

Aplicación Android: *PicPals*

Hecho por: Galder García



Segunda Entrega (23/04/2024)

Índice de contenidos

Índice de contenidos.....	2
Introducción.....	4
Código.....	4
Descripción de la aplicación.....	4
Descripción técnica de la aplicación.....	5
Diseño y estructura.....	5
Seguridad.....	5
Base de datos.....	6
Tabla usuarios.....	6
Tabla imagenes.....	7
Tabla image_sharing.....	7
Servicios PHP.....	8
index.php.....	8
imagenes.php.....	8
test.php.....	8
Clases java.....	9
Actividades.....	9
MainActivity.....	9
Gestión de permisos.....	10
LoginActivity.....	11
RegisterActivity.....	12
UploadImageActivity.....	13
ViewUploadedPhotosActivity.....	14
ViewSharedPhotosActivity.....	15
Clases para la conexión a la base de datos.....	16
ConexionDBUsuarios.....	16
ConexionDBSubirImagenes.....	16
ConexionDBBajarImagenes.....	16
Otras clases java.....	17
UploadedPhotosAdapter.....	17
ServicioFirebase.....	17
Requisitos cumplidos.....	18
Posibles mejoras.....	19
Manual de usuario.....	20
Menú principal.....	20
Registro y logueo.....	21
Subida de imágenes.....	22
Guía: Subir imagen desde la cámara.....	23
Guía: Subir imagen desde la galería.....	25
Guía: Ver imágenes subidas por el usuario.....	27
Guía: Ver imágenes compartidas con el usuario.....	28
Guía: Recibiendo notificaciones.....	29

Bibliografía y recursos utilizados.....	30
--	-----------

Introducción

Código

Esta documentación es parte de la segunda entrega de la asignatura Desarrollo Avanzado de Software. El código tanto de la aplicación android, como del servicio web desplegado en el servidor, se puede encontrar en el siguiente repositorio público de GitHub:

<https://github.com/galdergcupv/PicPals/tree/master>

Descripción de la aplicación

PicPals es una aplicación en la que los usuarios pueden almacenar imágenes en un servidor remoto de forma privada, de modo que solo ellos pueden acceder a sus imágenes; también permite compartir imágenes con otros usuarios de la aplicación. **PicPals** ofrece la opción de sacar una imagen nueva usando la cámara del dispositivo, o elegir una foto ya existente de la galería para subirla al servidor y/o compartirla con otros usuarios. Además, si otro usuario comparte una foto con nosotros mientras tenemos la sesión iniciada en nuestro dispositivo, recibiremos una notificación a través de FCM.

Descripción técnica de la aplicación

Diseño y estructura

La aplicación **PicPals** está dividida en dos partes: Por un lado tenemos la aplicación android en nuestro dispositivo, y por otro lado está el servidor web que gestiona la base de datos de la aplicación.

En la aplicación android tenemos distintas actividades que permiten:

- Gestionar la sesión de usuarios (registrarse en nuestro sistema, iniciar y cerrar sesión).
- Sacar fotos desde la cámara o elegir las desde la galería.
- Subir las fotos al servidor (dándoles nombre y eligiendo con quién compartirlas).
- Ver las fotos subidas por el usuario.
- Ver las fotos compartidas con el usuario (además de ver quién las ha compartido).

En el servidor tenemos una base de datos MariaDB en la que guardamos la información de los usuarios, las imágenes, y las relaciones entre usuarios e imágenes compartidas. Para poder acceder y gestionar esta base de datos, disponemos de distintos servicios php que se encargan de recibir las peticiones HTTP enviadas desde la aplicación android.

Seguridad

Para hacer que nuestro sistema sea más seguro, hemos utilizado las siguientes técnicas relacionadas con la gestión de la base de datos:

- Guardar las contraseñas de los usuarios utilizando *hash + salt*. Esto hace que sean menos vulnerables en caso de que la base de datos quede expuesta, ya que no se pueden obtener las contraseñas de forma directa, como sí se podría si estuviesen en texto plano. Además, al usar *salt*, conseguimos que se generen hashes diferentes aunque varios usuarios tengan la misma contraseña, lo que protege contra ataques de fuerza bruta o ataques de diccionario.
- Sanear las entradas de las instrucciones SQL escapando caracteres especiales para evitar inyecciones SQL.

Base de datos

La base de datos remota de nuestra aplicación está formada por 3 tablas: usuarios, imagenes, e image_sharing. Para acceder y administrar la base de datos se utiliza phpMyAdmin, que está alojado en el servidor de la asignatura con la siguiente dirección <http://34.29.139.252:8890>. El usuario es “admin” y la contraseña es “test”.

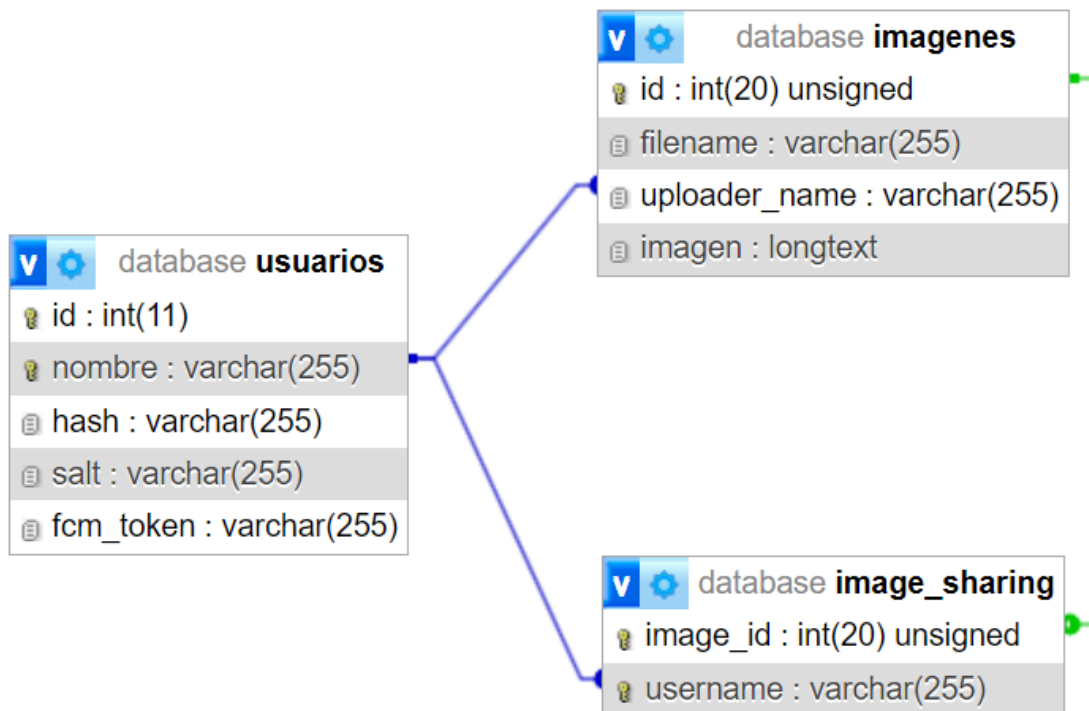


Imagen 1. Esquema de la base de datos

Tabla usuarios

Esta tabla se utiliza para almacenar la información de los usuarios. Está formada por los siguientes campos:

- id, de tipo int(11) que actúa como clave primaria.
- nombre, de tipo varchar(255) único. Almacena el nombre de usuario, al ser único solo permite que haya un único usuario con ese nombre.
- hash, de tipo varchar(255). Almacena el hash de la contraseña del usuario.
- salt, de tipo varchar(255). Almacena la sal utilizada para generar el hash de la contraseña del usuario.
- fcm_token, de tipo varchar(255). Almacena el token fcm asociado al usuario. El valor por defecto es NULL (si el usuario no tiene la sesión iniciada en ningún dispositivo).

Tabla imagenes

Esta tabla se utiliza para almacenar la información de las imágenes. Está formada por los siguientes campos:

- id, de tipo int(20) unsigned. Actúa como clave primaria.
- filename, de tipo varchar(255). Almacena el nombre de la imagen.
- uploader_name, de tipo varchar(255). Clave externa de usuarios(nombre). Relaciona al usuario que ha subido la foto.
- imagen, de tipo longtext. Almacena el string de la foto en formato Base64.

Tabla image_sharing

Esta tabla se utiliza para crear una relación de varios a varios entre usuarios e imágenes. Nos permite asociar a cada imagen varios usuarios con los que compartirla:

- image_id, de tipo int(20) unsigned. Clave externa de imagenes(id). Forma la clave primaria junto con el campo username.
- username, de tipo varchar(255). Clave externa de usuarios(nombre). Forma la clave primaria junto con el campo image_id.

