APLICACIONES MÓVILES MULTIPLATAFORMA

LABORATORIO N° 11

Estilos y componentes de UI

Alumno(s):		***Nota***	
Grupo:		***Ciclo:V***	

Criterio de Evaluación	*Excelente (4pts)*	*Bueno (3pts)*	*Requiere mejora (2pts)*	*No acept. (0pts)*	*Puntaje Logrado*
Agregar componentes de estilos					
Reutiliza componentes de UI					
Crea sus propios componentes de UI					
Realiza con éxito lo propuesto en el laboratorio					
Es puntual y redacta el informe adecuadamente					

Laboratorio 11:
Estilos y componentes de UI

Objetivos:

Al finalizar el laboratorio el estudiante será capaz de:

- Entender el funcionamiento de los estilos en React Native
- Desarrollar componentes reutilizables visuales para toda la aplicación
- Importar con éxito una librería creada para un proyecto web y acomodarla al proyecto móvil

Seguridad:

- Ubicar maletines y/o mochilas en el gabinete del aula de Laboratorio.
- No ingresar con líquidos, ni comida al aula de Laboratorio.
- Al culminar la sesión de laboratorio apagar correctamente la computadora y la pantalla, y ordenar las sillas utilizadas.

Equipos y Materiales:

- Una computadora con:
  - Windows 7 o superior
  - VMware Workstation 10+ o VMware Player 7+
  - Conexión a la red del laboratorio
- Máquinas virtuales:
  - Windows 7 Pro 64bits Español - Plantilla
- Instalador de node.js

Procedimiento:

Lab Setup

1. **Configuración de proyecto**
   1.1. Copie el contenido del laboratorio 10 (la clase anterior) a excepción de la carpeta node_modules en una nueva carpeta llamada lab11 y reinstale todas las dependencias:

```
>npm install
```

   1.2. En la nueva carpeta lab11, instalaremos las siguientes dependencias:

```
>npm install --save md5
```

```
>npm install --save react-native-elements
```

2. **Configuración de nuevas vistas**
   2.1. Modificaremos el archivo **App.js** para agregar las nuevas vistas que tendrá nuestro proyecto. Dichas vistas no serán elaboradas en su totalidad en este laboratorio así que deberemos crear las carpetas y los archivos tal cual se indica en la importación realizada.

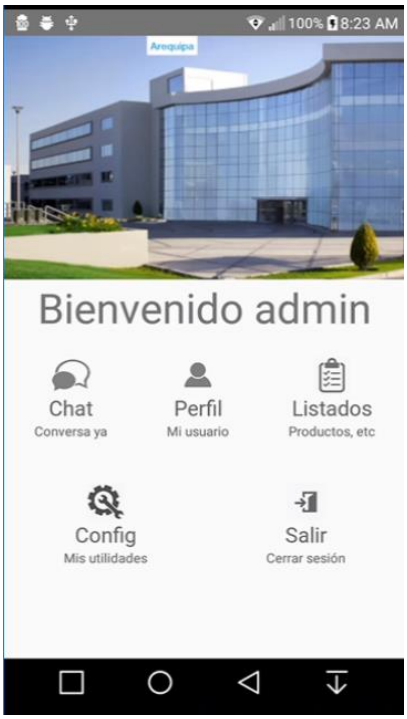
```
import Camera from './src/screens/Camera/Camera';
import Map from './src/screens/Map/Map';
import Profile from './src/screens/Profile/Profile';
import ProfileEdit from './src/screens/Profile/ProfileEdit/ProfileEdit';
import Lists from './src/screens/Lists/Lists';
import Settings from './src/screens/Settings/Settings';

const AppStack = createDrawerNavigator({
    Home: HomeScreen,
    Chat: ChatScreen,
    Location: Location,
    Camera: Camera,
    Map: Map,
    Profile: Profile,
    ProfileEdit: ProfileEdit,
    Lists: Lists,
    Settings: Settings
});
```

2.2. Las vistas antes mencionadas **(Profile, ProfileEdit, Lists, Settings)** deberán tener el siguiente contenido: (solamente es para que la aplicación no falle y lo deje avanzar, luego cada una tendrá un contenido personalizado)

```
import React from 'react';
import { View, Text } from 'react-native';

const Settings = () => {
    return (
        <View>
            <Text>Settings</Text>
        </View>
    );
};

export default Settings;
```

