

Curso:

**Fase 2 | Desarrollo Móvil Android
Kotlin Fundamentals Santander 2022**

Proyecto:

Aplicación en Android para decidir que película ver.

Integrantes:

Alejandro Pineda Hernández
Arturo Urbieto Reyes
David Miguel Muñoz García
Héctor Oscar Islas Leyva
Javier David García Vega
Ulises Falcón Romero

Proyecto: Aplicación en Android para decidir que película ver.

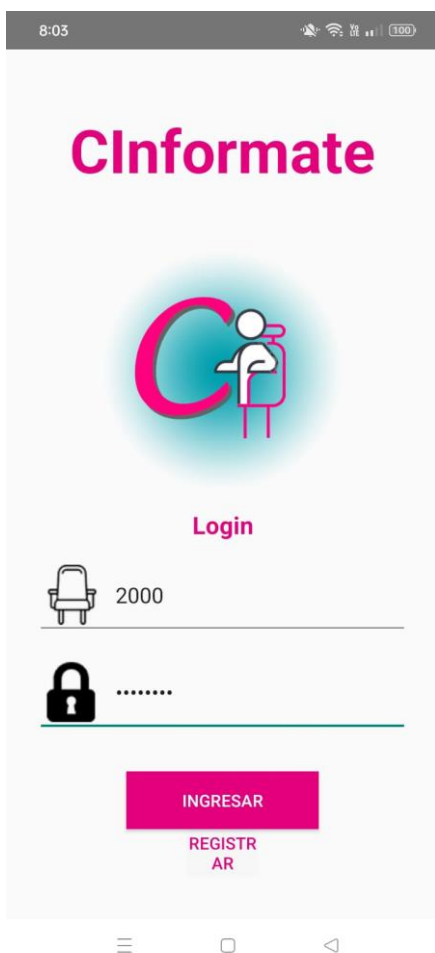
Cinformato

Objetivo:

Proyecto de catálogo de películas que se pueden comprar boletos en páginas de cine, ayudando al usuario leer y saber más de la película como una descripción, calificación y año de estreno, además al ya elegir que película se le direccionará a la página respectiva para verla.

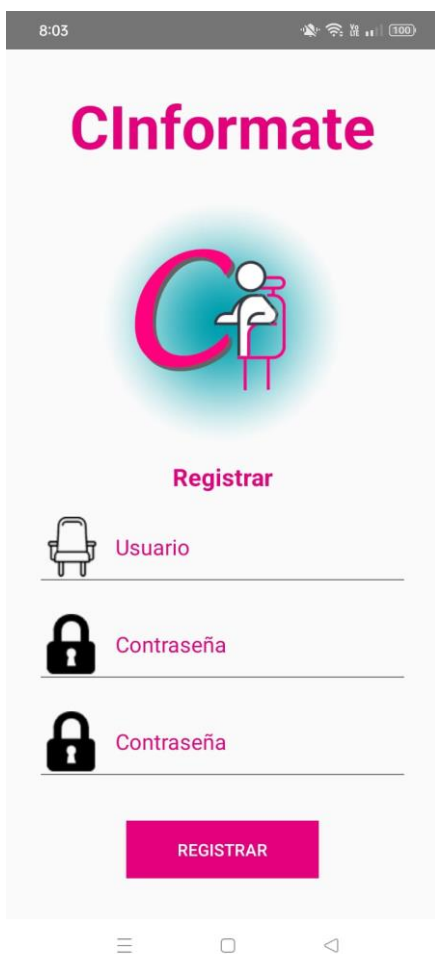
Manual de usuario

Al iniciar la aplicación nos dará dos opciones inicio de sesión o registro de usuario en caso de no tener usuario o contraseña. Para iniciar sesión se necesita el correo electrónico registrado y la contraseña.



