Alumno: Javier Martínez Arias Máster MIMO – UPSA

Nombre del proyecto: Rickpedia

# Documentación - Proyecto final Desarrollo de aplicaciones cross-platform

## Introducción

En este documento se va a presentar la aplicación desarrollada como proyecto final de la asignatura Desarrollo de aplicaciones cross-platform.

Se trata de una aplicación cuya temática está basada en la serie de animación “Rick y Morty”. En ella, se muestra un listado de contenidos diferenciados en 3 categorías: personajes, ubicaciones y episodios de la serie. A partir de los elementos del listado, podemos acceder al detalle de cada uno de ellos y añadirlos a listas de contenidos favoritos, según la categoría a la que pertenecen. Toda esta información estará asociada al perfil del usuario identificado, ya que la aplicación permite tener hasta 5 perfiles de usuario por dispositivo, cada uno de ellos con sus preferencias de contenido.

Por último, la aplicación incluye un juego que consiste en adivinar los personajes de la serie, dado el nombre y 4 imágenes, siendo solo una de ellas la opción correcta.

Así podemos consultar el registro de partidas de los diferentes perfiles de la aplicación y competir entre ellos.

La aplicación consta de un flujo de pantallas que se van sucediendo en un orden coherente. En primer lugar, se nos da la bienvenida a la app, dándonos la opción de identificarnos si ya tenemos un perfil, o bien crear uno nuevo. Una vez identificados, pasaremos a la pantalla de home, que nos permite acceder a todas las secciones disponibles. Por defecto, se muestra la pantalla de contenidos sugeridos. Podremos navegar principalmente hacia el listado por categorías, juego y la sección del perfil.

Con esta aplicación se pretende poner de manifiesto las capacidades obtenidas durante el curso, según el temario visto durante la asignatura.

## Librerías utilizadas

Para la realización de este proyecto, además de la librería principal de react native (y todas aquellas que vienen durante la instalación por defecto), se han utilizado otros módulos, entre los que identificamos:

* **@react-native-community/async-storage:** Librería que permite almacenar a nivel de fichero, las sesiones de usuario abiertas desde el último uso de la aplicación. De esta forma, evitamos al usuario el paso de identificarse cada vez, si lo había hecho ya con anterioridad. Al cambiar de perfil, se elimina la información de la sesión, para que no vuelva a iniciarse con su usuario por defecto. La información guardada para la sesión es el nickname del usuario, y un token aleatorio de acceso de la última sesión, que se irá actualizando en cada sesión por seguridad.
* **@react-navigation/bottom-tabs:** Librería encargada de gestionar la navegación a nivel de pestañas en la parte inferior de la pantalla. Se ha utilizado para navegar entre los distintos apartados de la aplicación, una vez el usuario se ha identificado.
* **@react-navigation/native:** Módulo de React Navigation que permite obtener el componente *“NavigationContainer”, necesario para encapsular* la pila de pantallas de navegación en un “*Wrapper*” interpretable por los sistemas de navegación nativos de los dispositivos móviles.
* **@react-navigation/stack:** Librería encargada de gestionar la pila de navegaciones disponible para cada sección o pestaña de la aplicación. Se ha utilizado para navegar entre los distintos apartados de la aplicación, una vez el usuario se ha identificado.
* **axios:** Librería que permite realizar peticiones HTTP desde la aplicación hacia las distintas APIs utilizadas y gestionar las respuestas recibidas. En este caso, las peticiones realizadas han sido de tipo GET, pues solo se quería obtener la información que tienen estas APIs. Sin embargo, la librería permite realizar todo tipo de peticiones: POST, PUT, etc.
* **react:** Se trata del Corede React, que permite hacer uso de todas las funcionalidades propias del framework: componentes, hooks, ciclo de vida, etc.
* **react-native:** Se trata del Corede React-Native, que permite combinar las funcionalidades de React con las funcionalidades y componentes propios, compatibles con los Sistemas operativos Android e IOS.
* **react-native-gesture-handler:** Modulo que permite manejar los gestos del usuario en los dispositivos, y asociarles una funcionalidad. En este caso, se ha incluido por ser requerida por la librería *@react-navigation/stack.*
* **react-native-get-random-values:** Librería que permite la generación aleatoria de valores. Se ha instalado por ser requerida en el módulo uuid, que veremos a continuación.
* **react-native-safe-area-context:** Modulo que permite detectar el viewport disponible en cada pantalla, respetando los márgenes “safe”.Se ha incluido por ser requerida en la instalación de “*react-navigation”.*
* **react-native-screens:** Modulo que permite gestionar las pantallas de la aplicación de react native. Se ha incluido por ser requerida en la instalación de “*react-navigation”.*
* **react-native-sqlite-2:** Librería que permite gestionar la persistencia de datos de la aplicación a nivel de base de datos. Se trata de una base de datos, con el motor sqlite, que almacena de forma segura la información de los usuarios de la aplicación y todos sus contenidos relacionados. Por tanto, nos permite crear tablas y trabajar con ellas.
* **react-native-vector-icons:** Librería que permite incluir iconos vectoriales en la aplicación, mejorando la identificación de ciertos elementos con los que interactuar, así como la estética.
* **uuid:** Librería que permite generar identificadores únicos de tipo alfanumérico. Se ha utilizado para crear tokens de sesión aleatorios, cada vez que el usuario inicia sesión. Si se modificara a mano el token generado en el fichero de *“async-storage”* podrían producirse errores de identificación.

## Descripción funcional

A nivel funcional, la aplicación se desarrolla en un total de 5 flujos principales de navegación, cada uno de ellos consta de varias pantallas que analizaremos según esta clasificación.

