuno (un único envío por grupo). El archivo pdf que entreguen debe también incluir el nombre de cada alumno, e indicar la url del repositorio.
2. En la semana del 21 de octubre se discutirá el diagrama de clases entregado por los profesores y que los alumnos deberían utilizar en la implementación del caso práctico.
3. Cada grupo debería aplicar una estrategia de desarrollo iterativo completando de forma incremental la funcionalidad de la especificación y escribiendo el código necesario para cada capa: a) diseño e implementación de la interfaz de usuario a partir de los conocimientos adquiridos en los seminarios de Swing (el diseño de ventanas expuesto arriba pretende facilitar el trabajo de los alumnos); b) diseño e implementación del controlador; c) implementación de la lógica del dominio; d) implementación de la persistencia.

**Fecha de entrega final: 16 de enero de 2025**

Se creará una **tarea** en el Aula Virtual para la entrega de la documentación comentada abajo (archivo pdf) y la aplicación como con un fichero zip con el proyecto Eclipse desarrollado.

## Documentación

---

Constará de las siguientes partes:

1. Índice de contenidos
2. Historias de usuario y diagrama de clases del dominio.
3. Un diagrama de colaboración o secuencia para la operación añadir un nuevo contacto a un grupo creado por el usuario.
4. Una breve explicación de la arquitectura de la aplicación y decisiones de diseño que se consideren de interés para la comprensión del trabajo.

5. Explicación de los patrones de diseño utilizados.
6. Un pequeño manual de usuario que explique cómo usar la aplicación
7. Observaciones finales (deben incluir una estimación del tiempo dedicado)

## **Evaluación**

---

Cada parte se valorará con el siguiente porcentaje:

- Modelado de requisitos y del dominio de clases (10%)
- Diseño de la interfaz (30%)
- Ejercicios de patrones (5%),
- Código (separación en tres capas, legibilidad, uso de principios básicos de programación OO, implementación de patrones de diseño, uso de swing, comentarios) (40%),
- Documentación entregada (15%).
- **Es obligatorio el uso de Git y Maven.**



## ANEXO: Mostrar imágenes de internet en Swing

---

```
String path =  
    "https://widget-assets.geckochat.io/69d33e2bd0ca2799b2c6a3a3870537a9.png";  
URL url = new URL(path);  
BufferedImage image = ImageIO.read(url);  
JLabel label = new JLabel(new ImageIcon(image));
```