3. Vista de inicio

3.1. La vista de inicio es nuestra bienvenida a nuestros usuarios después de su autenticación. Para nuestro proyecto, trabajaremos con una vista con el siguiente diseño:



3.2. Para lograr este resultado, primero procederemos a crear un archivo **base.js** dentro de una nueva carpeta dentro de **src** que llamaremos **styles**. Este archivo contendrá nuestras variables globales de estilos, para que, en caso cambie la vista de la aplicación, solamente debamos modificar un solo archivo y no todos al mismo tiempo.

```
import { StyleSheet, Dimensions } from 'react-native';

export const dimensions = {
    fullHeight: Dimensions.get('window').height,
    fullWidth: Dimensions.get('window').width
};

export const colors = {
    primary: '#226B74',
    secondary: '#254B5A',
    tertiary: '#5DA6A7'
};

export const padding = {
    sm: 10,
    md: 20,
    lg: 30,
    xl: 40
};

export const fonts = {
    sm: 12,
    md: 18,
    lg: 28,
    xl: 38,
    primary: 'Cochin'
};

export const textColor = '#666';
```

3.3. Ahora crearemos el archivo **global.js** dentro de la misma carpeta **styles**. Aquí crearemos los estilos que reciclaremos a lo largo de la aplicación e invocaremos al archivo **base.js** ya antes creado.

```
import { StyleSheet } from 'react-native';

import { fonts } from './base';

const styles = StyleSheet.create({
    title: {
        fontSize: fonts.xl,
        textAlign: 'center'
    },
    subtitle: {
        fontSize: fonts.lg,
        textAlign: 'center'
    },
    iconContainer: {
        flex: 1,
        flexDirection: 'row',
        justifyContent: 'space-around'
    }
});

export { styles };
```

3.4. Vemos en la pantalla principal unos íconos que representan a las vistas principales. Todas estas imágenes (y las necesarias de este laboratorio) se encuentran en el comprimido **assets.zip** adjunto con este laboratorio. Descárguelo, descomprímalo y pegue todos los archivos dentro de la carpeta **img** que a su vez se encuentra dentro de la carpeta **assets**.

3.5. Así mismo, para no repetir líneas de código, crearemos un componente para los íconos de dicha vista. Crearemos el archivo **IconBox.js** dentro de la carpeta UI ya que será un componente de la Interfaz de Usuario.

```
import React from 'react';
import { StyleSheet, Image, Text, View, TouchableOpacity } from 'react-native';

import { textColor } from '../../styles/base';

const IconBox = props => {
    return (
        <TouchableOpacity onPress={props.onPress}>
            <View style={styles.container}>
                <Image style={styles.icon} source={props.icon} />
                <Text style={styles.value}>{props.value}</Text>
                <Text style={styles.label}>{props.label}</Text>
            </View>
        </TouchableOpacity>
    );
};

const styles = StyleSheet.create({
    container: {
        justifyContent: 'center',
        padding: 20
    },

    icon: {
        flex: 1,
        alignSelf: 'center',
        height: 34,
        width: 34
    },

    value: {
        flex: 1,
        fontSize: 19,
        fontWeight: '200',
        color: textColor,
        textAlign: 'center'
    },

    label: {
        flex: 1,
        fontSize: 12,
        color: textColor,
        textAlign: 'center'
    }
});

export default IconBox;
```