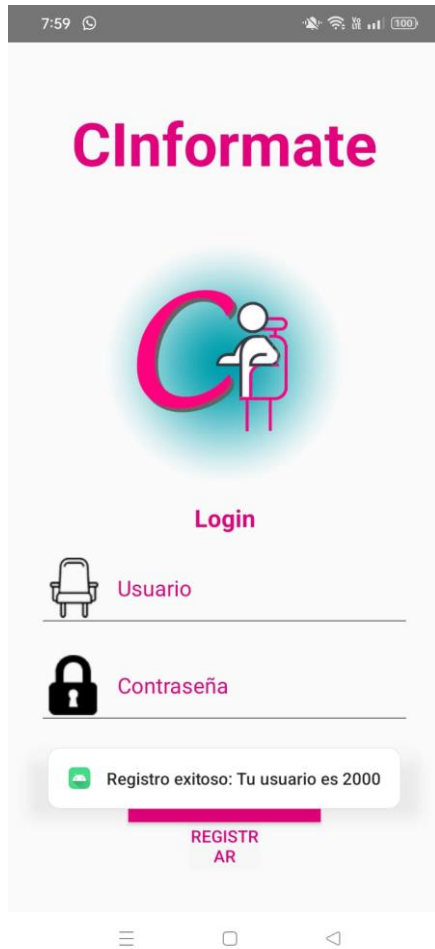
Registro

Sí se quiere registrar, agregaremos el usuario que nosotros queremos, correo electrónico y contraseña para registrarse.

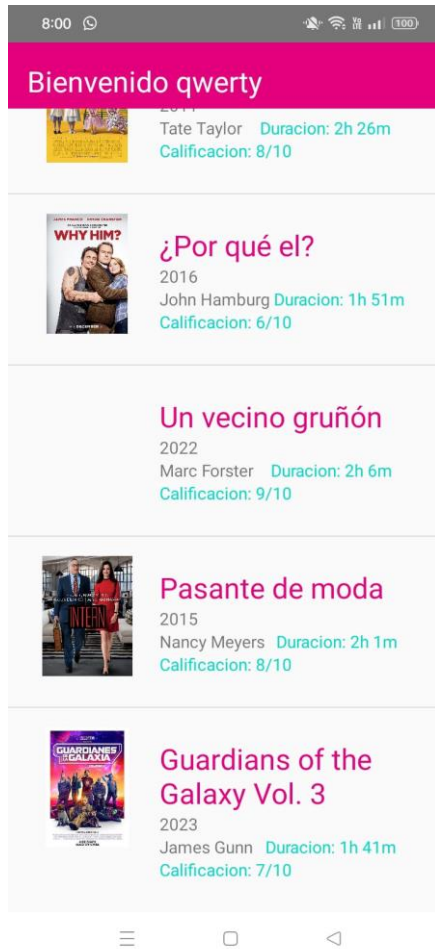


The screenshot shows a mobile application interface for 'CInformate'. At the top, the status bar displays the time 8:03 and various connectivity icons. The app header features the 'CInformate' logo in pink. Below the logo is a circular icon depicting a person sitting at a desk with a large 'C' in the background. A pink 'Registrar' button is positioned below the icon. The registration form consists of three input fields: the first is labeled 'Usuario' with a chair icon, and the next two are labeled 'Contraseña' with padlock icons. A pink 'REGISTRAR' button is located at the bottom of the form. The Android navigation bar is visible at the very bottom.

Después de registrarnos nos saldrá un mensaje con el usuario creado



Al ingresar tendremos una lista de las películas disponibles con una pequeña descripción como, título, año de estreno, director, duración y calificación.



Después seleccionaremos el título que deseemos y nos mostrará una descripción más completa de la película

8:00

Pasante de moda

Nancy Meyers
2015



A new film by NANCY MEYERS
Robert DE NIRO | Anne HATHAWAY

Calificación: 8/10 Duración: 2h 1m

Ben Whittaker, un hombre viudo de 70 años, revoluciona un negocio online dedicado al mundo de la moda cuando empieza a trabajar como pasante allí, volviéndose indispensable y muy querido entre sus compañeros, incluida Jules, la jefa, quien al principio se mostró contraria a contratarlo debido a su edad.

Tendremos la posibilidad de ir al sitio para ver esa película o regresa.