Servicios PHP

Para acceder a la base de datos de nuestro servidor utilizamos distintos servicios PHP que se encargan de recibir y gestionar las peticiones HTTP enviadas por la aplicación android .

index.php

Es el servicio principal, lo usamos para gestionar todo lo relacionado con los usuarios. Cuando recibe una petición POST, comprueba los campos que se incluyen en la petición, y dependiendo del valor del campo *action* puede hacer lo siguiente:

- *action* = register: Registra al usuario en la base de datos, genera un hash y salt con la contraseña y los almacena.
- *action* = login: Obtiene el hash y la salt para ese usuario, y comprueba si son compatibles con la contraseña proporcionada. Si lo son, guarda el token fcm del dispositivo en el usuario.
- *action* = logout: Simplemente elimina el token fcm del usuario para indicar que ya no se encuentra logueado en ese dispositivo.

imagenes.php

Este servicio se encarga de gestionar todo lo relacionado con las imágenes. Al igual que index.php, recibe una petición POST, comprueba los campos y ejecuta diferentes acciones dependiendo del valor del campo *action*:

- *action* = save_image: Guarda la imagen en la base de datos y comprueba si el campo *shareWith* está vacío. Si no lo está, separa los usuarios por comas, y por cada usuario comprueba si existe en la base de datos, lo añade a la tabla image_sharing junto con la imagen asociada, e intenta enviar la notificación usando fcm (puede que el usuario no esté logueado en ningún dispositivo, en ese caso no podrá recibir la notificación).
- *action* = get_user_images: Simplemente obtiene una lista de las imágenes subidas por el usuario especificado, en formato json.
- *action* = get_shared_images: Similar al caso de get_user_images, obtiene las imágenes que han sido compartidas con el usuario especificado.

test.php

Este servicio se utiliza únicamente para probar las notificaciones FCM. Se accede desde el navegador (<http://34.29.139.252:81/test.php>), y carga una página HTML muy sencilla que nos permite rellenar las cabeceras para simular una petición HTTP similar a la que se enviaría desde la aplicación android cuando compartimos una foto con algún usuario (lo que hace que internamente se ejecute una petición para mandar la notificación FCM). Para facilitar su uso se han establecido unos parámetros por defecto extraídos de una petición HTTP real generada por la aplicación android.

Clases java

Actividades

MainActivity

Es la actividad principal que se ejecuta al iniciar la aplicación. También se encarga de solicitar los permisos cuando se abre. Está compuesta por los siguientes elementos:

- Una barra superior que gestiona la sesión del usuario:
 - Si no hay ningún usuario logueado, mostrará los botones para iniciar sesión o registrarse. Si hay un usuario logueado, mostrará el nombre del usuario y un botón para cerrar sesión.
 - **Botón “Iniciar sesión”**: Lanza la actividad *LoginActivity* que permite al usuario iniciar sesión. Esto hace que se guarde el usuario en las preferencias y se actualice la base de datos para relacionar el token fcm del dispositivo con el usuario.
 - **Botón “Registrarse”**: Lanza la actividad *RegisterActivity* que permite al usuario registrarse.
 - **Botón “Cerrar sesión”**: Primero muestra un diálogo de confirmación, si el usuario pulsa “Sí”, entonces elimina el usuario de las preferencias y elimina el token fcm del usuario en la base de datos.
- Una imagen con el logo de la aplicación. (Puramente estética)
- **Un botón “Sacar Foto”**: Si el usuario está logueado y tenemos el permiso de usar la cámara, abre la aplicación de cámara para sacar una foto. Cuando se recibe la foto: se reescala, se convierte a base64, y se lanza la actividad *UploadImageActivity* para continuar con el proceso de subir la foto al servidor.
- **Un botón “Elegir Foto Existente”**: Si el usuario está logueado y tenemos el permiso de acceder a la galería, abre la aplicación de galería para que el usuario elija una foto. Cuando se recibe la foto: se reescala, se convierte a base64, y se lanza la actividad *UploadImageActivity* para continuar con el proceso de subir la foto al servidor.
- **Un botón “Ver Fotos Subidas”**: Si el usuario está logueado, lanza la actividad *ViewUploadedPhotosActivity* para mostrar las fotos subidas por el usuario. Si el usuario no tiene ninguna foto subida, muestra un toast indicándolo.
- **Un botón “Ver Fotos Compartidas”**: Si el usuario está logueado, lanza la actividad *ViewSharedPhotosActivity* para mostrar las fotos compartidas con el usuario. Si el usuario no tiene ninguna foto compartida, muestra un toast indicándolo.

Nota importante: Para usar cualquiera de los cuatro últimos botones (los que no están relacionados con la sesión), es necesario que el usuario esté logueado. Por ello, si se intenta usar alguna de estas funciones sin estarlo, se abre un diálogo indicando al usuario que es necesario iniciar sesión. Además, desde este diálogo se ofrece la posibilidad de abrir la actividad para iniciar sesión si el usuario pulsa el botón “Iniciar Sesión” dentro del diálogo.

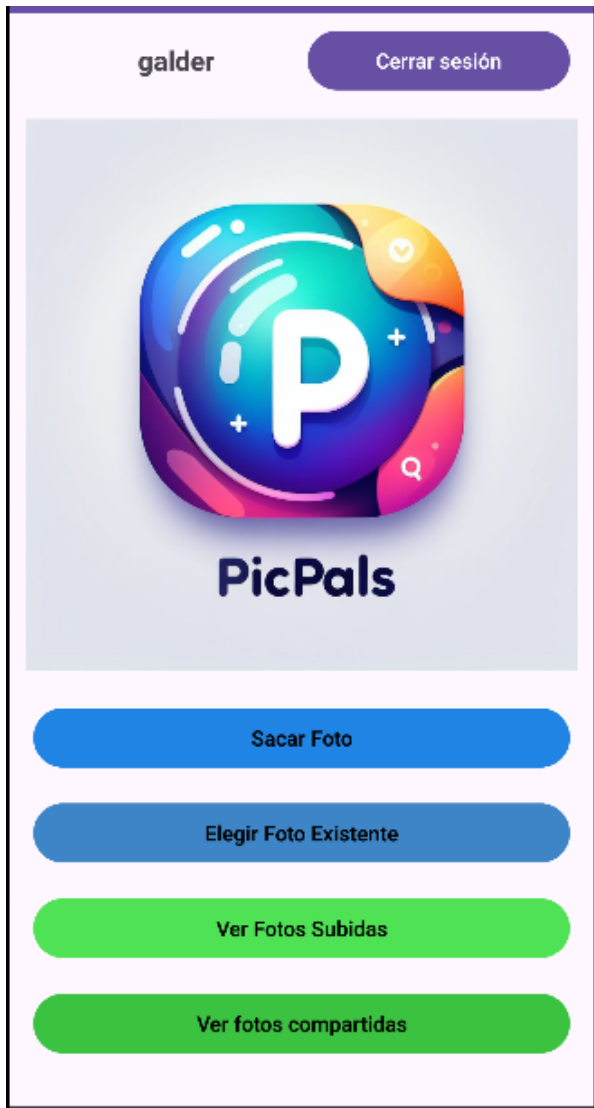


Imagen 2. MainActivity (Sesión Iniciada)

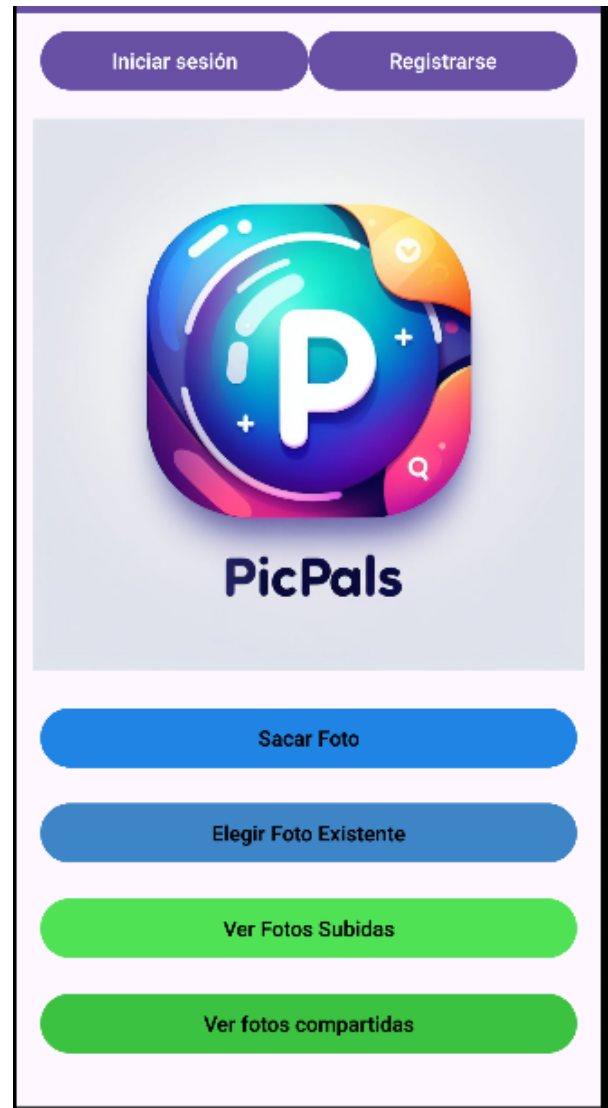


Imagen 3. MainActivity

Gestión de permisos

Al iniciar la aplicación, desde MainActivity se comprueba si se tienen los permisos para usar la cámara, la galería, y enviar notificaciones (solo en versiones android 13 o superiores).