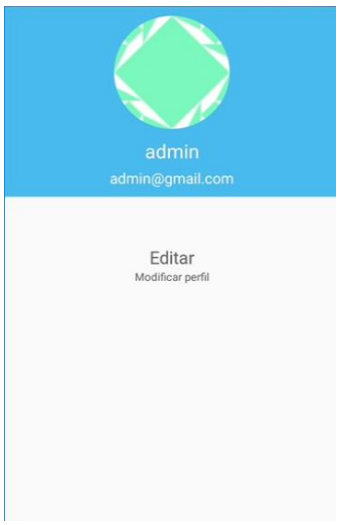
3.6. Una vez creado este archivo, procederemos a modificar **Home.js** con el siguiente contenido.

```
import React from 'react';
import { ScrollView, StyleSheet, View, Image, Text } from 'react-native';
import Ionicons from 'react-native-vector-icons/Ionicons';
import AsyncStorage from '@react-native-community/async-storage';

import IconBox from '../../components/UI/IconBox';

import { styles } from '../../styles/global';

import imgCampus from '../../assets/img/arequipa.jpg';

export default class HomeScreen extends React.Component {
    state = {
        userName: ''
    };
    static navigationOptions = {
        title: 'Bienvenido a la App!',
        tabBarIcon: ({ focused, horizontal, tintColor }) => {
            return <Ionicons name="ios-clipboard" size={25} color={tintColor} />;
        }
    };
    componentDidMount = async () => {
        const userName = await AsyncStorage.getItem('userName');
        this.setState({ userName: userName });
    };
    go2Chat = () => {
        this.props.navigation.navigate('Chat');
    };
    go2Profile = () => {
        this.props.navigation.navigate('Profile');
    };
    go2Lists = () => {
        this.props.navigation.navigate('Lists');
    };
    go2Settings = () => {
        this.props.navigation.navigate('Settings');
    };
    logoutHandler = async () => {
        await AsyncStorage.clear();
        this.props.navigation.navigate('Auth');
    };
    render() {
        return (
            <ScrollView>
                <Image source={imgCampus} />
                <Text style={styles.title}>Bienvenido {this.state.userName}</Text>
                <View style={styles.iconContainer}>
                    <IconBox
                        value={'Chat'}
                        label={'Conversa ya'}
                        icon={require('../../assets/img/icon-chat.png')}
                        onPress={this.go2Chat}
                    />
                    <IconBox
                        value={'Perfil'}
                        label={'Mi usuario'}
                        icon={require('../../assets/img/icon-profile.png')}
                        onPress={this.go2Profile}
                    />
```

```
                    <IconBox
                        value={'Listados'}
                        label={'Productos, etc'}
                        icon={require('../../assets/img/icon-list.png')}
                        onPress={this.go2Lists}
                    />
                </View>
                <View style={styles.iconContainer}>
                    <IconBox
                        value={'Config'}
                        label={'Mis utilidades'}
                        icon={require('../../assets/img/icon-settings.png')}
                        onPress={this.go2Settings}
                    />
                    <IconBox
                        value={'Salir'}
                        label={'Cerrar sesión'}
                        icon={require('../../assets/img/icon-logout.png')}
                        onPress={this.logoutHandler}
                    />
                </View>
            </ScrollView>
        );
    }
}
```

4. Vista de Perfil de Usuario

4.1. Deseamos lograr una vista parecida a la siguiente:



4.2. Vemos que está usando el avatar generado en los laboratorios de **Desarrollo de Aplicaciones Web Avanzado**, así que copiaremos el archivo **avatar.js** del proyecto **dawa_api** y lo copiaremos dentro de la carpeta **lib**

```
import md5 from 'md5';

const getAvatar = email => {
    return `https://www.gravatar.com/avatar/${md5(email)}?d=identicon`;
};

export default getAvatar;
```

4.3. Así mismo, la portada de dicha vista nos puede ser útil por ejemplo para cuando queramos mostrar los detalles de otro usuario, así que reciclaremos eso en un componente llamado **Badge.js** dentro de **UI**

```
import React from 'react';
import { Text, View, Image, StyleSheet } from 'react-native';

const Badge = props => {
    return (
        <View style={styles.container}>
            <Image style={styles.image} source={{ uri: props.avatar }} />
            <Text style={styles.name}> {props.userName} </Text>
            <Text style={styles.handle}> {props.email} </Text>
        </View>
    );
};
var styles = StyleSheet.create({
    container: {
        backgroundColor: '#48BBEC',
        paddingBottom: 10
    },
    name: {
        alignSelf: 'center',
        fontSize: 21,
        marginTop: 10,
        marginBottom: 5,
        color: 'white'
    },
    handle: {
        alignSelf: 'center',
        fontSize: 16,
        color: 'white'
    },
    image: {
        height: 125,
        width: 125,
        borderRadius: 65,
        marginTop: 10,
        alignSelf: 'center'
    }
});

export default Badge;
```

