

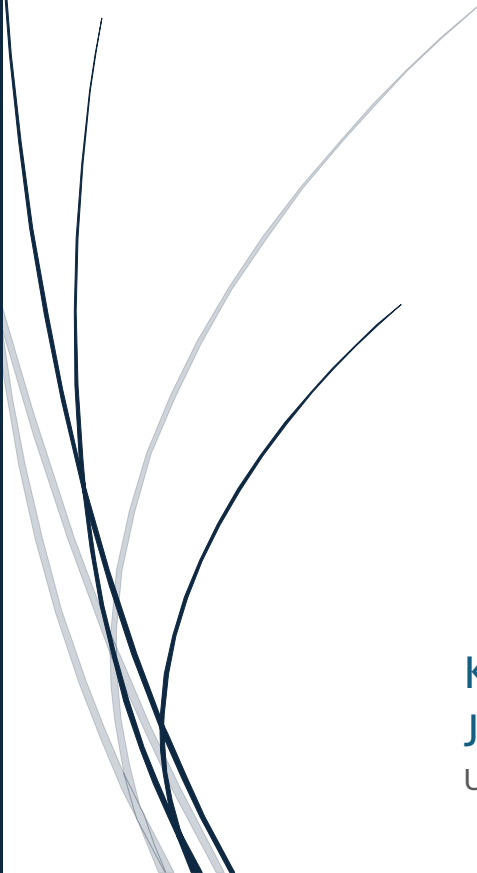


10-9-2024

Desafío Practico 2

DPS - GL01

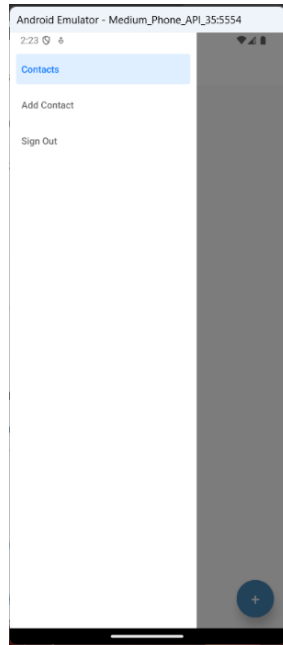
Ing. Karens Medrano



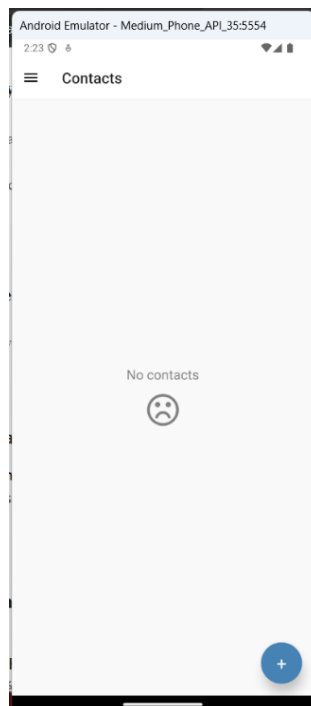
Karla Elizabeth Murillo Urrutia
Javier Ernesto Pérez Joaquín
UNIVERSIDAD DON BOSCO

Ejercicio 1: Diseñar una aplicación móvil para ayudarte a gestionar contactos de manera eficiente y recordar los cumpleaños importantes de tus amigos, familiares y colegas

- Menú Drawer que se puede acceder desde el lado izquierdo de la pantalla el cual nos permite acceder a Registrar Contacto y Cerrar sesión. (0.5%)



- Botón flotante circular agregar (0.5%) y pantalla de información de no hay registros. (0.5%)



- Almacenamiento de datos con AsyncStorage (10%)

Para el registro de usuarios

```

1 import React, { useState } from 'react';
2 import { View, TextInput, Button, Alert, StyleSheet, Text } from 'react-native';
3 import AsyncStorage from '@react-native-async-storage/async-storage';
4
5 const RegisterScreen = ({ navigation }) => {
6   const [user, setUser] = useState('');
7   const [email, setEmail] = useState('');
8   const [password, setPassword] = useState('');
9
10  const handleRegister = async () => {
11    if (!user || !email || !password) {
12      Alert.alert('Error', 'Imcompleted');
13    }
14    return;
15
16    try {
17      await AsyncStorage.setItem('user', user);
18      await AsyncStorage.setItem('email', email);
19      await AsyncStorage.setItem('password', password);
20