Si no se tiene alguno de estos permisos, entonces se solicitan al usuario.

Además, por si el usuario decide rechazar alguno de estos permisos al iniciar la aplicación, cuando se hace click en los botones “Sacar Foto” o “Elegir Foto Existente”, se vuelve a comprobar si se tienen los permisos correspondientes y se vuelven a pedir si es necesario.

LoginActivity

Esta actividad permite al usuario iniciar sesión. Está compuesta por dos editText para que el usuario introduzca sus credenciales, y un botón para enviar el inicio de sesión. Para iniciar sesión, se envía una petición HTTP al servidor usando la clase *ConexionDBUsuarios*. Dentro de la petición, además de los datos del usuario, se envía también el token fcm del dispositivo.

Cuando se recibe una respuesta del servidor, ésta se muestra en un toast indicando si ha ocurrido algún problema ("Contraseña incorrecta", "Usuario no encontrado", etc..) o si el inicio de sesión ha sido exitoso. En caso de ser exitoso, se actualizan las preferencias para guardar el usuario, se cierra la actividad, y se vuelve a la actividad principal.

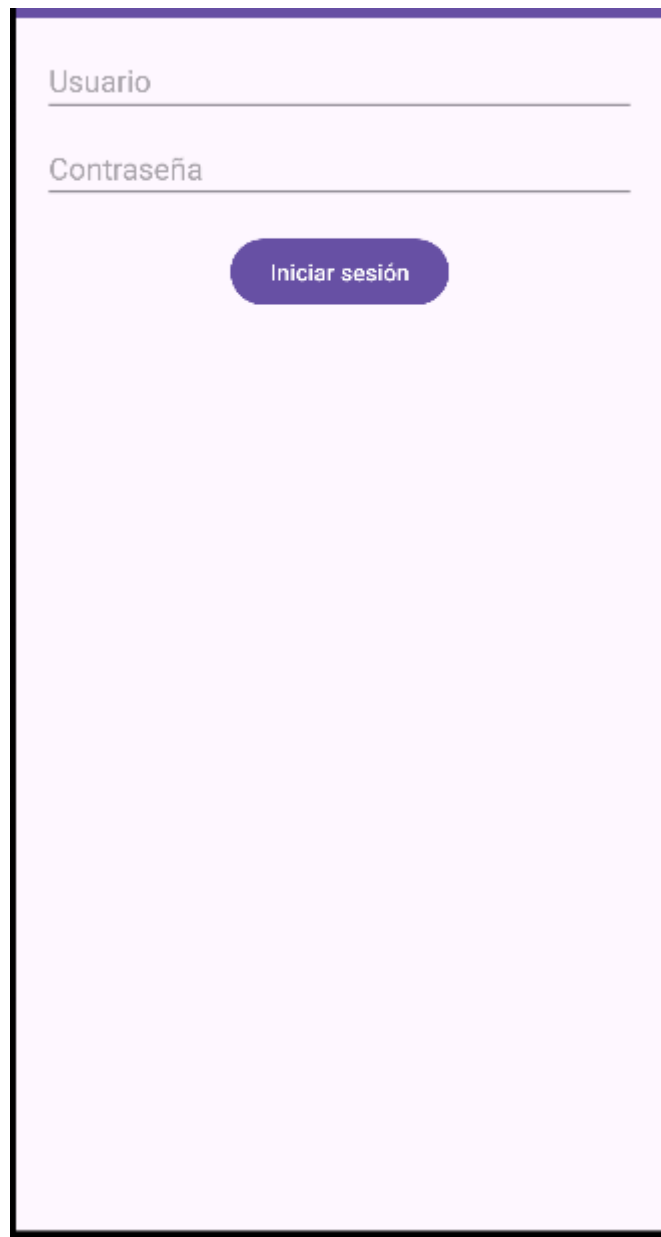
The image shows a mobile application screen for a login activity. It has a light purple background. At the top, there are two input fields. The first one is labeled 'Usuario' in a light gray font. The second one is labeled 'Contraseña' in a light gray font. Below these fields, there is a rounded rectangular button with a dark purple background and the text 'Iniciar sesión' in white. The entire screen is enclosed in a black border.

Imagen 4. LoginActivity

RegisterActivity

Esta actividad permite al usuario registrarse. Está compuesta por dos editText para que el usuario introduzca sus credenciales, y un botón para enviar el registro. Para hacer el registro, se envía una petición HTTP al servidor usando la clase *ConexionDBUsuarios*.

Cuando se recibe una respuesta del servidor, ésta se muestra en un toast indicando si ha ocurrido algún problema ("El usuario ya existe", "Error al registrar usuario", etc..) o si el registro ha sido exitoso. En caso de ser exitoso, se cierra la actividad, y se vuelve a la actividad principal.

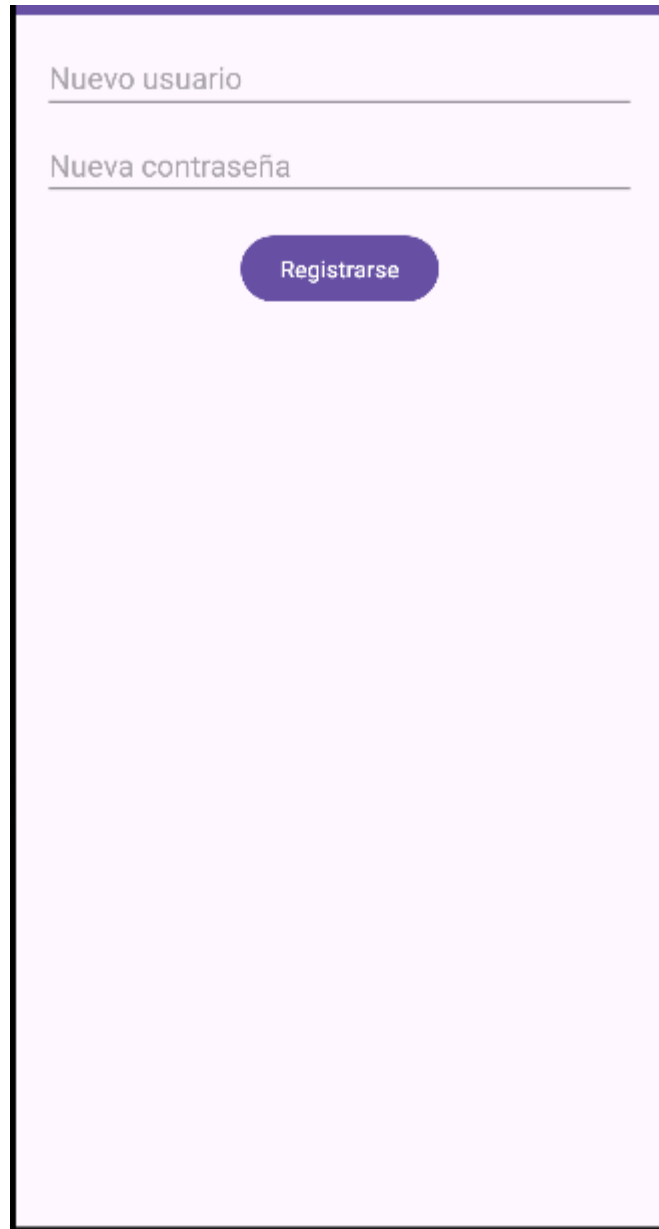
The image shows a mobile application screen for user registration. It has a light purple background. At the top, there is a title "Nuevo usuario" followed by a horizontal line. Below this, there is another horizontal line with the label "Nueva contraseña" to its left. In the center of the screen, there is a rounded rectangular button with a dark purple background and the text "Registrarse" in white.

Imagen 5. RegisterActivity

UploadImageActivity

Esta actividad permite al usuario subir una foto al servidor. Se abre después de que el usuario haya sacado o elegido una foto que quiera subir. Está compuesta por un imageView que muestra la foto, dos editText para que el usuario introduzca el nombre de la foto y, si quiere compartirla con alguien, los nombres de los usuarios a compartir (separados por comas), y un botón para subir la imagen.

Al pulsar subir imagen, se envía una petición HTTP al servidor usando la clase *ConexionDBSubirImagenes*. En esta petición van los datos introducidos y la propia imagen transformada en Base64.

Cuando se recibe una respuesta del servidor, se muestra un toast mostrándola, y si se ha guardado con éxito, se cierra la actividad, y se vuelve a la actividad principal.

