MEMORIA - Movie Explorer: Aplicación React Native

Github: https://github.com/Mariinaaa/movies-expo-app

Marina Amaya y Cristina Barandiarán

Contenido

1. Introducción y Objetivos

Objetivo General

Portar la aplicación web de explorador de películas (desarrollada en React con API de TMDB en la práctica 2) a la plataforma React Native utilizando Expo, manteniendo toda la funcionalidad y adaptándola para dispositivos móviles.

Objetivos Específicos

- Implementar la visualización de películas populares desde TMDB
- Desarrollar sistema de favoritos con persistencia local
- Crear interfaz de navegación intuitiva y responsive
- Adaptar el diseño para diferentes tamaños de pantalla
- Implementar sistema de temas (oscuro/claro) como funcionalidad bonus
- Crear pantalla de detalle de películas con información completa
- Ejecutar la aplicación correctamente en Expo Go en web, iOS y Android

2. Tecnologías Utilizadas

Tecnología	Versión	Propósito
React Native	0.81.5	Framework base para desarrollo móvil
React	19.1.0	Librería de componentes
Expo	~54.0.25	Plataforma de desarrollo y deployment
React Navigation	7.1.22	Sistema de navegación (Tabs + Stack)
@react-navigation/native-stack	Última	Stack Navigator para pantalla de detalle
Axios	1.13.2	Peticiones HTTP a API TMDB
AsyncStorage	2.2.0	Persistencia de datos local (equivalente a localStorage)
Ionicons	15.0.3	Librería de iconos

API Externa

- **The Movie Database (TMDB)**
    - URL Base: https://api.themoviedb.org/3
    - Endpoints: /movie/popular, /search/movie
    - Autenticación: API Key (gratuita)

3. Análisis de Requisitos y Cumplimiento

Opción 1: Requisitos Específicos

Página Principal

La página principal de la aplicación muestra las películas populares al cargar la app. Incluye una barra de búsqueda funcional que permite filtrar películas en tiempo real. Mientras se obtienen los datos de la API de TMDB, se muestra un indicador de carga para mejorar la experiencia del usuario. El grid de películas es responsivo, adaptándose entre 2 y 5 columnas según el tamaño de pantalla.
Archivo: src/screens/Home.jsx

Sistema de Favoritos

La página principal de la aplicación muestra las películas populares al cargar la app. Incluye una barra de búsqueda funcional que permite filtrar películas en tiempo real. Mientras se obtienen los datos de la API de TMDB, se muestra un indicador de carga para mejorar la experiencia del usuario. El grid de películas es responsivo, adaptándose entre 2 y 5 columnas según el tamaño de pantalla.
Archivo: src/screens/Home.jsx

Navegación

La aplicación incluye un NavBar fijo que muestra el logo y título de la app. Los enlaces de navegación permiten acceder a las pantallas de Home y Favorites. La navegación se implementa mediante React Navigation, adaptando la funcionalidad que React Router tenía en la versión web a la plataforma móvil. Todo el diseño es responsive, asegurando una experiencia fluida en diferentes tamaños de pantalla.
Archivos: src/components/NavBar.jsx, App.jsx

Tarjetas de Películas

Cada tarjeta de película muestra la imagen del póster obtenida desde TMDB, junto con el título y el año de lanzamiento. Además, incluye un overlay que muestra información adicional como título, año, rating y botón de favoritos. El rating de la película se presenta con un código de color: verde para ≥7, amarillo para 5-7 y rojo para <5, ofreciendo una referencia visual rápida.
Archivo: src/components/MovieCard.jsx

Opción 2: Elección Libre

- Al menos 4 pantallas: Home, Favorites, Profile, MovieDetail
- Consumo de API TMDB
- Navegación con react-navigation (Bottom Tabs + Stack)
- Búsqueda/filtrado funcional
- Funcionamiento responsive en PC y móvil
- Implementación de tema oscuro/claro con persistencia
- Pantalla de detalle con información completa de cada película

4. Componentes Implementados

App.jsx

Componente principal de la aplicación. Configura los providers ThemeProvider y MovieProvider para manejo global de estado. Además, establece la navegación utilizando React Navigation, combinando Tab Navigator y Stack Navigator para permitir navegación entre pantallas y detalle de películas.

NavBar.jsx

Barra de navegación con logo y enlaces a Home y Favorites. Tiene altura de 40px para no interferir con la barra de estado en móviles. Se integra con el sistema de temas, adaptando colores a modo oscuro o claro.

Home.jsx

Pantalla principal que carga automáticamente películas populares al iniciar la aplicación. Incluye barra de búsqueda funcional en tiempo real, grid responsivo que adapta de 2 a 5 columnas según el tamaño de pantalla, indicador de carga mientras se obtienen los datos y mensaje de error en caso necesario. Integra el MovieContext para gestión de favoritos.