```

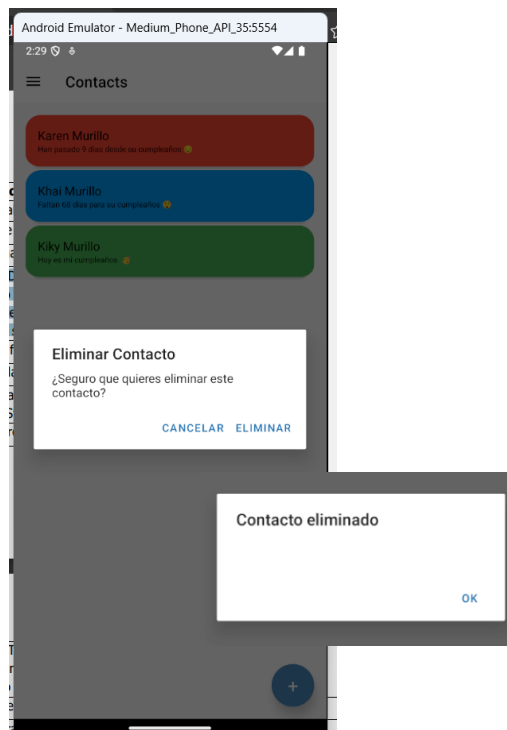
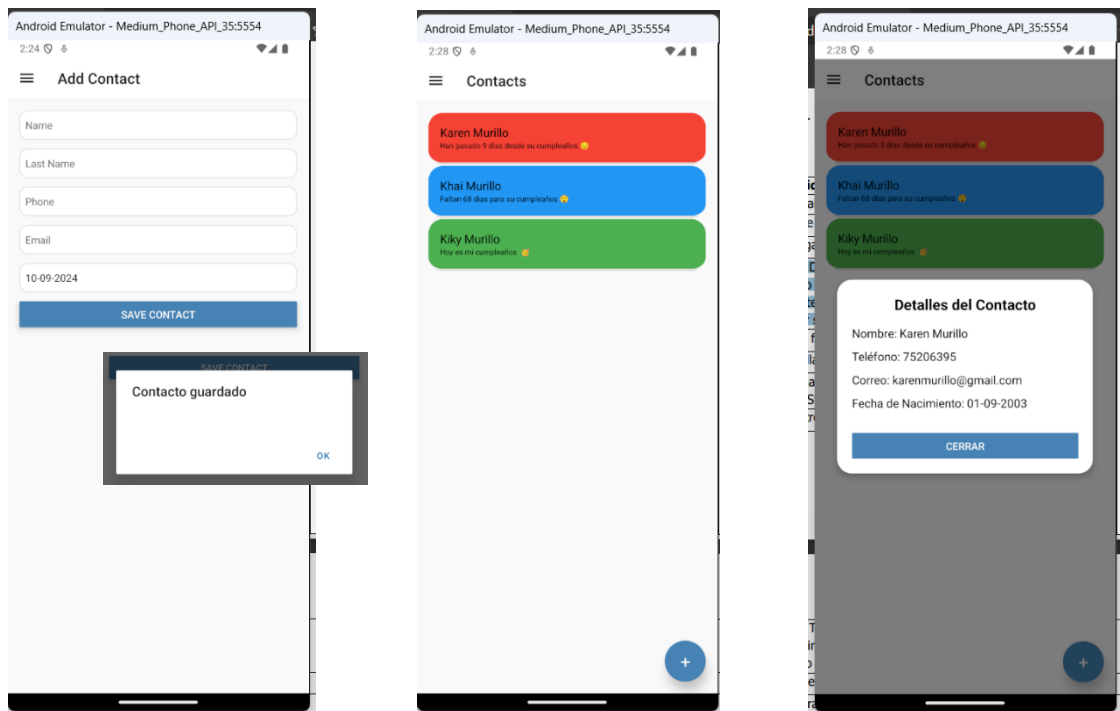
Para el registro de contactos

```

1 import React, { useState } from 'react';
2 import { View, TextInput, Button, Alert, StyleSheet, Platform } from 'react-native';
3 import AsyncStorage from '@react-native-async-storage/async-storage';
4 import DateTimePicker from '@react-native-community/datetimepicker';
5 import moment from 'moment';
6
7 const AddContactScreen = ({ navigation }) => {
8   const [name, setName] = useState('');
9   const [lastname, setLastname] = useState('');
10  const [phone, setPhone] = useState('');
11  const [email, setEmail] = useState('');
12  const [birthday, setBirthday] = useState(new Date());
13  const [showPicker, setShowPicker] = useState(false);
14
48 const handleSaveContact = async () => {
49   if (!validateFields()) return;
50
51   const newContact = { ...
52   };
53
54   // Verificar si el contacto ya existe
55   let contacts = await AsyncStorage.getItem('contacts');
56   contacts = contacts ? JSON.parse(contacts) : [];
57
58   const contactExists = contacts.some(contact => contact.phone === newContact.phone);
59
60   if (contactExists) { ...
61   }
62
63   // Si el contacto no existe, agregarlo
64   contacts.push(newContact);
65   await AsyncStorage.setItem('contacts', JSON.stringify(contacts));
66

