



Proyecto en equipo

Diego Díaz 166129

Jorge Luis Tovar Arriaga 174829

Pedro Escobedo Straffon 176110

Desarrollo de Aplicaciones Móviles

Funcionalidades clave

Creemos que las funcionalidades mas importantes son las de como la aplicación al iniciarla detecta si es la primera vez que el usuario ingresa, si existe una sesión iniciada o el rol del usuario que esta iniciando sesión para direccionarlo a esa vista. Para lograr esto fue que hicimos un nav host para así navegar por la app, ya que esta hecha en Jetpack Compose así es mas fácil navegar, todo eso esta en esta sección:

```
NavHost(  
    navController = navController,  
    startDestination = if (isLoggedIn) "home" else "login"  
) {  
    composable("login") {  
        if (isSignUpScreen) {  
            SignUpScreen(authManager) {  
                isLoggedIn = true  
                isRoleLoaded = false // 🔥 Se recarga el rol al registrarse  
            }  
        } else {  
            LoginScreen(authManager, {  
                isLoggedIn = true  
                isRoleLoaded = false // 🔥 Se recarga el rol al iniciar sesión  
            }, {  
                isSignUpScreen = true  
            })  
        }  
    }  
  
    composable("home") {  
        when (userRole) {  
            "Administrador" -> VistaAdmin(navController, onLogout)  
            "Estudiante" -> VistaEstudiante(navController, onLogout)  
            "Profesor" -> VistaProfesor(navController, onLogout)  
            else -> Text("Error: Rol no reconocido", color = Color.Red)  
        }  
    }  
}
```

De igual forma las funciones que utilizamos en comunicación con firebase para acceder a los datos de los usuarios existentes y del usuario que esta usando la aplicación al momento nos permiten tener una comunicación clara con la base de datos, creando y modificando los documentos necesarios para un correcto funcionamiento de la

aplicación, una de las funcionalidades clave fue la de crear una materia ya que las materias están acomodadas en una colección llamada materias y el segmento de código en el que se crea es este

```
data class Course(  
    val nombre: String,  
    val descripcion: String,  
    val horario: String,  
    val fechaCreacion: Timestamp = Timestamp.now(),  
    var id: String = ""  
)  
  
fun createCourse(db: FirebaseFirestore, course: Course, onSuccess: () -> Unit, onError: (Exception) -> Unit) {  
    db.collection("cursos").document(course.nombre)  
        .set(course)  
        .addOnSuccessListener {  
            Log.d("Firebase", "Curso creado con éxito: ${course.nombre}")  
            onSuccess()  
        }  
        .addOnFailureListener { e ->  
            Log.e("Firebase", "Error al crear curso", e)  
            onError(e)  
        }  
}
```

Otras funcionalidades son:

```
db.collection("cursos").document(materia)  
    .collection("estudiantes").document(uid).get()
```

Cuando se escanea un código QR, se extrae el identificador único del usuario (uid) y el nombre de la materia. A partir de esta información, Firestore consulta el documento correspondiente al estudiante dentro de la subcolección estudiantes, ubicada dentro del documento de la materia en la colección cursos:

```
val referenciaAsistencia = db.collection("asistencias")
    .document(materia)
    .collection(fechaActual)
    .document(uid)
```

La estructura de almacenamiento en Firestore es dinámica y jerárquica. La colección asistencias contiene un documento por cada materia. A su vez, cada materia contiene una subcolección nombrada con la fecha del día en formato dd_MM_yyyy. Dentro de esta subcolección, cada documento representa a un estudiante y almacena la hora exacta de llegada:

```
val asistencia = hashMapOf(
    "uid" to uid,
    "horaLlegada" to horaActual
)

referenciaAsistencia.set(asistencia)
```