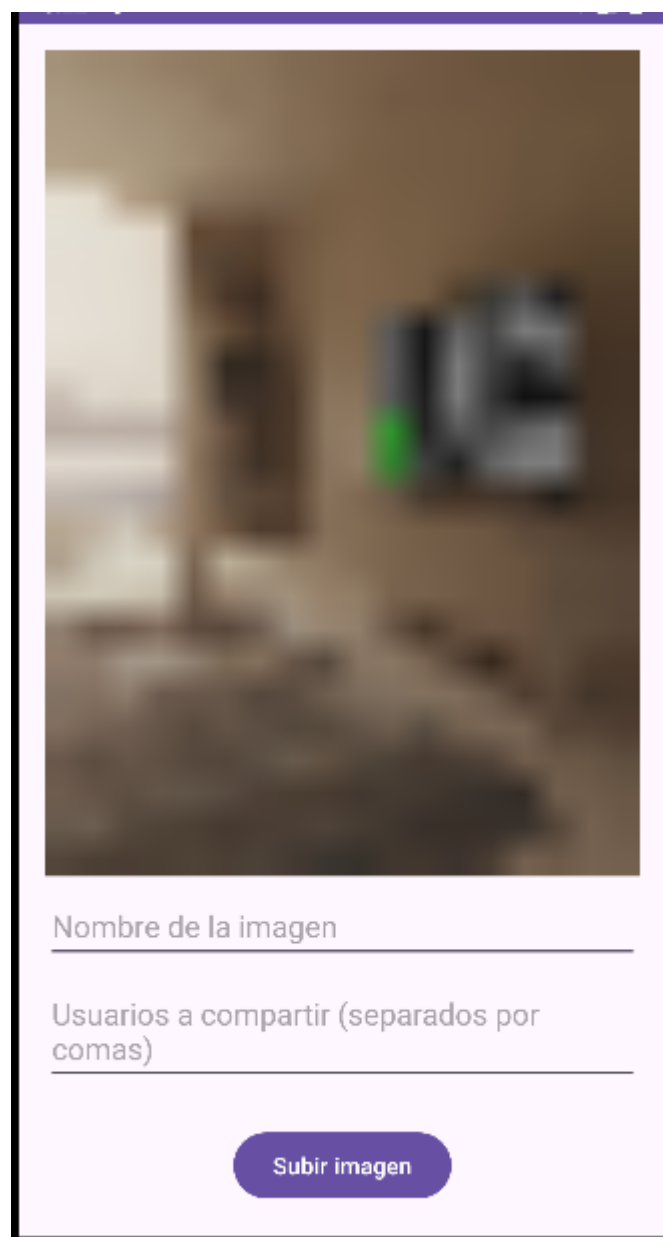


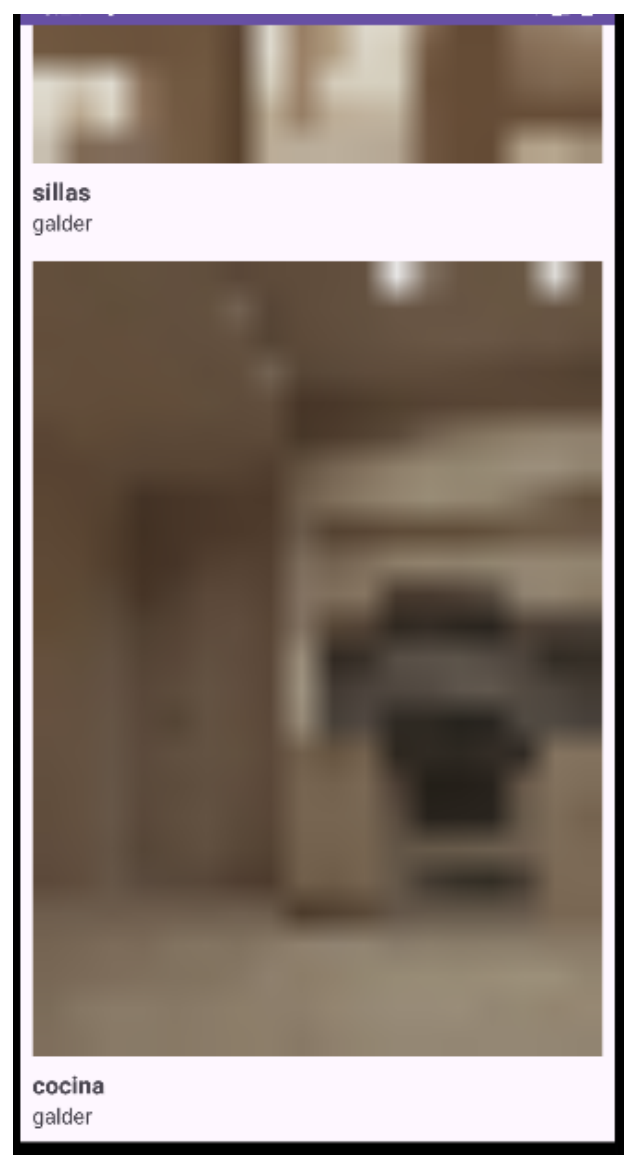
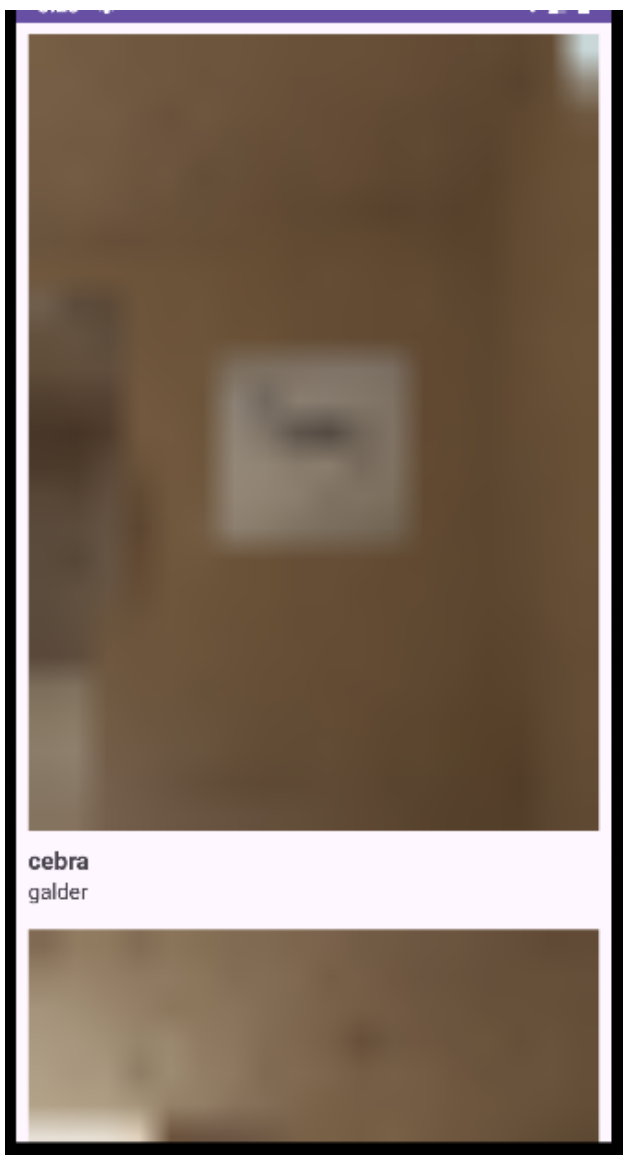
Imagen 6. UploadImageActivity

ViewUploadedPhotosActivity

Esta actividad permite al usuario visualizar todas las fotos que ha subido al servidor. Está compuesta por un `recyclerView` que usa el adapter *UploadedPhotosAdapter* para mostrar las imágenes con su nombre y el usuario que la ha subido (en este caso es el mismo usuario).

Cuando se abre, se envía una petición HTTP al servidor para obtener las imágenes del usuario, a través de la clase `ConexionDBBajarImágenes`.

Una vez obtiene las imágenes en formato json, las procesa y, usando el adapter *UploadedPhotosAdapter*, llena el `recyclerView` con la información. En caso de que el usuario no tuviese ninguna foto subida, muestra un toast indicándolo, cierra la actividad, y vuelve a la actividad principal.



Imágenes 7 y 8. ViewUploadedPhotosActivity

ViewSharedPhotosActivity

Esta actividad permite al usuario visualizar todas las fotos que han sido compartidas con él. Al igual que *ViewUploadedPhotosActivity*, está compuesta por un *recyclerView* que usa el mismo adapter *UploadedPhotosAdapter* para mostrar las imágenes con su nombre y el usuario que la ha subido (también es el usuario que la ha compartido).

Cuando se abre, se envía una petición HTTP al servidor para obtener las imágenes compartidas con el usuario, a través de la clase *ConexionDBBajarImagenes*.

Una vez obtiene las imágenes en formato json, las procesa y, usando el adapter *UploadedPhotosAdapter*, llena el *recyclerView* con la información. En caso de que el usuario no tuviese ninguna foto compartida, muestra un toast indicándolo, cierra la actividad, y vuelve a la actividad principal.