```

- Registro y eliminación de contactos (10%)



- Pantalla de Inicio de Sesión y Registro

Android Emulator - Medium_Phone_API_35:5554
2:39

Login

Log In

Karla Murillo

.....

LOG IN

Create an account
[Sign In](#)

Murillo Mueillo Murillo I

q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123 , .

Android Emulator - Medium_Phone_API_35:5554
2:39

Login

Log In

karla.murillo@gmail.com

.....

LOG IN

Create an account
[Sign In](#)

Android Emulator - Medium_Phone_API_35:5554
2:42

← Sign In

Sign In

Julio Murillo

julio@gmail.com

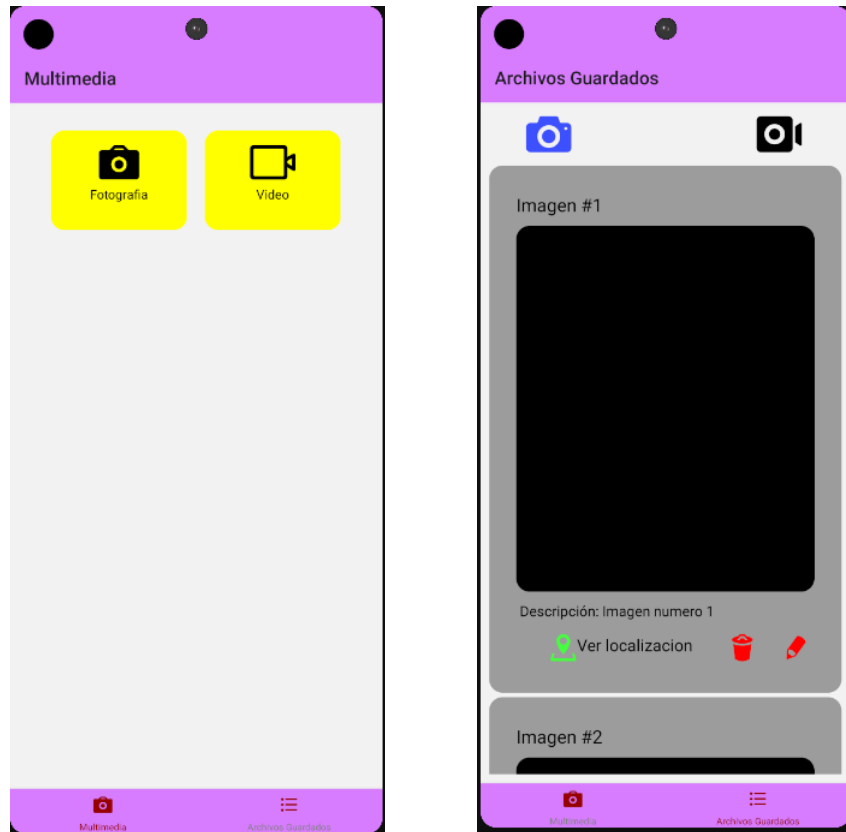
.....