### Flujo de navegación de bienvenida

* **Bienvenida/Acceso**: Se trata del primer flujo que encontramos al acceder a la aplicación. Es el que nos permite acceder a nuestro perfil o crear uno nuevo si no tenemos ya uno creado. Superado este proceso se avanza hacia la siguiente pantalla y navegación, de Home/Inicio.

Este flujo puede verse modificado la segunda vez que iniciamos la aplicación, pasando directamente de la bienvenida al home gracias a la sesión guardada en fichero de configuración (*AsyncStorage*), que se establece por defecto, si un usuario se ha identificado previamente y no ha hecho logout.

Este primer flujo está compuesto por una sola vista:

* + Welcome: vista que nos permite realizar el comportamiento descrito con anterioridad. Para ello, está compuesta por un listado de elementos con nombre y avatar (asignado automáticamente), que, al pulsar sobre alguno de ellos, nos identifica y nos da acceso a la Home. El listado se crea con un FlatList Horizontal.

Si no hemos creado ningún perfil, este listado esta vacío y no habrá elementos en la lista con los que interactuar. Por otro lado, tenemos un campo de texto donde introducir el “*nickname”* del perfil a crear y un botón para confirmar. Al pulsar, se añadirá un elemento al listado, con el nombre introducido y un avatar asociado.

Interfaz de usuario gráfica, Aplicación, Sitio web

Descripción generada automáticamente Una captura de pantalla de un celular con la imagen de una caricatura

Descripción generada automáticamente con confianza media

### Flujos de navegación principal

Una vez analizada la pantalla de bienvenida e identificación, pasamos a analizar los principales flujos de navegación, disponibles para los usuarios identificados.

Son aquellos flujos correspondientes con las opciones que nos ofrece el BottomTabNavigator, es decir: Home, Categorías, Juego y Perfil. Cada uno de ellos se encuentra encapsulado dentro de un Stack o pila de navegación, que veremos a continuación:

* **Home**: Se trata de uno de los flujos principales de la aplicación, ya que es el primero al que se accede una vez el usuario se ha identificado. Al cargar el componente podemos distinguir varios elementos:
  + En la barra superior, tenemos el título de sección y un botón para refrescar el contenido sugerido.
  + Título indicando el tipo de contenido sugerido.
  + Listado de elementos sobre el tipo de contenido sugerido. Siempre se muestran un total de 10 elementos independientemente del tipo. Cada uno de ellos se corresponde con el mismo componente, formado por una imagen y 3 textos. Este listado, se obtiene haciendo uso de FlatList con disposición vertical.
  + En la parte inferior se muestra el BottomNavigationTab, con las 4 opciones comentadas anteriormente.

Si pulsamos en alguno de los ítems del listado, accedemos al detalle correspondiente. Se trata de una pantalla cuyo componente es reutilizado en muchos otros flujos de navegación principal, como veremos más adelante.

* + **Detalle:** Pantalla ampliando la información que se ve a simple vista del elemento pulsado. En ella tenemos:
    - Barra superior, con el título del contenido seleccionado.
    - Imagen del contenido.
    - Título del contenido.
    - Subtítulo del contenido.
    - Información adicional del contenido.
    - Descripción y enlace externo al artículo con más información.
    - Listado de elementos relacionados con el contenido: En caso de ser un contenido de tipo personaje, el listado será de episodios y en otro caso, de personajes. Este listado está hecho con FlatList vertical y al pulsar en alguno de los elementos nos llevará al detalle de cada uno.
    - Solo en caso de ser contenido de tipo personaje, se muestra también la ubicación de origen y la actual correspondientes. Al pulsar sobre ellas se redirige al detalle asociado de la ubicación.
* **Categorías**: Flujo de navegación que permite visualizar las categorías en las que se encuentran clasificados los distintos contenidos. Al cargar el componente asociado, tenemos:
  + Barra superior con el título de sección.
  + Listado de categorías. En este caso tenemos siempre los 3 tipos de categorías comentadas: Personajes, Ubicaciones y episodios. Cada elemento de la lista tiene un ImageBackground y un texto asociado, con el nombre de la categoría. Nuevamente, el listado se crea con una FlatList en disposición vertical.
  + En la parte inferior se muestra el BottomNavigationTab.

Si pulsamos en alguno de los ítems de categorías, accedemos al listado de contenido de la categoría seleccionada. Se trata de una pantalla cuya vista es muy similar a la de Home, por mostrar un listado de contenidos, en este caso filtrado según la selección:

* **Listado de contenido por categoría:**
  + Barra superior, con el título de la categoría seleccionada.
  + Listado de elementos de la categoría. Se muestran siempre un máximo de 20 elementos por página. Se utiliza para ello un FlatList vertical.
  + Footer con la paginación disponible de elementos: Primera, anterior, actual, siguiente y última página. Permite organizar el contenido e mostrando poco a poco, de 20 en 20, actualizándolo cada vez que se pulsa en algún ítem de la paginación. En algunos casos, estos botones se deshabilitan, cuando no tienen más contenido que mostrar.
* **Detalle de contenido:** Como ya ocurría en la vista de home, si aquí pulsamos sobre algún elemento de la lista anterior, accedemos al detalle de contenido del mismo. La vista es la misma, al reutilizar el componente ya visto.
* **Juego:** Flujo de navegación que permite jugar a adivinar la imagen del personaje de Rick y Morty, conociendo su nombre. El primer componente principal consta de:
  + Barra superior con el título de sección.