Imagen 9. ViewSharedPhotosActivity

Clases para la conexión a la base de datos

Para conectar con la base de datos del servidor utilizamos peticiones HTTP. Estas peticiones se crean utilizando clases que extienden la clase Worker en java. Como el formato de las peticiones varía ligeramente si son para gestionar usuarios, subir imágenes, o bajar imágenes, se utiliza una clase específica para cada caso. Se podrían combinar en una sola clase, pero separándolas de esta forma hace que el código sea más legible y fácil de entender.

ConexionDBUsuarios

Esta clase extiende la clase Worker y se encarga de enviar las peticiones HTTP al servidor relacionadas con los usuarios. No procesa ningún tipo de información, solo recibe los datos para las cabeceras, crea la petición, y devuelve la respuesta. También se han añadido logs para ver la solicitud enviada y la respuesta.

ConexionDBSubirImagenes

Esta clase extiende la clase Worker y se encarga de enviar las peticiones HTTP al servidor relacionadas con la subida de imágenes. No procesa ningún tipo de información, solo recibe los datos para las cabeceras, crea la petición, y devuelve la respuesta. También se han añadido logs para ver la solicitud enviada y la respuesta.

ConexionDBBajarImagenes

Esta clase extiende la clase Worker y se encarga de enviar las peticiones HTTP al servidor relacionadas con la bajada de imágenes. No procesa ningún tipo de información, solo recibe los datos para las cabeceras, crea la petición, y devuelve la respuesta. También se han añadido logs para ver la solicitud enviada y la respuesta.

Otras clases java

UploadedPhotosAdapter

Es el adaptador que se usa para los recyclerView de *ViewUploadedPhotosActivity* y *ViewSharedPhotosActivity*. Se encarga de recibir las imágenes en Base64 y decodificarlas para mostrarlas en un imageView. Además recibe el nombre de la imagen y el usuario que la ha subido y, con esta información, genera los ítems para el recyclerView.

ServicioFirebase

Esta clase se encarga de gestionar todo lo relacionado con las notificaciones FCM. Cuando se genera un nuevo token FCM para el dispositivo, lo guarda en las preferencias de la aplicación. Cuando se recibe un mensaje FCM, se encarga de crear la notificación con la información del mensaje y enviarla.

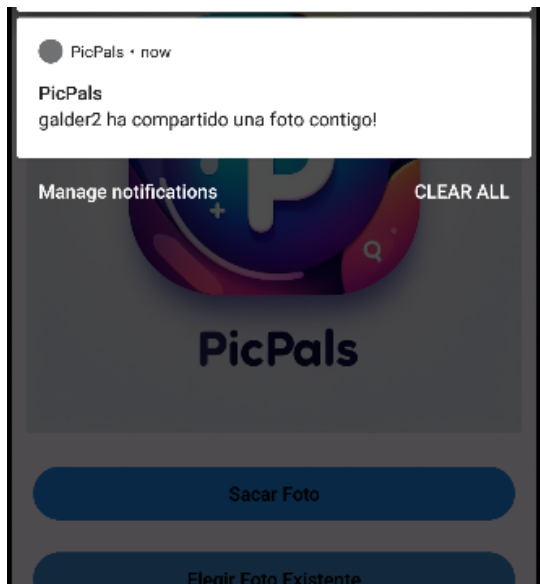


Imagen 10. Notificación dentro de la app

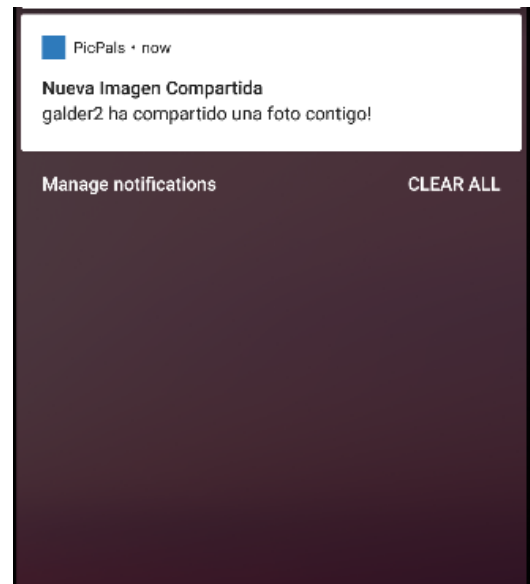


Imagen 11. Notificación fuera de la app

Requisitos cumplidos

- Se utiliza una base de datos remota para el registro y la identificación de usuarios. También se utiliza para almacenar las imágenes y la información de los distintos tipos de relaciones con los usuarios (autor de la imagen, con quién se ha compartido, etc...).
- Se utiliza mensajería FCM para notificar a los usuarios cuando alguien ha compartido una foto con ellos (mientras están logueados en algún dispositivo).
- Se puede probar la mensajería FCM a través del siguiente servicio web, que simula la petición HTTP que se enviaría desde el dispositivo cuando se comparte una foto con un usuario (lo que hace que internamente se ejecute una petición para mandar la notificación FCM): <http://34.29.139.252:81/test.php>
- La aplicación permite captar imágenes desde la cámara, también permite elegir imágenes ya existentes en la galería.
- Las imágenes se guardan en el servidor usando la base de datos, también se guarda información adicional como el autor de la imagen y con quién se ha compartido.
- La aplicación permite mostrar las imágenes guardadas en el servidor. Requiere tener un usuario logueado y permite mostrar tanto las imágenes subidas por ese usuario, como las imágenes que han sido compartidas con ese usuario.
- Se gestionan los giros para que no se pierda información. Además las actividades MainActivity y UploadImageActivity cambian el layout dependiendo de si están en portrait o landscape.



Imagen 12.

Si el dispositivo está en orientación landscape, no se muestra el logo de la aplicación.

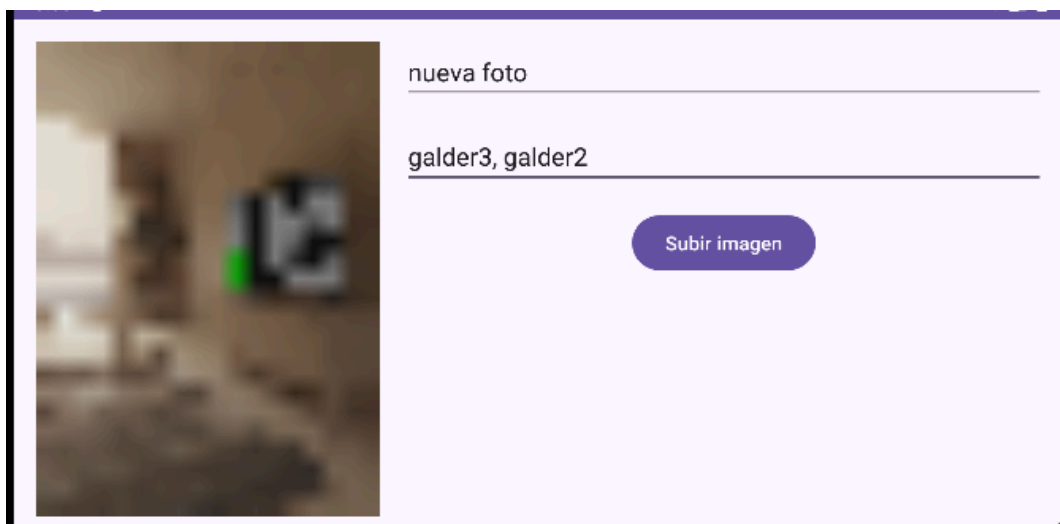


Imagen 13.

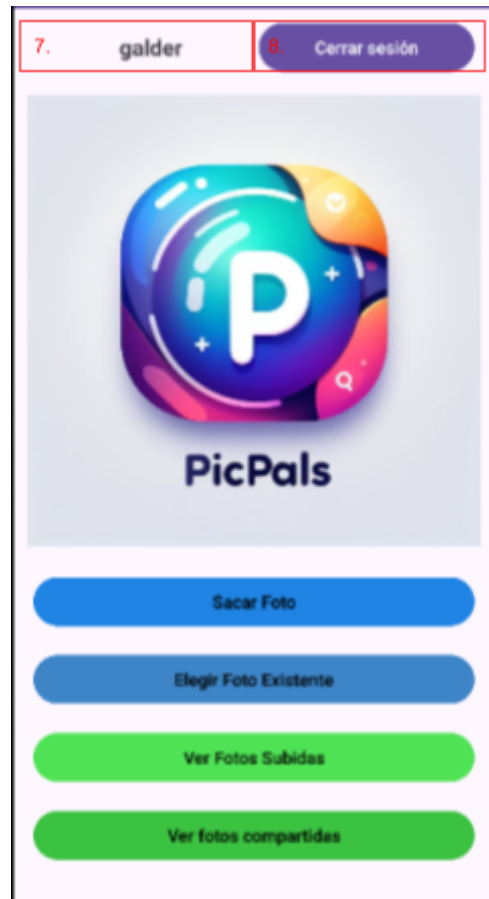
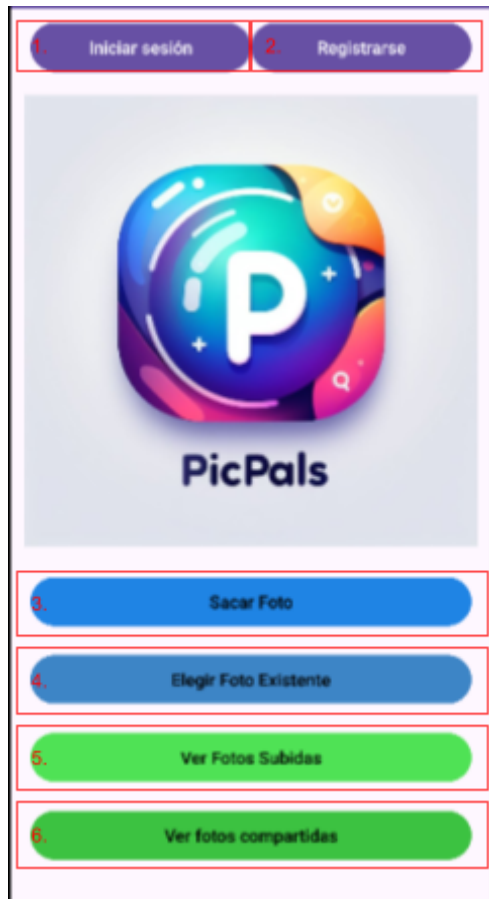
Si el dispositivo está en orientación landscape, se utiliza un layout completamente diferente.