4.4. Pero para usar el avatar correctamente necesitaremos el email del usuario que inició sesión. Modificaremos SignIn.js para que podamos almacenar este dato.

```
        this.setState({ loading: true });
        axios({
            method: 'POST',
            url: 'api/user/signin',
            data: {
                username: this.state.user,
                password: this.state.password
            }
        })
            .then(async response => {
                await AsyncStorage.setItem('userId', response.data.data._id);
                await AsyncStorage.setItem('userName', response.data.data.username);
                await AsyncStorage.setItem('userEmail', response.data.data.email);
                await AsyncStorage.setItem('userToken', response.data.token);
                ToastAndroid.showWithGravity(
                    response.data.message,
                    ToastAndroid.LONG,
                    ToastAndroid.TOP
                );
                this.props.navigation.navigate('App');
            })
```

4.5. Modificaremos ahora el contenido de Profile.js

```
import React, { Component } from 'react';
import { ScrollView, Image, View } from 'react-native';
import AsyncStorage from '@react-native-community/async-storage';

import { styles } from '../../styles/global';

import getAvatar from '../../lib/avatar';

import IconBox from '../../components/UI/IconBox';
import Badge from '../../components/UI/Badge';
import loadingGif from '../../assets/img/loading.gif';

class Profile extends Component {
    state = {
        avatar: '',
        userName: '',
        userEmail: ''
    };
    componentDidMount = async () => {
        const userName = await AsyncStorage.getItem('userName');
        const userEmail = await AsyncStorage.getItem('userEmail');
        this.setState({
            userName: userName,
            userEmail: userEmail,
            avatar: getAvatar(userEmail)
        });
    };
    editProfileHandler = () => {
        this.props.navigation.navigate('ProfileEdit');
    };
    render() {
        return (
            <ScrollView>
                {this.state.avatar !== '' ? (
                    <Badge
                        userName={this.state.userName}
                        email={this.state.userEmail}
                        avatar={this.state.avatar}
                    />
                ) : (
                    <Image source={loadingGif} />
                )}
                <View style={styles.iconContainer}>
                    <IconBox
                        value={'Editar'}
                        label={'Modificar perfil'}
                        icon={require('../../assets/img/icon-edit.png')}
                        onPress={this.editProfileHandler}
                    />
                </View>
            </ScrollView>
        );
    }
}

export default Profile;
```

4.6. Así mismo, y para terminar de retocar esta parte, modificaremos **ProfileEdit.js**

```
import React, { Component } from 'react';
import { ScrollView, Text, StyleSheet } from 'react-native';
import { Input, Button } from 'react-native-elements';
import AsyncStorage from '@react-native-community/async-storage';

import { styles } from '../../../styles/global';

class ProfileEdit extends Component {
    state = {
        userId: null,
        userName: '',
        userEmail: '',
        picture: ''
    };
    static navigationOptions = {
        drawerLabel: () => null
    };
    componentDidMount = async () => {
        const userName = await AsyncStorage.getItem('userName');
        const userEmail = await AsyncStorage.getItem('userEmail');
        this.setState({
            userName: userName,
            userEmail: userEmail
        });
    };
    inputHandler = (text, field) => {
        this.setState({ [field]: text });
    };
    render() {
        return (
            <ScrollView style={formStyles.container}>
                <Text style={styles.subtitle}>Editar perfil</Text>
                <Input
                    placeholder="Nombre de usuario"
                    leftIcon={{ type: 'font-awesome', name: 'user' }}
                    inputContainerStyle={formStyles.input}
                    value={this.state.userName}
                    onChangeText={text => this.inputHandler(text, 'userName')}
                />
                <Input
                    placeholder="Correo electrónico"
                    leftIcon={{ type: 'font-awesome', name: 'envelope' }}
                    inputContainerStyle={formStyles.input}
                    value={this.state.userEmail}
                    onChangeText={text => this.inputHandler(text, 'userEmail')}
                />
                <Button title="Guardar Perfil" containerStyle={formStyles.button} />
            </ScrollView>
        );
    }
}

const formStyles = StyleSheet.create({
    container: {
        padding: 10
    },
    input: {
        marginTop: 10
    },
    button: {
        marginTop: 10
    }
});

export default ProfileEdit;
```

5. **Finalizar la sesión**
   5.1. Apagar el equipo virtual
   5.2. Apagar el equipo

Conclusiones:

Indicar las conclusiones que llegó después de los temas tratados de manera práctica en este laboratorio.