|  |  |  |
| --- | --- | --- |
| Asignatura: *Desarrollo de Aplicaciones Móviles Avanzadas* | Día y hora: 30/10/2024 | Grupo: 01 |
| Docente: Ing. Benjamín Magaña | **E-mail:** [**ia.torrescallejas2002@ufg.edu.sv**](mailto:ia.torrescallejas2002@ufg.edu.sv) | **Aula:** |
| Estudiante:  Torres Callejas | **Michael Fernano** | **Firma Michael** |

Apellidos

Nombres

Firma

|  |  |  |  |
| --- | --- | --- | --- |
| **Carrera: Ingenieria en Desarrollo de Software** | **Carné: TC100220** | **Fecha: 30/10/2024** | **Calificación:** |

**Contexto del Ejercicio**:

Una empresa de turismo ecológico ha solicitado una aplicación que permita a sus guías capturar fotos y grabar videos de los recorridos y experiencias de los visitantes en tiempo real. La aplicación debe ser fácil de usar, permitiendo a los guías capturar momentos únicos rápidamente y con opciones de personalización. También debe incluir controles avanzados para mejorar la calidad de las imágenes y videos en entornos naturales de baja luz o con mucha vegetación.

**Instrucciones**: Realice una aplicación móvil que cumpla con los requerimientos a continuación, integrando captura de fotos, grabación de video y control de la cámara. Asegúrate de seguir las indicaciones de funcionalidad y diseño solicitadas.

Este examen es individual, y deberá entregar el código fuente, capturas de pantalla y un video explicativo de la implementación .

**Aplicación de Cámara para Turismo Ecológico**

**Objetivo**: Diseñar una aplicación que permita capturar fotos, grabar videos y ajustar configuraciones de cámara para mejorar la calidad de la captura en entornos naturales.

**Requerimientos**:

1. **Captura de Foto**:
   * Implemente un botón de captura que permita tomar fotos al instante, almacenándolas en una carpeta específica de la app.
   * Las fotos capturadas deben mostrarse en una galería dentro de la aplicación, permitiendo a los guías revisarlas rápidamente.
   * Incluya una opción para agregar un filtro automático que mejore los colores naturales (opcional para estudiantes más avanzados).
2. **Grabación de Video**:
   * La aplicación debe permitir grabar videos cortos con un botón de grabación que se puede iniciar y detener con facilidad.
   * Guarde los videos en la misma galería de la aplicación y muestra un mensaje de confirmación al guardar.
   * Implemente una función de reproducción dentro de la app para revisar los videos grabados.
3. **Controles de Cámara**:
   * Incluya controles ajustables de **zoom** y **enfoque** que el usuario pueda modificar fácilmente desde la interfaz.
   * Agregue un botón para alternar entre cámara frontal y trasera.
   * Añade una opción de **flash** para tomar fotos y grabar videos en áreas con baja luz natural.
4. **Interfaz de Usuario**:
   * Diseñe una interfaz intuitiva para que los guías puedan cambiar entre foto, video y configuraciones de cámara rápidamente.
   * Los botones y controles deben estar claramente identificados y ser accesibles con una sola mano.
   * La aplicación debe mostrar un mensaje de éxito o error en cada operación (captura, grabación o cambio de configuración).

**Evaluación**:

* **Funcionalidad Completa**: Cada una de las funciones (captura de foto, grabación de video, controles de cámara) debe estar correctamente implementada y ser funcional.
* **Interfaz de Usuario**: La interfaz debe ser intuitiva y fácil de usar en un entorno de campo (textos claros, botones de buen tamaño, etc.).
* **Código**: Organización del código, claridad y uso de buenas prácticas.
* **Documentación**: Explicación breve del código y de cada función implementada, detallando las bibliotecas utilizadas y las decisiones de diseño.

**Criterios de Calificación**:

* Captura de Foto y Galería: 25%
* Grabación de Video y Reproducción: 25%
* Controles de Cámara: 25%
* Diseño de Interfaz y Documentación: 25%

# Documentación de la Aplicación: Galería y Visualización de Videos

Hecho por: Michael Fernando Torres Callejas, TC100220.  
Actividad: Parcial 03

## Introducción

Este documento tiene como objetivo explicar detalladamente el código de la aplicación Android que implementa una galería con capacidades de captura de fotos y videos, así como la visualización de estos elementos desde una pantalla de galería. La aplicación está desarrollada con Kotlin y utiliza Jetpack Compose para la creación de la interfaz de usuario. Se hace uso de diferentes bibliotecas y componentes modernos de Android, como CameraX y ExoPlayer, que nos permiten lograr una experiencia de usuario fluida y rica visualmente.

## Estructura del Proyecto

El proyecto se compone principalmente de varios archivos fuente importantes:  
- `MainActivity.kt`: Actividad principal que contiene la configuración de la navegación entre diferentes pantallas.  
- `MainMenuScreen.kt`: Pantalla de menú principal, que permite al usuario elegir entre abrir la cámara o la galería.  
- `CameraView.kt`: Pantalla que permite capturar fotos y grabar videos con la cámara del dispositivo.  
- `GalleryScreen.kt`: Pantalla donde se muestra la galería de fotos y videos capturados.  
- `VideoPlayerScreen.kt`: Pantalla que permite la reproducción de videos seleccionados de la galería.  
- `Utils.kt`: Archivo que contiene funciones auxiliares que se usan en varios puntos del código.

