

# **PROYECTO BUSSINESONE**

**ERP MOVIL**

Presentado por Jorge José Dumitrache Chust

# Índice

- Introducción a BussinesOne
- Análisis del problema
- Estudio de mercado
- Propuesta de proyecto
- Diseño de la aplicación
- Desarrollo
- Conclusiones
- Bibliografía

# Introducción a BussinesOne

## Que es ?

Bussines One es un ERP diseñado para dispositivos móviles en el que los usuario pueden gestionar los recursos y actividades de sus empresa desde la facilidad de sus dispositivos móviles.

Como todo ERP posee un sistema modular al que se le pueden añadir funcionalidades extra a la aplicación mediante la adición de nuevos modulos.



# Introducción a BussinesOne

## Objetivos

### General

El objetivo general del proyecto es el desarrollo de una app que garantiza a empleados y empleadores herramientas eficientes y fáciles de usar para la gestión y uso de los recursos de una empresa.

### Específicos

- Implementar una estructura modular que permita añadir nuevos módulos .
- Garantizar un rendimiento en la aplicación eficiente.
- Implementar módulos básicos para la gestión de recursos de una empresa
- Manejar los datos mediante un controlador de base de datos.
- Permitir compatibilidad en 2 lenguajes Android y Kotlin Compose para mayor facilidad de desarrollar nuevos módulos

# Introducción a BussinesOne

## Justificación y motivación

Los ERP están orientados en su mayoría a grandes empresas y los complejos sistemas que tienen estas, esta situación resulta en que las pymes y pequeños comercios no tienen suficiente capacidad como para requerir uno de estos inmensos y complicados ERP para gestionar la cantidad de recursos que poseen. El motivo principal de Bussines One es ofrecer a estos pequeños comercios las herramientas necesarias para gestionar sus negocios.



# Introducción a BussinesOne

## Publico objetivo

### Pymes

Son las pymes (Pequeñas y medianas empresas) las que menos facilidades tienen a la hora de gestionar sus recursos debido a la diferencia de dificultad que si que tienen las grandes empresas, lo cual no quita que este asunto les siga siendo un inconveniente.

Pequeños negocios tradicionales como lo son cafeterías, bares, quioscos o peluquerías se beneficiarían mucho por la necesidades que tienen de tener una buena gestión, un control eficiente, un registro...

# Análisis del problema

## Necesidades

- El proyecto necesita abarcar a la mayor cantidad posible de usuarios y como Android tiene una mayor cuota de mercado el desarrollo de la aplicación será en ese sistema operativo
- Gestionar los datos de los usuarios y los módulos.
- Ofrecer una estructura modular y que soporte varios lenguajes de programación.
- La aplicación debe ser accesible siempre.

## Limitaciones

- Al estar desarrollado en Android los usuarios de otros sistemas operativos no podrán acceder a la aplicación.
- Al soportar dos lenguajes (Java y Kotlin Compose) todos los recursos deben ser accesibles mediante los dos lenguajes