8:10

Calificación: 8/10 Duración: 2h 1m

Ben Whittaker, un hombre viudo de 70 años, revoluciona un negocio online dedicado al mundo de la moda cuando empieza a trabajar como pasante allí, volviéndose indispensable y muy querido entre sus compañeros, incluida Jules, la jefa, quien al principio se mostró contraria a contratarlo debido a su edad.

IR A VER

REGRESAR



Diagrama de Flujo del proceso

Código

User

La clase User tiene un constructor primario que recibe los valores para las propiedades "noCliente", "nombre" y "contraseña". Dentro del constructor, estas propiedades se inicializan con los valores proporcionados.

La clase User también tiene métodos para acceder y modificar las propiedades. Los métodos "getNoCliente()", "getNombre()" y "getContraseña()" son métodos de acceso que devuelven el valor de las propiedades correspondientes. Los métodos "setNombre()" y "setContraseña()" son métodos de modificación que permiten establecer nuevos valores para las propiedades "nombre" y "contraseña", respectivamente.

Además, las propiedades "noCliente", "nombre" y "contraseña" son declaradas como variables privadas dentro de la clase User. Esto significa que solo se pueden acceder a ellas desde dentro de la propia clase.

MyAdapter

El adaptador se utiliza para mostrar una lista de elementos de la clase "Movie" en una interfaz de usuario. Recibe dos parámetros en su constructor: un objeto de tipo "Context" y una lista de objetos "Movie" llamada "arrayList".

La clase "MyAdapter" implementa los métodos necesarios de la clase "BaseAdapter" para funcionar correctamente:

- El método "getCount()" devuelve el número de elementos en la lista, que se obtiene llamando al método "size" de la lista "arrayList".

- El método "getItem(position: Int)" devuelve el elemento en la posición indicada. En este caso, devuelve simplemente la posición en sí misma.
- El método "getItemId(position: Int)" devuelve un identificador único para el elemento en la posición indicada. En este caso, se devuelve la posición convertida a tipo "Long".
- El método "getView(position: Int, convertView: View?, parent: ViewGroup): View?" se encarga de crear y devolver la vista correspondiente al elemento en la posición indicada. En este método, se utiliza el enlace de datos ("ItemPelículaBinding") para inflar el diseño del elemento de película ("ItemPelículaBinding.inflate") y obtener una referencia a los elementos de la interfaz de usuario dentro del diseño.

Luego, se asignan los valores correspondientes a los elementos de la vista utilizando los datos del objeto "Movie" en la posición indicada de la lista "arrayList". Finalmente, se utiliza la biblioteca Picasso para cargar la imagen de la película desde la URL proporcionada en el objeto "Movie" en el elemento correspondiente de la vista.

Variables

1. Usuarios:
 - Se definen dos objetos de la clase "User": "USER1" y "USER2". Cada uno tiene un número de cliente, nombre y contraseña.
 - Se crea una lista mutable de usuarios llamada "USERS" y se le asignan los objetos "USER1" y "USER2".
 - Se define una variable "NoUser" con el valor 2000.
2. Películas:
 - Se definen seis objetos de la clase "Movie": "movie1", "movie2", "movie3", "movie4", "movie5" y "movie6". Cada uno tiene propiedades como el título, año, calificación, director, duración, URL de la imagen y URL de la película.
 - Se crea una lista mutable de enteros llamada "images" que está vacía.

HomeActivity

- La clase "HomeActivity" define una serie de variables y objetos que se utilizarán en la actividad.
- En el método "onCreate", se infla el diseño de la actividad utilizando el enlace de datos "ActivityHomeBinding" y se establece como el contenido de la actividad.

- Se obtiene un "Bundle" que contiene los extras pasados a la actividad a través del intent.
- Se obtiene el valor de "count" del "Bundle" y se utiliza para mostrar el nombre de usuario correspondiente en un elemento de la vista de texto.
- Se crea un "ArrayList" llamado "peliculas" y se agregan objetos de la clase "Movie" a la lista.
- Se crea un adaptador personalizado llamado "itemAdapter" de la clase "MyAdapter" y se le pasa la lista de películas y el contexto actual.
- Se establece el adaptador en un ListView llamado "listaPeliculas" mediante la propiedad "adapter".
- Se configura un listener de clics en los elementos de la lista. Cuando se hace clic en un elemento, se crea un intent para abrir la actividad "MovieDetailsActivity" y se pasan los detalles de la película seleccionada como extras en el intent.

