# UNIVERSIDAD DON BOSCO


**FACULTAD DE INGENIERÍA**

**ESCUELA DE COMPUTACIÓN**

CICLO 02-2022

**Asignatura:**

Diseño y Programación de Software Multiplataforma

**Alumnos**

Cesar Antonio Ramírez Barahona RB191879

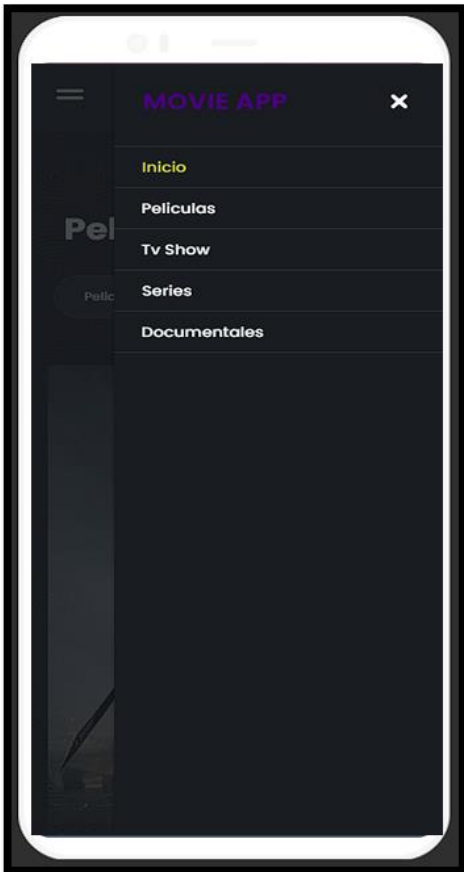
**Segundo desafio practico**

**Fecha de entrega:**

Sábado 22 de octubre, 2022

Ejercicio 1: consta de una aplicación móvil creada para simular una plataforma de streaming de películas.

EN ESTE caso el menú solo funciona la parte de películas, por cuestión de tiempo no programe las otras secciones




```
export default function App(){
  //MENU LATERAL O SIDEBAR MENU
  // <====== SE CREA EVENTO CLICK PARA QUE REDIRIJA A los mudulos ==================>
  // Se asigna id para capturar dato al redirgir
  const Onclick =() =>
  {
    <View >
        <select>
          <option Onclick value="in" id='in'>Inicio </option>
          <option Onclick value="Pe" id='Pe'>Peliculas</option>
          <option Onclick value="Tv" id='TV'>Tv Show</option>
          <option Onclick value="Se" id='SE'>Series</option>
          <option Onclick value="Do" id='Do'>Documentales</option>
        </select>
     </View>
  }
```

- **Función que lleva a los módulos**

```
  //Toma el id y lo returna con la vista del modulo
  function RedirigrModulos (){
    if(ViewComponent.option.id = 'in')
    {
      component={Home}
    }
    if(ViewComponent.option.id = 'Pe')
    {
      component={Peliculas}
    }
    if(ViewComponent.option.id = 'TV')
    {
      component={TvShow}
    }
    if(ViewComponent.option.id = 'SE')
    {
      component={Series}
    }
    if(ViewComponent.option.id = 'Do')
    {
      component={Documetal}
    }
    //Captura si no ha seleccionado una opcion del menu
    // y muestra un mensaje de informacion
    else{
      ViewComponent("No existe esa opcion en el menu")
    }
  }
return(
  <View >
    RedirigrModulos ()
    {
      ViewBase()
    }
  </View>
```

**Vista de película general:**


En este caso utilice un array como método alterno para guardar información de la película y solo mandarlo a llamar en una función y mostrarla al darle click según id

```
//Se crea array que guarda la informacion de las peliculas
//Se crea esta opcion alterna para no utilizar base de datos por el momento
const DATA = [
  {
    id: '1',
    title: 'Free Guy',
    Año: '2022',
    Tiempo: '115 min',
    src:require('../src/img/img1.png'),
  },

  {
    id: '2',
    title: 'Halo',
    Año: '2021',
    Tiempo: '120 min',
    src:require('../src/img/img2.png'),
  },
```