# Análisis del problema

## Retos técnicos

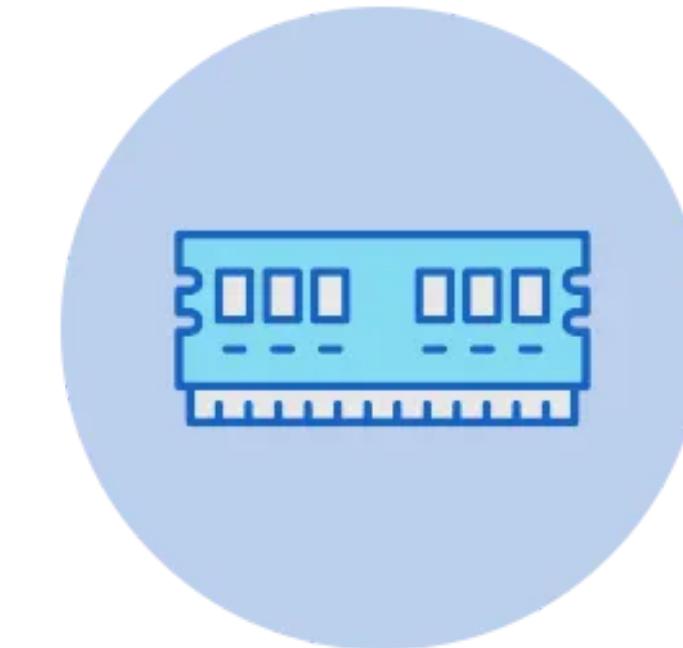
**Compatibilidad CameraX**



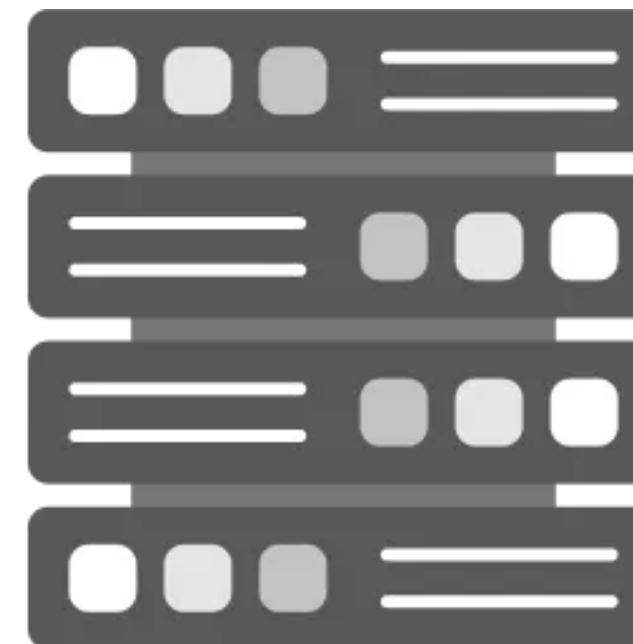
**Sincronización de datos**



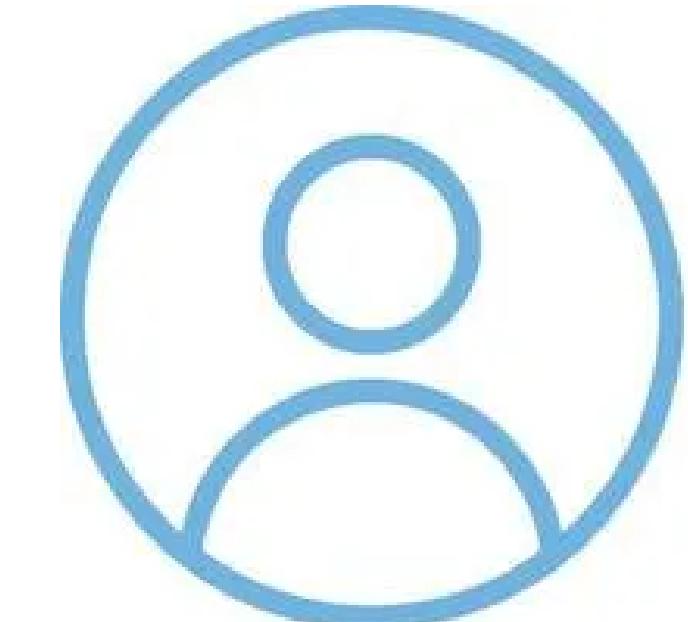
**Manejo de imágenes y memoria**



**Gestión de llamadas http**



**Gestión de usuarios**