Posibles mejoras

- Uno de los problemas más obvios que encontramos es la falta de seguridad al hacer peticiones con un usuario. Actualmente basta con mandar el nombre del usuario, esto hace que sea muy vulnerable a ataques de suplantación de identidad. Una forma de mejorar esto sería implementar un sistema de tokens temporales que se asignen cuando el usuario se loguea exitosamente y usar estos tokens para comprobar las futuras peticiones del usuario.
- Debido a la forma que tenemos de manejar las imágenes en Base64, podemos tener problemas si éstas ocupan demasiado espacio. Para evitar esto hemos limitado de forma exagerada la resolución de las imágenes. Se podría mejorar cómo gestionamos las imágenes para admitir mayor resolución.
- Añadir capacidad multi idioma como se hizo en la primera entrega.
- Añadir un menú de opciones que permita al usuario modificar preferencias de la aplicación, similar a lo que se hizo en la primera entrega.
- Permitir al usuario modificar las fotos existentes en la base de datos, ya sea borrarlas, cambiarles el nombre o compartirlas con otros usuarios.
- Implementar mejoras visuales y de UX.

Manual de usuario

Menú principal



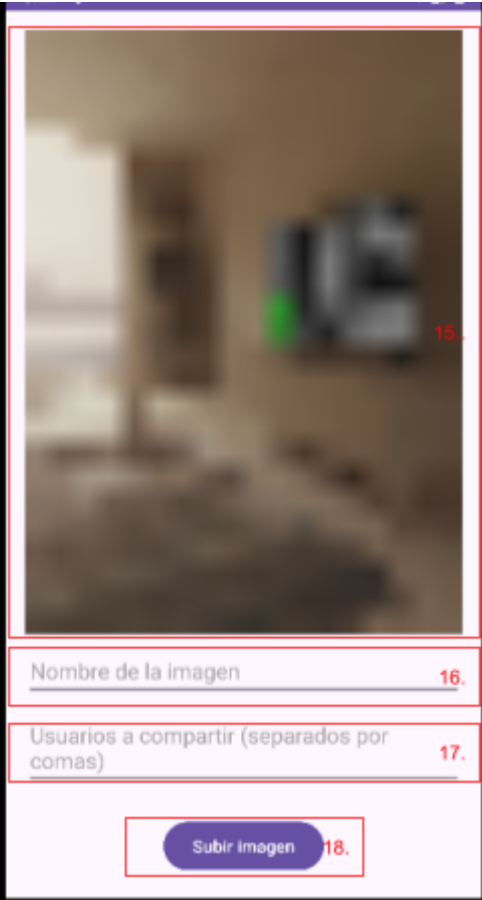
1. Botón "Iniciar sesión"
2. Botón "Registrarse"
3. Botón "Sacar Foto"
4. Botón "Elegir Foto Existente"
5. Botón "Ver Fotos Subidas"
6. Botón "Ver Fotos Compartidas"
7. Nombre del usuario logueado
8. Botón "Cerrar sesión"

Registro y logueo

The image displays two mobile application screens side-by-side. The left screen is for registration, featuring two text input fields at the top: 'Nuevo usuario' (labeled 9.) and 'Nueva contraseña' (labeled 10.). Below these fields is a blue button with the text 'Registrarse' (labeled 11.). The right screen is for login, featuring two text input fields at the top: 'Usuario' (labeled 12.) and 'Contraseña' (labeled 13.). Below these fields is a blue button with the text 'Iniciar sesión' (labeled 14.). Both screens have a light purple background and a black border.

- 9. Campo para introducir nombre de usuario a registrar
- 10. Campo para introducir contraseña a registrar
- 11. Botón "Registrarse"
- 12. Campo para introducir nombre de usuario a loguear
- 13. Campo para introducir contraseña a loguear
- 14. Botón "Iniciar sesión"

Subida de imágenes



The form is a vertical rectangle with a light purple background. It contains a large image placeholder at the top, followed by two text input fields, and a button at the bottom. Red boxes highlight the image placeholder, the first input field, the second input field, and the button. Red numbers 15., 16., 17., and 18. are placed to the right of each respective element.

15.

Nombre de la imagen 16.

Usuarios a compartir (separados por comas) 17.

Subir imagen 18.

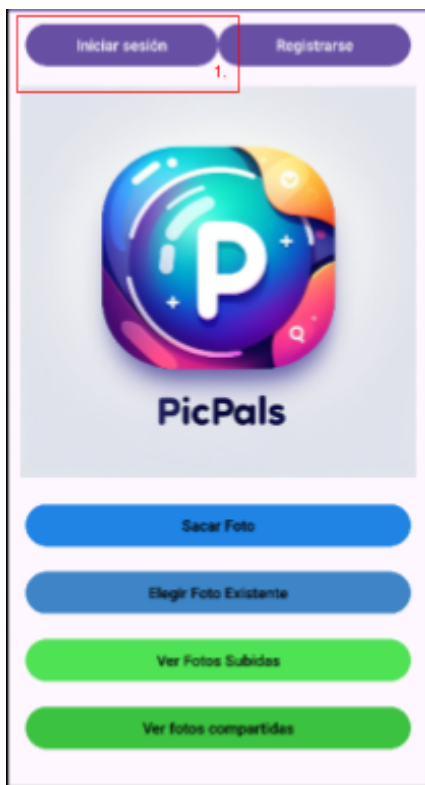
15. Imagen a subir

16. Campo para el nombre de la imagen

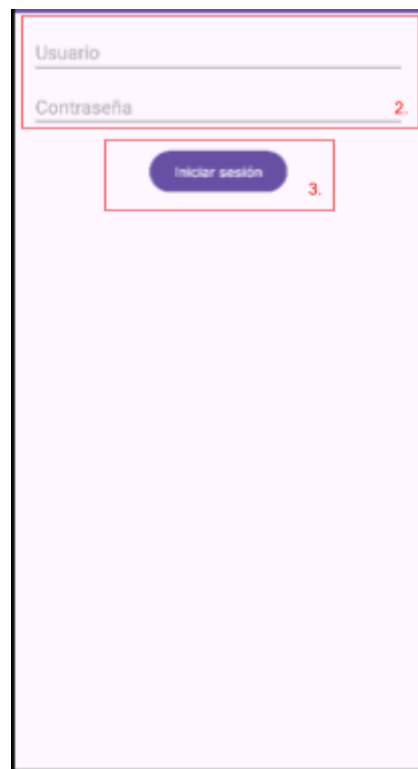
17. Campo para introducir usuarios a compartir

18. Botón "Subir Imagen"

Guía: Subir imagen desde la cámara

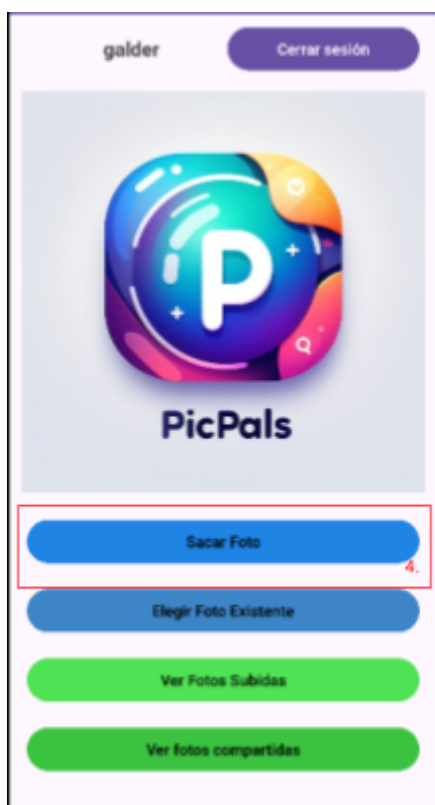


1. Pulsar “Iniciar sesión”



2. Introducir credenciales

3. Pulsar “Iniciar sesión”



4. Pulsar “Sacar Foto”



5. Sacar la foto y confirmar

Nombre de la imagen 6.

