Trabajo Final Mobile 2023

El presente trabajo consiste en realizar un cliente mobile con React Native para la API realizada en trabajos anteriores.
En el trabajo se evaluará tanto el trabajo final presentado como el código escrito y su organización, por lo que se debe prestar atención a las buenas prácticas vigentes.

La aplicación debe tener al menos:

1. Pantalla principal que muestre una especie de dashboard con el resumen del modelo adoptado. Aquí se deberá mostrar información general sobre el dominio y dar una bienvenida al usuario. Puede ser la pantalla de lista del punto 3 si es que el dominio no es lo suficientemente amplio para armar una pantalla principal.
2. Una pantalla para carga de datos con un formulario, donde se pueda agregar datos y llamar al backend para guardarlos.
3. Pantalla con un despliegue de los elementos de su dominio presentado en forma de lista.
4. Pantalla de detalles de un elemento. Al presionar sobre un elemento de la lista se deberá ir a una pantalla con detalles de ese elemento.

No hace falta que le dediquen mucho tiempo al diseño o animaciones. Con que generen sus propios componentes con los elementos core de React Native o directamente la utilización de estos últimos es suficiente. Si se quiere, se pueden utilizar librerías de terceros como Material UI, React Native Elements o similares.

Cosas a tener en cuenta que serán evaluadas:

1) Estructurar el proyecto de manera correcta. Pueden seguir el patrón visto en clase:

```
-Pages
        -–Home
            –-Home.js
            —Styles.css
        -–Details
            –-Details.js
            —Styles.css
 -Components
        -–PrimaryButton
            –-PrimaryButton.js
            —Styles.css
        -–Title
            –-Title.js
            —Styles.css-Routes
```

-Services
—miModelo.service.js
-Contants
—constans.js
—Otros archivos con constantes que se utilizan a lo largo del proyecto

O utilizar algún otro patrón mientras que la estructura logre separar la lógica de los estilos, poder diferenciar una página o una pantalla de un componente y tener los servicios o funciones para conectar la aplicación con la API de manera separada de los elementos visuales.

3) La pantalla de lista debe soportar paginado infinito, es decir, mientras el usuario haga scroll hacia abajo, debe seguir cargando nuevos datos. Si su API no soporta paginado, deberá hacer los cambios necesarios.

4) Tengan en cuenta la reutilización y modularización de código. Si ven que hay código similar dentro de los jsx, significa que deberían crear un componente para poder reutilizarlo.

5) Se puede utilizar cualquier plugin/package/librería que vean útil.

Algunas recomendaciones:

- Utilicen Expo para poder ejecutar y probar las aplicaciones desde sus teléfonos. En su defecto, pueden emular un dispositivo Android con Android Studio.
- La documentación oficial de React Native va a ser su mayor aliado, revisen la api de los componentes propios:
    - https://reactnative.dev/docs/components-and-apis
- Dentro de los componentes propios, presten mayor atención a View, ScrollView, Button, Image, Text, TextInput, TouchableOpacity y FlatList.
- Para navegar entre distintas pantallas hay dos grandes librerías, la que quieran aplicar va a estar bien
    - https://reactnavigation.org/
    - https://v5.reactrouter.com/native/guides/quick-start
- Recuerden los hooks vistos en clase:
    - UseState: https://reactjs.org/docs/hooks-state.html
    - UseEffect: https://reactjs.org/docs/hooks-effect.html

Forma de Entrega

Se debe entregar:

- Un informe completo que incluya una breve explicación de la arquitectura elegida y cómo se relacionan las diferentes partes de su proyecto.
- Código fuente en un repositorio en GitHub con un archivo readme.md con explicaciones de cómo ejecutar y emular el proyecto (que comando se deben ejecutar, qué dependencias se necesitan instalar).
- La aplicación se debe poder emular, sea con expo o Android Studio, para su testeo en la corrección.
- Finalmente se deberá defender frente a la cátedra.