La clase "HomeActivity" define la lógica de la actividad principal de la aplicación. Se infla el diseño, se muestra el nombre de usuario, se crea una lista de películas y se configura un adaptador personalizado para mostrar las películas en un ListView. Además, se establece un listener de clics en los elementos de la lista para abrir una nueva actividad y mostrar los detalles de la película seleccionada.

LoginActivity

- Se declaran las siguientes variables como propiedades de la clase **LoginActivity**:
- **homeTitle**: TextView para mostrar el título de inicio de sesión en la pantalla.
- **usuario**: EditText para que el usuario ingrese su nombre de usuario.
- **password**: EditText para que el usuario ingrese su contraseña.
- **inicioButton**: Button para el botón de inicio de sesión.
- **registroButton**: Button para el botón de registro.
- Se anota el método **onCreate** con **@SuppressWarnings("ResourceAsColor")**, lo cual suprime una advertencia específica relacionada con el uso de recursos de color.
- Dentro del método **onCreate**, se establece el diseño de la actividad utilizando **setContentView(R.layout.activity_main)**, que vincula la actividad con el archivo de diseño XML correspondiente.
- Se inicializan las variables declaradas anteriormente mediante la búsqueda de las vistas correspondientes en el archivo de diseño XML utilizando **findViewById**.

- Se establece un **OnClickListener** para el botón de inicio de sesión (**inicioButton**). Cuando se hace clic en este botón, se ejecuta el código dentro de la función **onClick**.
- Dentro del **OnClickListener**, se realiza un bucle **for** a través de una lista llamada **USERS**. Para cada elemento **i** en la lista, se verifica si el número de cliente (**i.getNoCliente()**) coincide con el texto ingresado en el campo de usuario. Si es así, se verifica si la contraseña (**i.getContraseña()**) coincide con la contraseña ingresada en el campo de contraseña.
- Si el número de cliente y la contraseña coinciden, se imprime el valor de **count**, se crea un **Bundle** que almacena el valor de **count** y se crea un **Intent** para iniciar la **HomeActivity**. El **Bundle** se pasa al **Intent** utilizando **putExtras**.
- Si la contraseña no coincide, se muestra un mensaje de toast indicando que la contraseña es incorrecta.
- El botón de registro (**registroButton**) también tiene un **OnClickListener** que inicia la **RegisterActivity** cuando se hace clic en él.

Define una actividad de inicio de sesión en una aplicación de Android. Al ingresar un número de cliente y una contraseña, se verifica si coinciden con los valores almacenados en la lista de usuarios. Si coinciden, se inicia la actividad de inicio de sesión. Si no coinciden, se muestra un mensaje de contraseña incorrecta. El botón de registro inicia la actividad de registro.

MovieDetailsActivity

- Se declaran las siguientes variables como propiedades de la clase **MovieDetailsActivity**:
- **goToMovie**: Button utilizado para abrir un enlace relacionado con la película.
- **binding**: Objeto de tipo **ActivityMovieDetailsBinding** utilizado para acceder a las vistas y recursos definidos en el archivo de diseño XML correspondiente.
- En el método **onCreate**, se inicializa la variable **binding** utilizando el método **inflate** de **ActivityMovieDetailsBinding**, que infla el archivo de diseño XML **ActivityMovieDetailsBinding** y asigna las vistas correspondientes.
- Se establece el diseño de la actividad utilizando **setContentView(binding.root)**, que vincula la actividad con la vista raíz del archivo de diseño XML.
- Se inicializa la variable **goToMovie** mediante la búsqueda de la vista correspondiente en el archivo de diseño XML utilizando **findViewById**.