Usuarios a compartir (separados por comas) 7.

Subir imagen

- 6. Introducir nombre de la imagen
- 7. Introducir usuarios a compartir

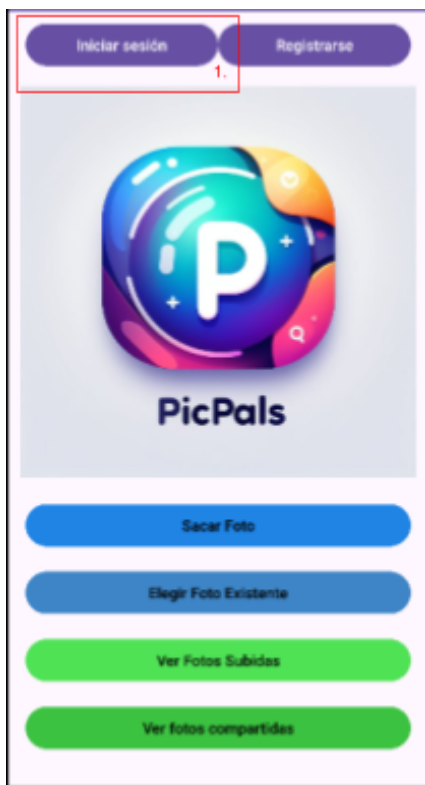
nueva foto

galder3, galder2

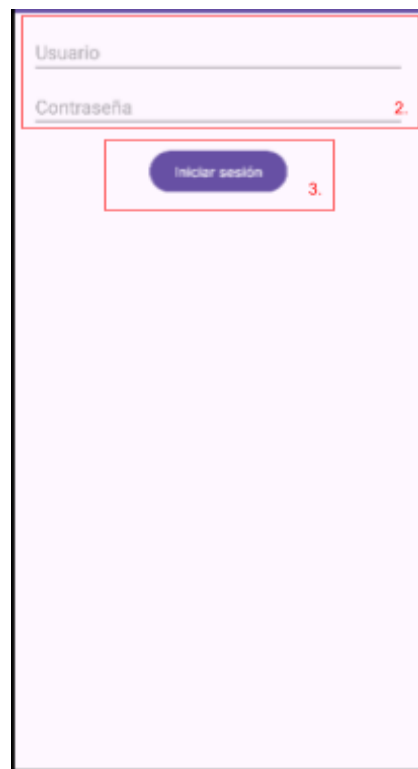
Subir imagen 8.

- 8. Pulsar “Subir imagen”

Guía: Subir imagen desde la galería

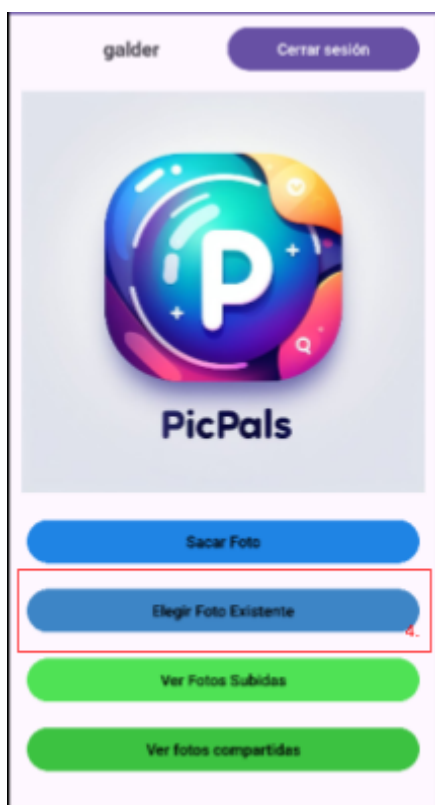


1. Pulsar “Iniciar sesión”



2. Introducir credenciales

3. Pulsar “Iniciar sesión”



4. Pulsar “Elegir Foto Existente”



5. Elegir la foto a subir

Nombre de la imagen 6.

Usuarios a compartir (separados por comas) 7.

Subir imagen

6. Introducir nombre de la imagen
7. Introducir usuarios a compartir

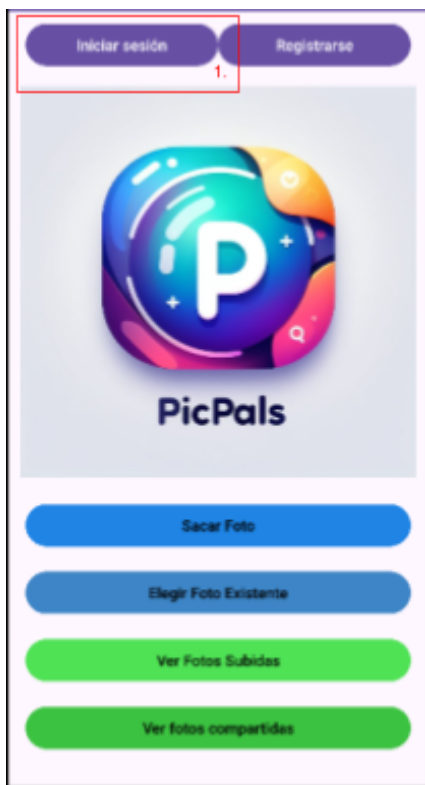
nueva foto

galder3, galder2

Subir imagen 8.

8. Pulsar "Subir imagen"

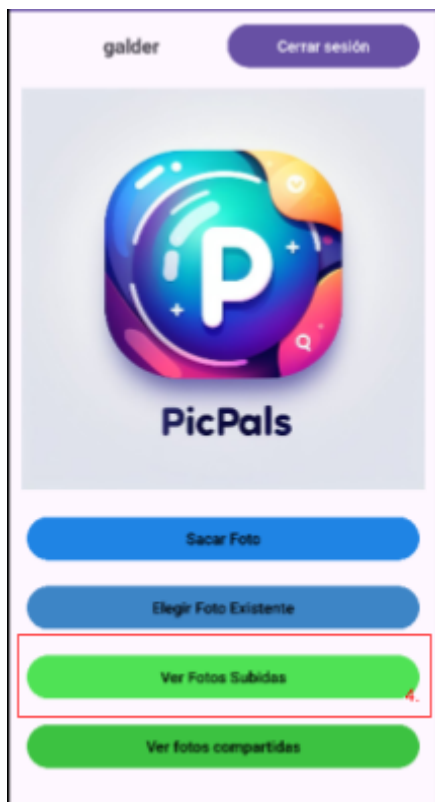
Guía: Ver imágenes subidas por el usuario



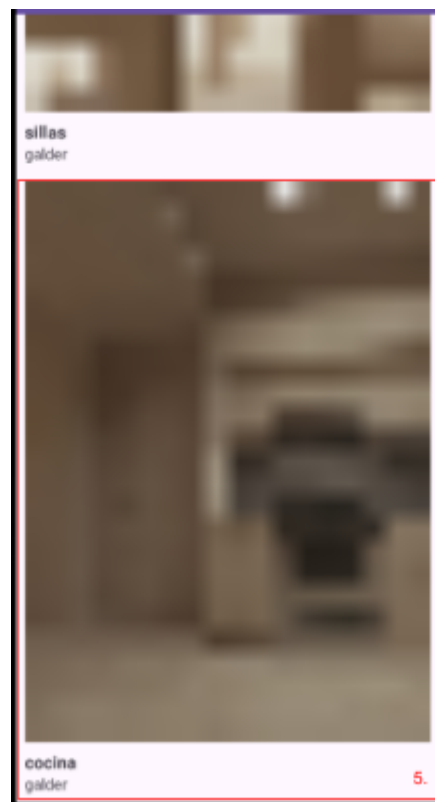
1. Pulsar “Iniciar sesión”



