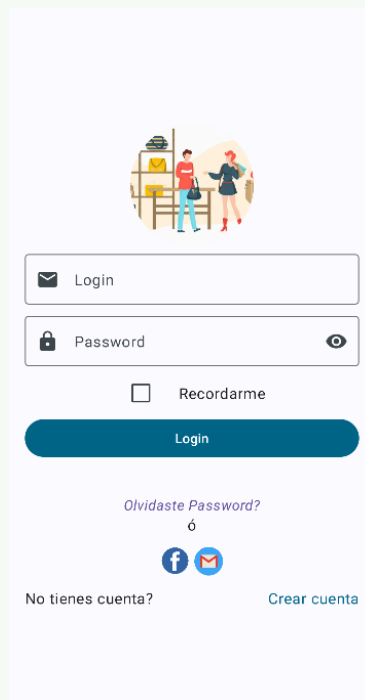


Primera versión de Login

[Descargar estos apuntes](#)

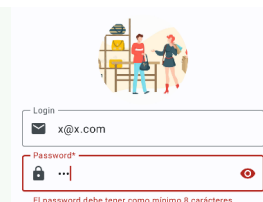
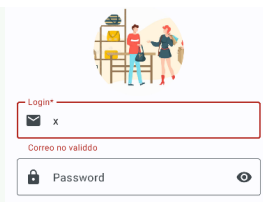
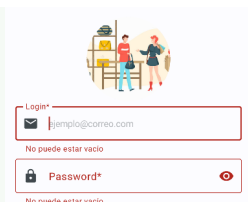
Ejercicio

Vamos a crear una aplicación con aspecto similar al siguiente, usaremos los **TextField** explicados en el tema con el tipo **Validacion**, si no queremos crear un TextField propio para el password que permita visualizar o no la contraseña (será sencillo encontrarlo en la web), podemos usar los componentes que se proporcionan en la librería que proporcionan los profesores y que está localizada en `github.pmdmiesbalmis.components`. La pantalla de la imagen, la crearemos a partir de la función **LoginScreen** que deberemos situar en el lugar correcto, según la arquitectura explicada en temas anteriores.

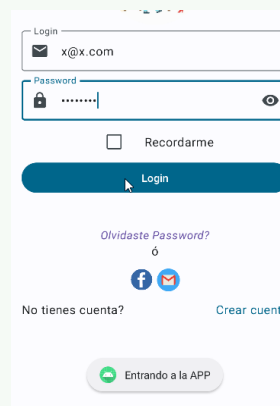
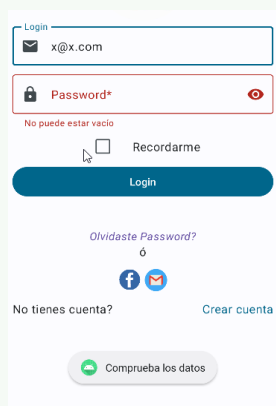


⚠ Aviso: Como ya hemos visto a lo largo del tema, para acceder a los componentes TextField modificados y poder controlar su validación, deberemos importar la librería de components alojada en **github.pmdmiesbalmis.components**, si has usado en proyecto base, estará añadida por defecto en los archivos correspondientes.

Habrás que validar las **entradas incorrectas** de los TextField, de forma que se mostrarán los mensajes de error: indicando que no pueden estar vacíos, para el correo que debe cumplir el formato de correo y para el password que debe ser mayor o igual a 8.



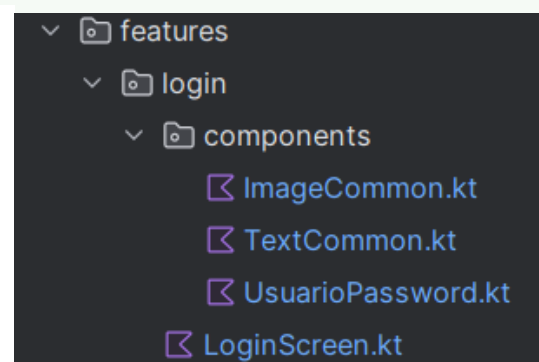
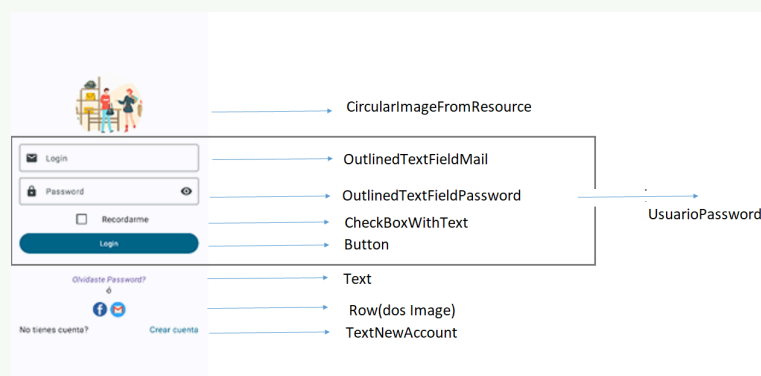
Si se pulsa el botón Login con datos que tengan el formato correcto, se mostrará un Toast con el mensaje de que se está entrando en la App. Mientras si alguno de los formatos o ambos son incorrecto, el mensaje será de aviso.



⚠️ Aviso: como en ejercicios anteriores, la lógica de la app se manejará **fuera** de los Composables que forman la screen (es decir en Preview o Activity), por lo que toda la parte de las validaciones tendrán que ir fuera de la función LoginScreen. Por tanto, necesitaremos pasar como variables de tipo State **validacionLogin** y **validacionPassword** además de los estados necesarias para los TextField a la función LoginScreen. Para definir una interfaz como estado deberemos de concretarle el tipo con `as`, por ejemplo:

```
var validacionLogin by remember { mutableStateOf(object : Validacion {} as Validacion) }
```

💡 Tips: Los componentes que debemos definir (la mayoría los podrás importar de las componentes de la librería y otros se os proporciona el código) son:



- Definidos en la librería de componentes de git hub

`OutlinedTextFieldMail` se basa en un `OutlinedTextFieldWithErrorState`

`OutlinedTextFilePassword` basado en `OutlinedTextField`

- Nos descargaremos los siguientes archivos que adjuntaremos en la carpeta componets del paquete login:
 - [ImageCommon.kt](#) que corresponderá al la imagen circular del inicio de la pantalla (CircularImageFromResource)
 - [TextCommon.kt](#) que corresponde al texto de nueva cuenta (TextNewAccount).
- El archivo **UsuarioPassword.kt** podéis implementarlo para que vuestro código quede más claro y puede incluir la inserción del login y password más checkbox y botón