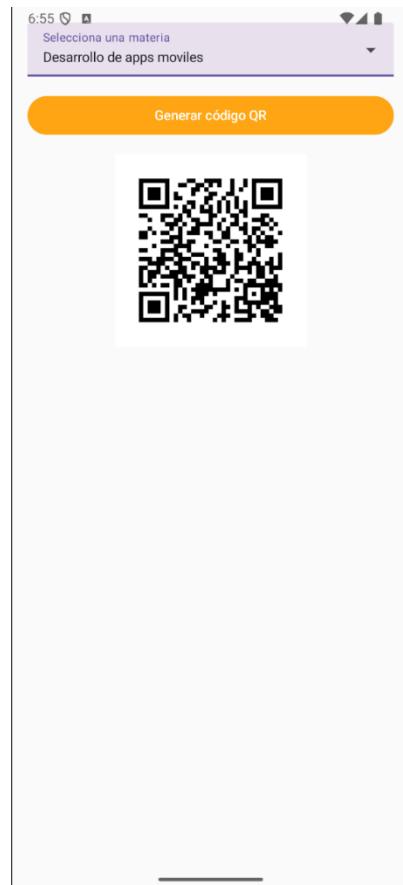
Cuando se confirma que un estudiante no ha registrado su asistencia, se procede a crear un nuevo documento en Firestore con su uid y la hora de llegada

Capturas de pantalla

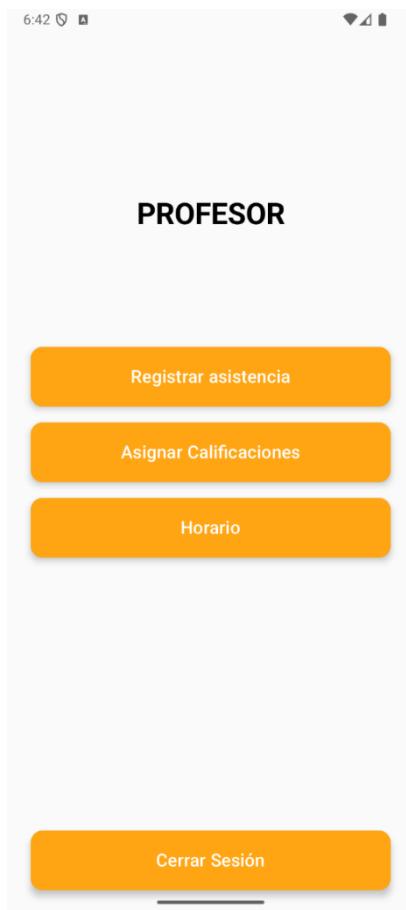
Menú de estudiante, consiste en 4 botones 3 funciones básicas como tomar asistencia en donde se genera el QR que el profesor después tendrá que escanear, ver horario en donde podrá ver las materias que tiene inscritas el estudiante junto con la hora en donde tiene el curso y finalmente la sección de calificaciones en donde el estudiante podrá consultar la calificación final ponderada de cada uno de sus cursos; finalmente tenemos el cuarto botón encargado de cerrar sesión.



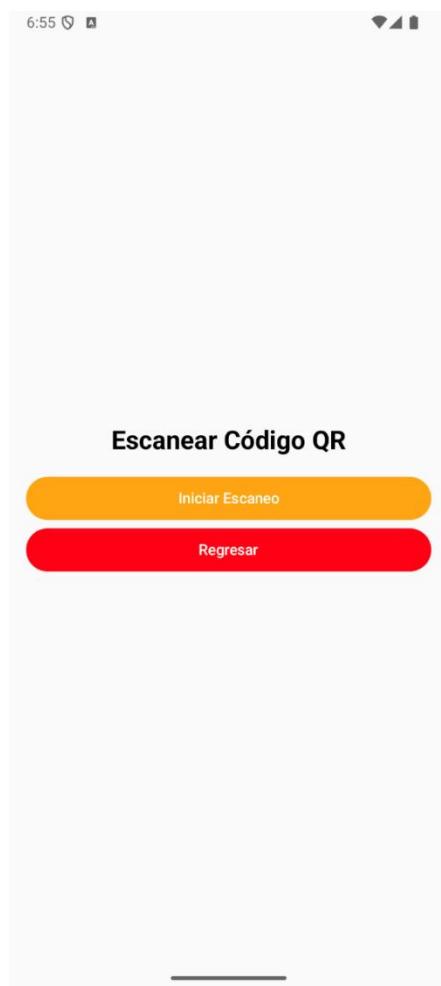
Esta es la vista de Tomar asistencia. Aquí, se presenta un spinner con diversas opciones, consistiendo en todos los cursos disponibles en los que el estudiante está inscrito. Al seleccionar uno de los cursos, se presiona el botón de "generar código QR", el cual arroja un código QR personalizado para que el profesor pueda escanear y tomar la asistencia del alumno.



