

# Aplicación Login con nuevo cliente

[Descargar estos apuntes](#)


En este ejercicio vamos a partir del proyecto de login del bloque [B3\\_5\\_Hilt](#). Lo primero que haremos será **renombrar el paquete login a tienda**, ya que la idea es seguir desarrollando este proyecto en temas posteriores.

Además añadiremos un paquete más dentro de mocks, llamado cliente, al que le añadiremos la clase [ClienteMock](#) que se pasa como recurso y los respectivos ClienteDaoMock, RepositoryConverter y ClienteRepository que tendremos que construir para que nos permita añadir un cliente a la lista de clientes (La lista se pasa como recurso en el comprimido anterior), y comprobar si un cliente existe en la lista.

En el paquete **features** deberemos crear el paquete **newuser** al que le añadiremos tres paquetes más correspondientes a los tres componentes que forman la pantalla para crear nuevo usuario, [el código](#) de estos tres paquetes se pasa como recurso.

En esta parte del proyecto vamos a practicar la validación de la entrada de datos, por lo que tendremos que usar los validadores que están implementados en la librería `components` de `com.github.pmdmiesbalmis` y las clases de Validación ya explicadas en ejercicios anteriores.

Para terminar el proyecto, tendremos que crear los ficheros **NewUserUiState**, **NewUserScreen**, **ValidacionNewUserUiState**, **ValidadorNewUser** y **NewUserViewModel** de forma que las validaciones se controlen usando las clases validas del paquete validadores y siguiendo las pautas de los anteriores ejercicios.

 **Aviso:** En el código que se pasa como base, podéis ver como se implementa las clases Validacion y las clases Validador de Login y de los paquetes parciales, para que lo podáis extender a NewUser.

La clase NewUserUiState estará formada por las propiedades:

- DatosPersonalesUiState
- DireccionUiState
- NewUserPasswordUiState

La clase ValidacionNewUserUiState está constituida por:

- ValidacionDatosPersonalesUiState
- ValidacionDireccionUiState

- ValidacionLoginPasswordUiState

Por su parte el ValidadorNewUser tiene las propiedades:

- ValidadorDatosPersonales
- ValidadorDireccion
- ValidadorLoginPassword

En la pantalla principal de la aplicación cargaremos los tres componentes que os hemos pasado como recurso mediante un HorizontalPager.



Como se puede ver en las imágenes, en este caso nos podremos mover entre las pantallas con la forma normal de los pager o con el botón de siguiente, para ello tendremos que incluir en el click del botón el siguiente código:

```
val scope = rememberCoroutineScope()
scope.launch() {
    pagerState.scrollToPage(
        pagina_a_la_que_queremos_ir
    )
}
```

Se deberá de validar los siguientes casos:

- el nombre y dni no pueden vacíos
- el teléfono si se incluye debe tener 9 cifras, aunque puede estar vacío.
- los tres campos de dirección no pueden estar vacíos.
- el login y password deben de seguir las mismas normas que para el screen de login hecho anteriormente.

Comprueba los datos personales

Comprueba los datos de la dirección

Ha ocurrido un error al crear la cuenta

Se podrá mover entre paginas deslizando, aunque los datos no estén correctos. Si lo que queremos es movernos con el botón, solo pasará de página si los datos están correctos mostrará un mensaje indicando el primer dato no correcto (puedes usar un SnackBar).

Al pulsar sobre Crear Cuenta, se validará que todos los datos son correctos, mostrando el error en caso contrario. Si no hay error y ese usuario no existe en el sistema se añadirá el usuario a la lista de usuarios y el cliente a la lista de clientes.

**⚠️ Aviso:** No olvides seguir la arquitectura aprendida y aplicar la inyección de dependencia con Hilt.