## Detalles de las Bibliotecas Utilizadas

- Jetpack Compose: Se usa para crear interfaces declarativas en la aplicación. Facilita la construcción de interfaces de usuario más intuitivas y menos propensas a errores, ya que la mayoría del trabajo se realiza declarativamente.  
- CameraX: Permite capturar fotos y videos de una forma moderna y eficiente, proporcionando una API más sencilla de usar que la API tradicional de la cámara de Android.  
- ExoPlayer: Se utiliza para la reproducción de videos. Ofrece una solución poderosa y optimizada para la visualización de contenido multimedia en Android.  
- ThumbnailUtils: Es una herramienta para crear miniaturas de videos, lo cual mejora la experiencia del usuario permitiendo una previsualización de los videos antes de reproducirlos.

## MainActivity.kt

Esta es la actividad principal de la aplicación y se encarga de gestionar la navegación entre las diferentes pantallas usando NavHost de Jetpack Compose.

val navController = rememberNavController()  
NavHost(  
 navController = navController,  
 startDestination = "main\_menu",  
 modifier = Modifier.padding(innerPadding)  
) {  
 composable("main\_menu") { MainMenuScreen(navController) }  
 composable("camera\_view") { CameraView(navController) }  
 composable("gallery") { GalleryScreen(navController) }  
 composable("video\_player/{videoUri}", arguments = listOf(navArgument("videoUri") { type = NavType.StringType })) { backStackEntry ->  
 val encodedUri = backStackEntry.arguments?.getString("videoUri")  
 val videoUri = encodedUri?.let { Uri.decode(it) }  
 videoUri?.let { VideoPlayerScreen(it) }  
 }  
}

## MainMenuScreen.kt

El menú principal tiene dos botones que permiten al usuario navegar a la cámara o a la galería. Es una pantalla sencilla que proporciona una experiencia intuitiva para el usuario.

Button(  
 onClick = {  
 navController.navigate("camera\_view")  
 },  
 modifier = Modifier  
 .fillMaxWidth()  
 .padding(vertical = 8.dp)  
) {  
 Text("Abrir Cámara")  
}

## CameraView.kt

Esta pantalla permite al usuario capturar fotos y videos. Utiliza CameraX para interactuar con el hardware de la cámara y proporcionar una experiencia de captura moderna.  
  
- Zoom y Flash: Se implementaron controles adicionales para permitir zoom y flash tanto en la cámara trasera como en la frontal. Esto ayuda a capturar mejores fotos en diferentes condiciones.

val zoomRatio = remember { mutableStateOf(1f) }  
Slider(  
 value = zoomRatio.value,  
 onValueChange = {  
 zoomRatio.value = it  
 camera?.cameraControl?.setZoomRatio(it)  
 },  
 valueRange = minZoom..maxZoom,  
 modifier = Modifier  
 .fillMaxHeight()  
 .padding(vertical = 32.dp)  
 .rotate(-90f)  
)

## GalleryScreen.kt

Esta pantalla muestra una lista de fotos y videos almacenados en el dispositivo. Se hace uso de LazyColumn para mostrar los elementos de la galería de manera eficiente.  
  
- Diferenciación Visual entre Videos e Imágenes: Para mejorar la experiencia del usuario, se ha agregado una superposición con la etiqueta "Video" a cada miniatura de video. Esto permite al usuario distinguir fácilmente entre videos e imágenes.

Box(  
 modifier = Modifier  
 .fillMaxWidth()  
 .height(200.dp)  
 .padding(4.dp)  
 .clickable {  
 val encodedUri = Uri.encode(file.absolutePath)  
 navController.navigate("video\_player/$encodedUri")  
 }  
) {  
 if (thumbnail != null) {  
 Image(  
 bitmap = thumbnail,  
 contentDescription = null,  
 modifier = Modifier.fillMaxSize(),  
 contentScale = ContentScale.Crop  
 )  
 }  
 Box(  
 modifier = Modifier  
 .fillMaxWidth()  
 .background(Color.Black.copy(alpha = 0.6f))  
 .align(Alignment.BottomCenter)  
 .padding(4.dp)  
 ) {  
 Text(  
 text = "Video",  
 color = Color.White,  
 fontWeight = FontWeight.Bold,  
 textAlign = TextAlign.Center,  
 modifier = Modifier.align(Alignment.Center)  
 )  
 }  
}

## VideoPlayerScreen.kt

En esta pantalla se usa ExoPlayer para reproducir videos seleccionados de la galería.

val exoPlayer = remember {  
 ExoPlayer.Builder(context).build().apply {  
 val uri = Uri.parse(videoUri)  
 setMediaItem(MediaItem.fromUri(uri))  
 prepare()  
 playWhenReady = true  
 }  
}

## Decisiones de Diseño

- Jetpack Compose se utilizó debido a su naturaleza declarativa, lo cual facilita la creación de interfaces de usuario complejas y hace el código más legible y mantenible.  
- CameraX fue elegido para gestionar la cámara porque simplifica muchas de las complejidades asociadas con el uso directo de la API de cámara en Android, permitiendo acceder a funciones avanzadas con facilidad.  
- ExoPlayer se usó para la reproducción de videos porque es altamente personalizable y soporta una amplia gama de formatos de video, lo cual mejora la flexibilidad de la aplicación.  
- La superposición con la etiqueta "Video" en la galería fue una decisión centrada en la experiencia del usuario, para ayudar a diferenciar claramente los elementos de la galería y hacer la interfaz más intuitiva.

## Conclusión

La aplicación proporciona una interfaz intuitiva y moderna para capturar fotos, grabar videos y ver los elementos almacenados en una galería interna. Las decisiones de diseño tomadas, como el uso de Jetpack Compose y CameraX, facilitaron la creación de una experiencia de usuario robusta y fácil de usar. Además, se agregaron detalles visuales para mejorar la usabilidad, como la indicación clara de cuáles elementos son videos y cuáles son imágenes. La combinación de tecnologías modernas y diseño centrado en el usuario hace que esta aplicación sea adecuada para el entorno de turismo ecológico en el que se utiliza.