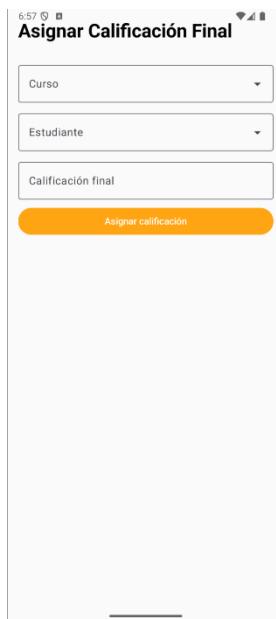
Este es el menú de profesor, en donde al igual que estudiante tiene 4 botones en donde 3 son encargados de hacer funciones básicas de la aplicación, en registrar asistencia se le pide un permiso especial al usuario para acceder a la cámara para que así, el profesor pueda escanear el código QR de cada uno de sus alumnos, asignar calificaciones, que como su nombre lo dice el profesor podrá asignar una calificación a cualquiera de las materias que tenga al estudiante que desee, el ultimo botón de función es el de horario que es lo mismo que el de estudiante, muestra todas las clases que el profesor tiene asignadas junto con su hora y finalmente el botón de cerrar sesión como lo tiene el estudiante.



Al presionar el botón de "Registrar asistencia", navegamos a la siguiente pantalla. En esta, se nos presentan dos botones: "Iniciar Escaneo" y "Regresar". El primero, nos envía a la pantalla de escaneo de QR; que debe ser apuntada al código QR que genere el alumno para registrar su asistencia correctamente. Por otra parte, el botón de "Regresar" simplemente nos hace volver a la pantalla anterior.



La siguiente función de profesor es la de asignar calificación final, aquí el profesor tiene 2 spinners uno con los cursos que el profesor imparte y el otro para ver los estudiantes que hay en ese curso, finalmente hay un campo para que el profesor ponga una calificación.