SIGN IN

1 2 3 4 5 6 7 8 9 0
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123 , .

Ejercicio 2: Desarrollar una aplicación móvil MemoriesApp que permita a los usuarios capturar imágenes y videos, agregar anotaciones(texto) sobre estas imágenes y videos, y almacenar la ubicación geográfica (coordenadas GPS) donde se capturaron.

- Menú Tab que se puede acceder desde la parte inferior de la pantalla que nos da acceso a 2 categorías solicitadas



- Almacenamiento de datos con AsyncStorage

```
const saveVideo = async () => {
  if (video) {
    try {
      const record = await MediaLibrary.createAssetAsync(video.uri);
      alert("Video guardado con exito");

      const newVideo = {
        token: record.uri,
        comm: text,
      };

      let videos = await AsyncStorage.getItem("Video");

      if (videos != null || videos != undefined) {
        const updateVideos = JSON.parse(videos);
        updateVideos.push(newVideo);
        await AsyncStorage.setItem("Video", JSON.stringify(updateVideos));
      } else {
        const array = [newVideo];
        await AsyncStorage.setItem("Video", JSON.stringify(array));
      }

      setText(null);
      setVideo(null);
    } catch (e) {
      console.log("Multimedia", e);
    }
  }
};
```

```
const saveVideo = async () => {
  if (video) {
    try {
      const record = await MediaLibrary.createAssetAsync(video.uri);
      alert("Video guardado con exito");

      const latitude = location.coords.latitude;
      const longitude = location.coords.longitude;

      const newVideo = {
        token: record.uri,
        comm: text,
        latitude: latitude,
        longitude: longitude
      };

      let videos = await AsyncStorage.getItem("Video");

      if (videos != null || videos != undefined) {
        const updateVideos = JSON.parse(videos);
        updateVideos.push(newVideo);
        await AsyncStorage.setItem("Video", JSON.stringify(updateVideos));
      } else {
        const array = [newVideo];
        await AsyncStorage.setItem("Video", JSON.stringify(array));
      }

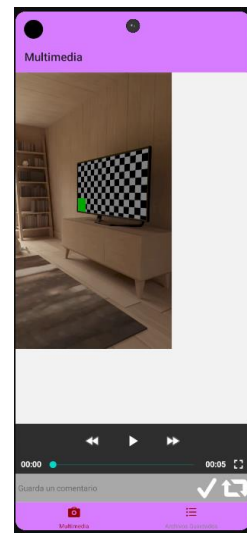
      setText(null);
      setVideo(null);
    } catch (e) {
      console.log("Multimedia", e);
    }
  }
};
```

- Captura de imagen o video

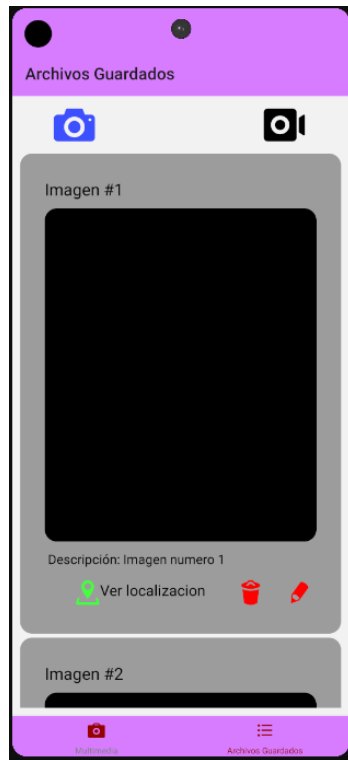
- Captura de imágenes



- Captura de Video



- Cada item de la lista presenta información más detallada (descripción de la imagen o video)



- Geocodificación del lugar donde se realizó la fotografía o video