- Se recuperan los datos pasados a esta actividad a través del intent utilizando los métodos **getStringExtra**. Estos datos incluyen el título de la película, el año de lanzamiento, la calificación, el director, la duración, la URL de la imagen, el enlace y la descripción.
- Se asignan los datos recuperados a las vistas correspondientes en el archivo de diseño XML utilizando **binding.movieTitle.text**, **binding.movieDescription.text**, etc.
- Se utiliza la biblioteca Picasso para cargar la imagen de la URL en la vista de la imagen de la película utilizando **Picasso.get().load(imageUrl).into(binding.movieImage)**.
- Se establece un **OnClickListener** para el botón **goToMovie**. Cuando se hace clic en este botón, se crea un **Intent** para abrir la URL relacionada con la película utilizando **Intent.ACTION_VIEW** y se asigna la URL al intent utilizando **openURL.data = Uri.parse(link)**. Luego se inicia la actividad correspondiente para abrir la URL utilizando **startActivity(openURL)**.
- Se declara una función **closeDialog** que se puede llamar desde la vista para cerrar la actividad actual utilizando **finish()**. Esta función no se utiliza en el código proporcionado, pero se puede utilizar para realizar ciertas acciones al cerrar la actividad.

En resumen, este código define una actividad en Android llamada **MovieDetailsActivity** que muestra los detalles de una película. Los detalles se reciben a través de un intent y se asignan a las vistas correspondientes en el archivo de diseño XML. Además, hay un botón que abre un enlace relacionado con la película y una función para cerrar la actividad.

RegisterActivity

- Se declara la clase **RegisterActivity**, que representa la actividad de registro de usuario.
- Se declaran las siguientes variables como propiedades de la clase **RegisterActivity**:
 - **usuarioRegistro**: EditText para que el usuario ingrese su nombre de usuario durante el registro.
 - **passwordRegistro**: EditText para que el usuario ingrese su contraseña durante el registro.
 - **passwordRegistroConfirmacion**: EditText para que el usuario ingrese nuevamente su contraseña durante el registro para confirmarla.
- **inicioButton**: Button para el botón de registro.

- En el método **onCreate**, se establece el diseño de la actividad utilizando **setContentView(R.layout.activity_register)**, que vincula la actividad con el archivo de diseño XML correspondiente.
- Se inicializan las variables **usuarioRegistro**, **passwordRegistro** y **passwordRegistroConfirmacion** mediante la búsqueda de las vistas correspondientes en el archivo de diseño XML utilizando **findViewById**.
- Se establece un **OnClickListener** para el botón de registro (**inicioButton**). Cuando se hace clic en este botón, se ejecuta el código dentro de la función **onClick**.
- Dentro del **OnClickListener**, se realizan comprobaciones para verificar si el campo de nombre de usuario no está vacío (**!usuarioRegistro.text.toString().isEmpty()**) y si la longitud de las contraseñas es mayor a 7 (**passwordRegistro.text.toString().length > 7 && passwordRegistroConfirmacion.text.toString().length > 7**).
- Si se cumplen las condiciones anteriores, se verifica si las contraseñas ingresadas son iguales (**passwordRegistro.text.toString().equals(passwordRegistroConfirmacion.text.toString())**). Si coinciden, se crea un nuevo objeto **User** con el número de usuario incrementado, el nombre de usuario y la contraseña, y se agrega a la lista **USERS**. Luego se muestra un mensaje de toast indicando el éxito del registro y el número de usuario registrado. Finalmente, se crea un **Intent** para iniciar la actividad de inicio de sesión (**LoginActivity**) y se inicia la actividad correspondiente utilizando **startActivity(intent)**.
- Si las contraseñas no coinciden, se muestra un mensaje de toast indicando que las contraseñas no son iguales.
- Si no se cumplen las condiciones de validación de nombre de usuario y longitud de contraseña, se muestran mensajes de toast correspondientes para informar al usuario.

Define una actividad en Android llamada **RegisterActivity** que permite a los usuarios registrarse ingresando un nombre de usuario y una contraseña. Se realizan validaciones en los datos ingresados y se muestra un mensaje de éxito o error según corresponda.