6:57 6:57 Asignar Calificación Final

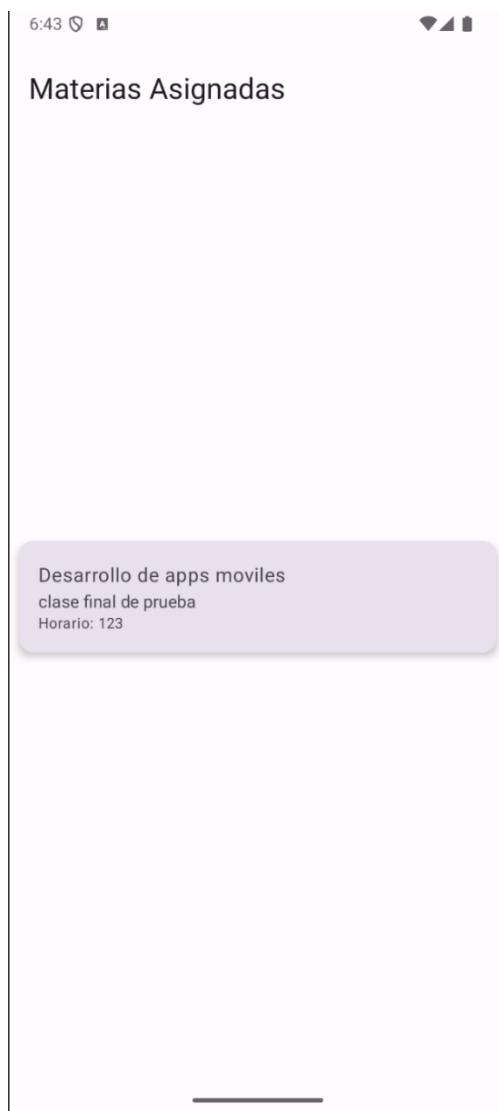
Curso

Estudiante

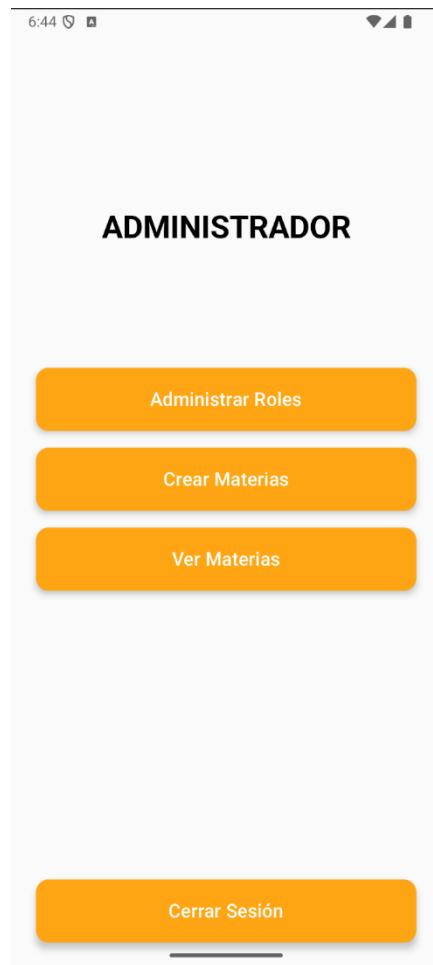
Calificación final

Asignar calificación

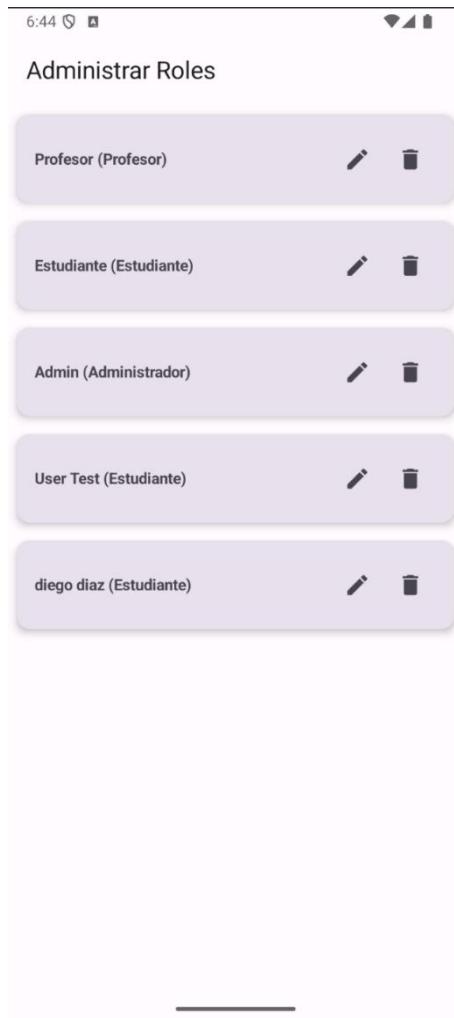
Finalmente tenemos la última vista en la que el profesor puede ver todos los cursos que tiene asignados



Este es el menú de administrador, se sigue respetando el mismo formato de los otros menús, 3 botones al centro con funciones principales y el ultimo de cerrar sesión, los 3 botones de función son los siguientes, Administrar roles en donde se muestran todos los usuarios existentes en la base da datos y el administrador puede modificar sus roles y hasta borrarlos, luego crear materias en donde justamente el administrador es capaz de crear nuevos cursos con estudiantes y profesores asociados y finalmente ver materias en donde el administrador puede ver todas las materias que ha creado para llevar un control.