- **Función que Muestra las películas según ID**

```
function MostrarPelicula()
{
  //Se instancia el array de la data
  //Se le pasa como parametro para que tome el id
  //de la pelicula y segun el id muestra la informacion guardada en el array.
  Datos = DATA
  DATA = DataTransfer();
  <View>
    if (DATA.id ='@id')
    {
        ViewComponent.Select("1")
        .DATA.MostrarPelicula(1)
    }
  </View>

}
```

**Vista de película Detalle**


En este caso es la misma función solo muestra la demás información que había sido guardada en el array

- **Función que Muestra las información detallada según ID**

```
function MostrarPeliculaDetalle()
{
  //Basicamente es reutilizar la funcion
  //Agregando la sipnosis de la pelicula,
  //la funcion toma el id y agrega la informacion.
  Datos = DATA
  DATA = DataTransfer();
  <View>
    if (DATA.id ='@id')
    {
        ViewComponent.Select("1")
        .DATA.MostrarPeliculaDetalle(1)

          .TITLE = DATA.TITLE
          .AÑO = DATA.AÑO
          .TIEMPO = DATA.TIEMPO
          .SIPNOSIS = "Un cajero de un banco descubre que en realidad es un personaje"
          -"sin papel dentro de un brutal videojuego de mundo interactivo."

    }
  </View>
```

**Ejercicio 2**

**Wheather APP**


**Instancia de API:**

```
//LLAVE PARA LA API
const API_KEY = "46a9246bebba16d42b36aac3fc3ba8af";

//Se instancia la api openweatherMap para poder usarla
export default function App() {
    const [weatherData, setWeatherData] = useState(null);
    const [loaded, setLoaded] = useState(true);

    async function fetchWeatherData(cityName) {
        setLoaded(false);
        const API = `https://api.openweathermap.org/data/2.5/weather?q=${cityName}&units=metric&appid=${API_KEY}`
        try {
            const response = await fetch(API);
            if(response.status == 200) {
                const data = await response.json();
                setWeatherData(data);
            } else {
                setWeatherData(null);
            }
            setLoaded(true);

        } catch (error) {
            console.log(error);
        }
    }
```

```
//Componente para establecer La ciudad Predeterminada
useEffect(() => {
    fetchWeatherData('Soyapango');
}, [])


if(!loaded) {
    return (
        <View style={styles.container}>
            <ActivityIndicator color='black'  size={36} />
        </View>

    )
}
```


Agregamos componentes como Un searchBar que nos ayuda a tener la ciudad la cual queremos consultar, y mediante la función FecthWeatherApi como parámetro que es parte de Las funciones de las apis, se consulta la info

- Funcion FetchWeatherData

Esta función le manda como parámetro la ciudad a la api y consulta la información, si la ciudad es valida, carga los datos, si no es valida, y la api no devuelve un valor, pues manda un mensaje de advertencia

```
useEffect(() => {
    fetchWeatherData('@CIUDAD');
}, [])


if(!loaded) {
    return (
        <View style={styles.container}>
            <ActivityIndicator color='black'  size={36} />
        </View>

    )
}

else if(weatherData === null) {
    return (
        <View style={styles.container}>
            <SearchBar fetchWeatherData={fetchWeatherData}/>
            <Text style={styles.primaryText}>La ciudad no existe Intenta con otra</Text>
        </View>
    )
}
```

```
export default function SearchBar({ fetchWeatherData }) {

    const [cityName, setCityName] = useState('');

    return (
        <View style={styles.searchBar}>
            <TextInput
                placeholder='Enter City name'
                value={cityName}
                onChangeText={(text) => setCityName(text)}
            />
            <EvilIcons name="search" size={28} color="black"  onPress={() => fetchWeatherData(cityName)}/>
        </View>
    )
}

const styles = StyleSheet.create({
    searchBar: {
        marginTop: 35,
        flexDirection: 'row',
        alignItems: 'center',
        justifyContent: 'space-between',
        width: Dimensions.get('screen').width - 20,
        borderWidth: 1.5,
        paddingVertical: 10,
        borderRadius: 25,
        marginHorizontal: 10,
        paddingHorizontal: 10,
        backgroundColor: 'lightgray',
        borderColor: 'lightgray'
    }
})
```

Posteriormente solo agregue estilo a la searchbar que en ves de que saliera una barra saliera el nombre de la ciudad en grande.