2. Introducir credenciales
3. Pulsar “Iniciar sesión”

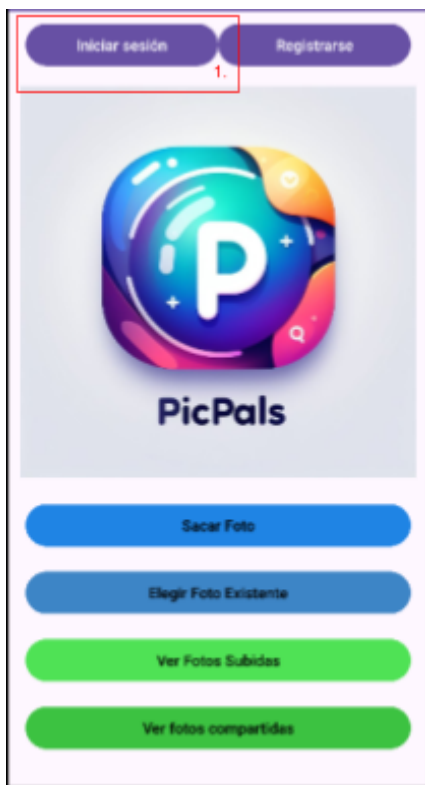


4. Pulsar “Ver Fotos Subidas”

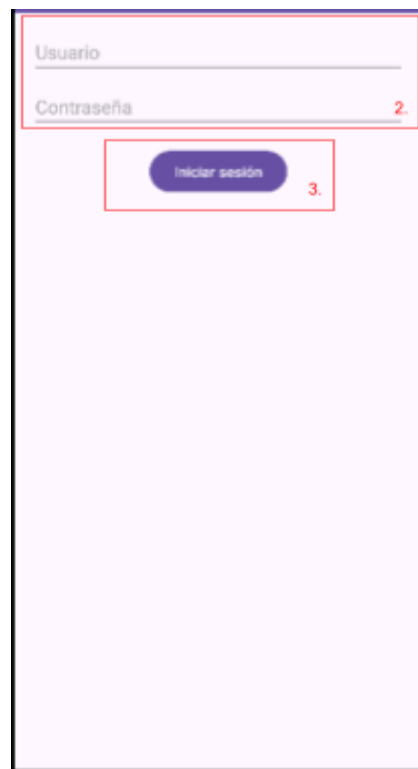


5. Se muestran las fotos del usuario

Guía: Ver imágenes compartidas con el usuario

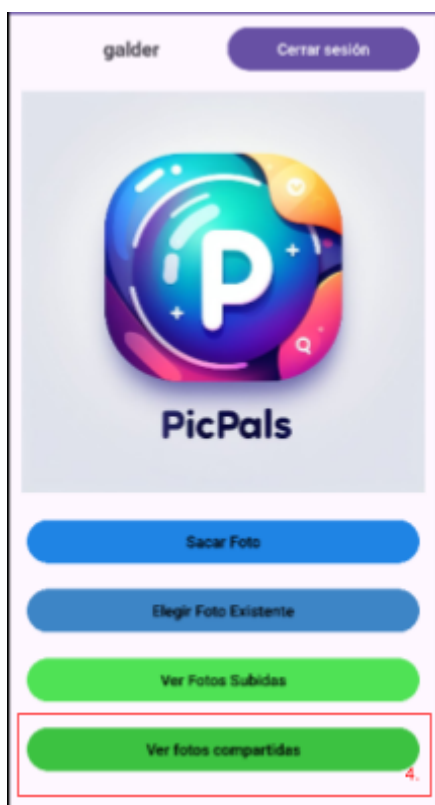


1. Pulsar “Iniciar sesión”

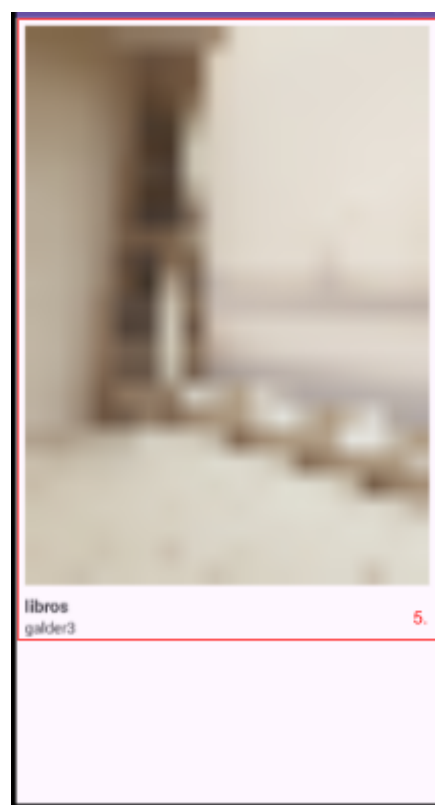


2. Introducir credenciales

3. Pulsar “Iniciar sesión”

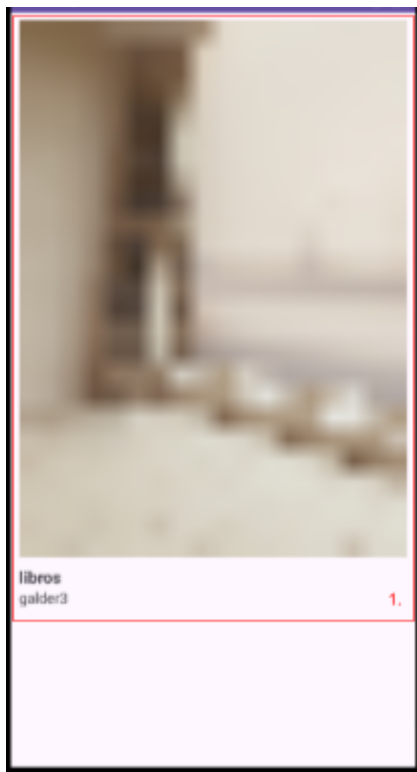


4. Pulsar “Ver Fotos Compartidas”

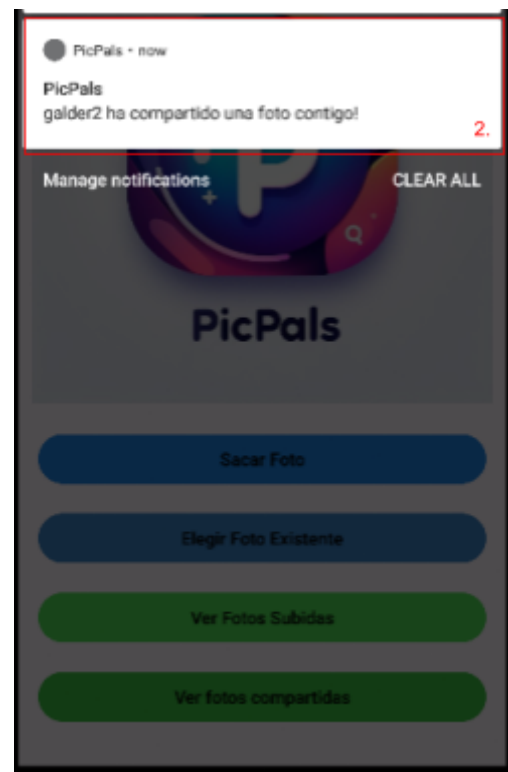


5. Se muestran las fotos compartidas

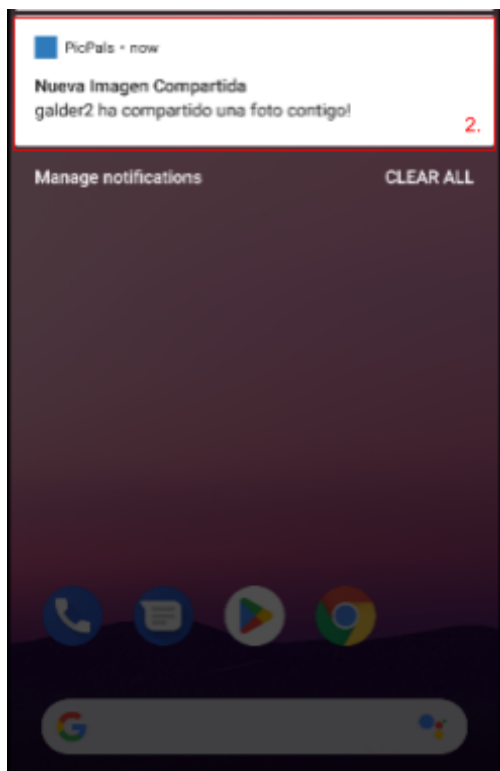
Guía: Recibiendo notificaciones



1. Solo tenemos una foto compartida



2A. Recibimos una notificación en la app



2B. Recibimos una notificación fuera de la app



3. Tenemos una nueva foto compartida

Bibliografía y recursos utilizados

<https://stackoverflow.com/questions/60155165/how-do-i-import-androidx-work-workmanager>

<https://www.youtube.com/watch?v=WYufSGgaCZ8>

<https://reqbin.com/post-online>

<https://www.color-hex.com/>

<https://stackoverflow.com/questions/31858374/android-button-background-color-not-changing>

https://www.bing.com/images/create/icono-de-aplicacion-3b3n-llamada-picpals-con-el-nombre-1-661db81bda0d4d3dbd437169a6bde508?id=KAJAQJ6QKhQFVdUtXOCddw%3d%3d&view=detailv2&idpp=genimg&idpclose=1&thld=OIG4.BEVtr5JHQE_ZxAXLPS5u&FORM=SYDBIC

<https://stackoverflow.com/questions/58312717/onmessagereceived-method-in-firebase-messaging-service-is-not-called>

<https://stackoverflow.com/questions/66567484/firebase-messaging-services-onmessagereceived-never-called>

<https://www.youtube.com/watch?v=q6TL2RyysV4>

<https://stackoverflow.com/questions/51443865/fcm-not-receiving-notification-messages>

<https://stackoverflow.com/questions/71953744/firebase-cloud-messaging-not-receiving-message-in-android-12>