Esta es la vista de administrar roles, como se observa hay unos usuarios de prueba en donde esta su nombre con su rol en el siguiente formato: *nombre(rol)* y junto dos botones, uno con forma de lápiz que es para editar el rol que tiene ese usuario en caso de que algún estudiante se haya puesto como profesor o algo por ese estilo y junto un bote de basura que es para eliminar a la persona en caso de que se haya dado de baja o simplemente sean usuarios basura, pretendemos que esta vista sea útil para el usuario ya que puede llevar el control de las personas inscritas dentro de la institución y pueda consultar el nombre de los estudiantes y los profesores para la vista de Crear materias.



Aquí está la vista de crear materias en la que el administrador debe asignar un nombre del curso, descripción que puede ser una clave del curso o una breve descripción de lo que se vera en el curso y el ultimo campo de la primera sección es el horario en la que el administrador debe escribir el horario en el que se impartirá la clase, este debe de estar en el siguiente formato Dia1 y Dia2 (HH:MM – HH:MM) y para finalizar esta sección de creación está el botón de registrar el curso.

Para la segunda parte de la sección tenemos la parte de asignación tanto de profesores como de estudiante, aquí el administrador debe de escribir el nombre completo de los estudiantes y el profesor que deberán tomar e impartir la materia, este proceso debe de ser uno a uno.

6:44 64% 64%

Crear Nueva Materia

Nombre del curso

Descripción

Horario

Registrar Curso

Agregar Estudiante

Nombre del estudiante

Agregar

Asignar Profesor

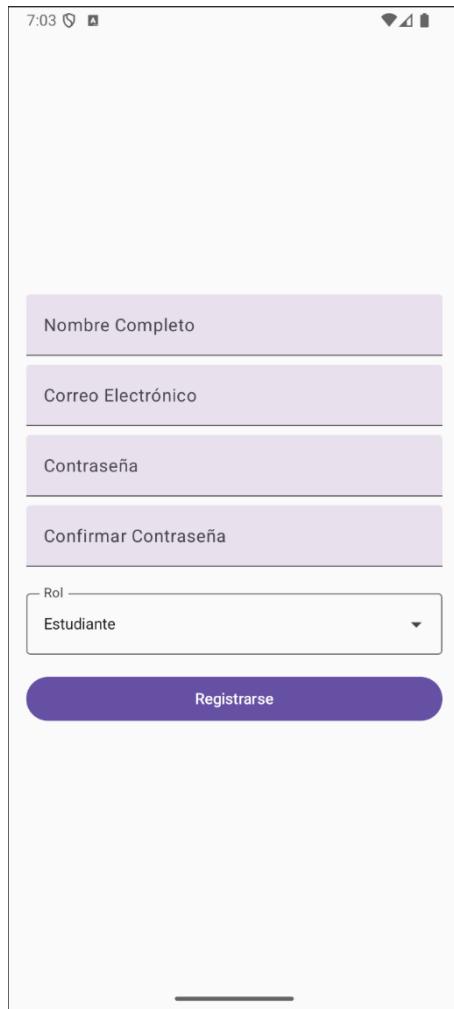
Nombre del profesor

Asignar

Finalmente, la última vista es para que el administrador vea los cursos que ha creado, en estos se ve el nombre del curso su descripción y el horario; pretendemos que esta vista sea usada por el administrador para que lleve un control de materias y siempre tenga a la mano los cursos que ha creado.



Ahora tenemos la pantalla para dar de alta una cuenta nueva en la plataforma, aquí lo más relevante es que la contraseña debe de ser de mínimo 6 caracteres y que el usuario puede elegir el rol para así poder tener acceso a las funciones correspondientes



Esta es la primera pantalla que ve el usuario al iniciar la aplicación es la pantalla de inicio de sesión en donde tiene que ingresar su contraseña y en caso de que no tenga cuenta se tenga que registrar