Favorites.jsx

Pantalla para visualizar películas favoritas. Muestra un mensaje “No tienes favoritos aún” si la lista está vacía. Incluye contador de favoritos, grid responsivo igual al de Home y consumo de MovieContext para agregar o quitar favoritos.

Profile.jsx

Pantalla adicional para mostrar estadísticas de favoritos, información de la aplicación (versión, plataforma, API) y control de tema mediante un switch de modo oscuro/claro. Integra ThemeContext para gestión global del tema.

MovieCard.jsx
Componente para renderizar cada tarjeta de película. Muestra imagen del póster, título, año y rating. Incluye botón de favorito interactivo y overlay con información. Se integra con el tema y permite navegación a la pantalla de detalle (MovieDetail) al hacer clic.

MovieDetail.jsx
Pantalla de detalle de película que muestra imagen ampliada, título, año, rating, sinopsis y estadísticas. Incluye botón de favorito interactivo y botón atrás para regresar. Funciona tanto en web como en móvil y está integrada con Stack Navigator.

MovieContext.jsx
Gestiona globalmente los favoritos. Proporciona funciones toggleFavorite() e isFavorite(). Implementa persistencia con AsyncStorage y expone un custom hook useMovies() para consumir el contexto desde cualquier componente.

ThemeContext.jsx
Gestiona el tema global de la aplicación (oscuro/claro). Incluye funciones toggleTheme() y loadTheme() para cambiar y cargar el tema, persistiendo la preferencia en AsyncStorage. Expone un custom hook useTheme() para acceder al tema desde cualquier componente.

tmdbConfig.js
El archivo tmdbConfig.js contiene la configuración para consumir la API de TMDB. Incluye la API Key, la URL base (https://api.themoviedb.org/3) y funciones auxiliares para construir las URLs de las imágenes y los endpoints de búsqueda y películas populares. Este archivo permite centralizar la configuración de la API y facilita cambios futuros sin modificar la lógica de los componentes.
Archivo: tmdbConfig.js

5. Hooks de React Utilizados

useState

- movies, loading, searchQuery, columns (Home.jsx)
- favorites (MovieContext.jsx)
- isDarkMode (ThemeContext.jsx)

useEffect

- Cargar películas populares
- Cargar y guardar favoritos
- Detectar cambios de tamaño de pantalla
- Cargar y guardar tema oscuro/claro

useContext

- Acceso a favoritos y funciones (useMovies())
- Acceso a tema (useTheme())

createContext

- MovieContext y ThemeContext con custom hooks

6. Problemas Encontrados y Soluciones

1. *Navegación en React Native:*
   React Router no funciona en React Native, por lo que se implementó React Navigation con Bottom Tabs y Stack Navigator, logrando navegación fluida entre pantallas y detalle.
2. *Persistencia local:*
   localStorage no existe en React Native, así que se utilizó AsyncStorage para guardar favoritos y preferencias de usuario de manera persistente entre sesiones.
3. *Interfaz móvil:*
   La NavBar interfería con la barra de estado en iOS y Android. Se redujo su altura a 40px y se colocó dentro de cada pantalla para una interfaz accesible.
4. *Alert no funciona en web:*
   Alert.alert() no mostraba información en la versión web. Se creó la pantalla MovieDetail para mostrar los datos de la película correctamente en web y móvil.
5. *Componente <img> en MovieDetail:*
   El uso de <img> generaba errores en React Native. Se reemplazó por el componente <Image>, permitiendo mostrar correctamente los pósters de las películas.
6. *Responsividad del Grid:*
   El grid de películas no se adaptaba a todos los tamaños de pantalla. Se implementó Dimensions.addEventListener para ajustar dinámicamente las columnas entre 2 y 5.
7. *Sistema de Temas:*
   Se creó un ThemeContext para gestionar modo oscuro y claro, con persistencia en AsyncStorage, permitiendo cambios globales y consistentes en todos los componentes.

7. Conclusiones

Logramos portar la app de web a React Native con éxito, cumpliendo los requisitos básicos y sumando algunas funcionalidades extra. La app se ve bien en cualquier pantalla y la experiencia de usuario mejoró bastante.

Tuvimos que resolver varios problemas, como pasar de localStorage a AsyncStorage, cambiar la navegación de React Router a React Navigation, hacer que el diseño se adaptara a distintos tamaños con Dimensions y ajustar la interfaz para que la barra de estado no molestara en móviles.

Para mejorar en el futuro, se podría agregar paginación de películas, filtros por género, año o rating, pantallas de detalle más completas, reseñas de usuarios, caché de imágenes y un modo offline.