# Estudio de mercado

## Competencia

**Odoo**



**ERPNext**



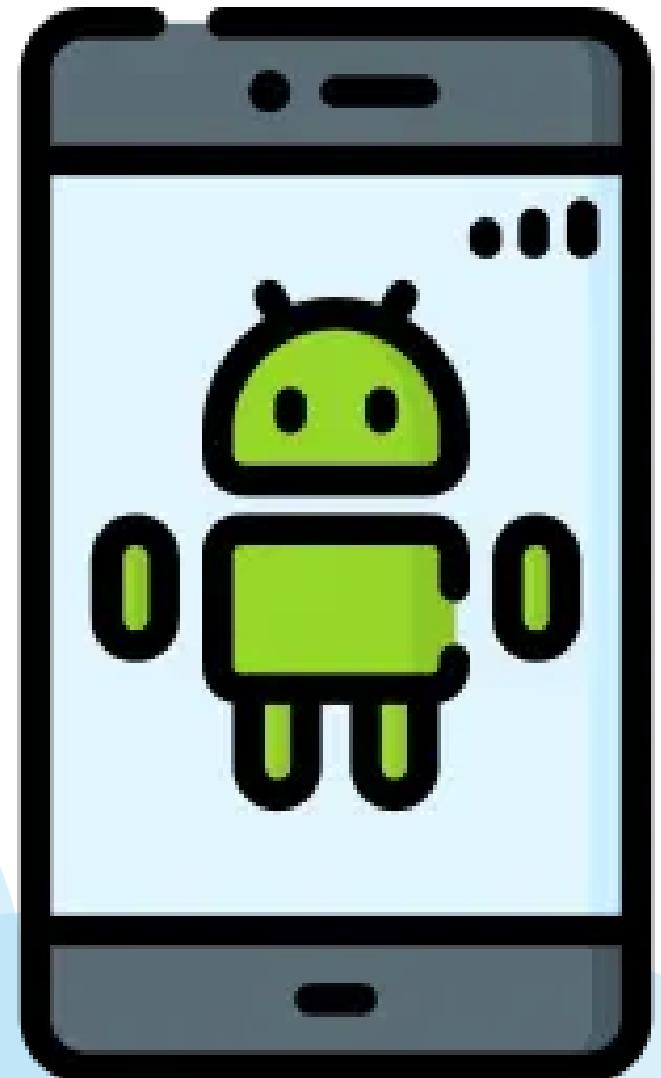
**Dolibarr**



# Estudio de mercado

## Factores diferenciadores

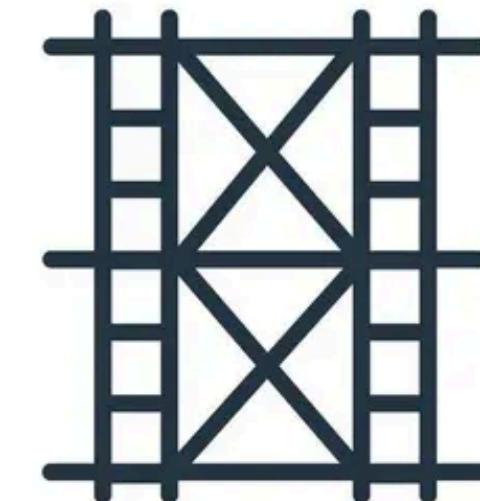
**App nativa**



**Arranque rápido**



**Arquitectura ligera**



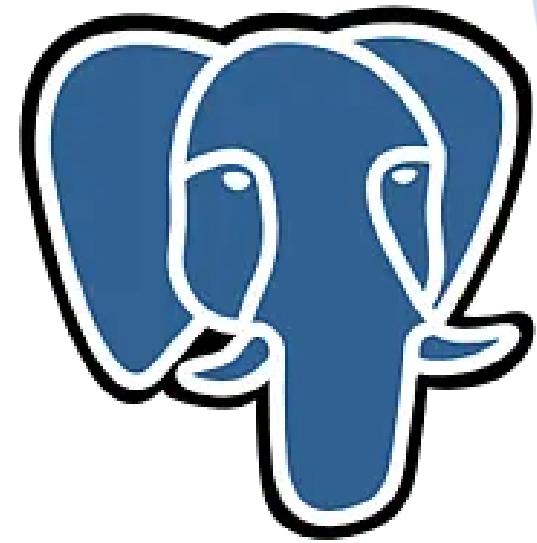
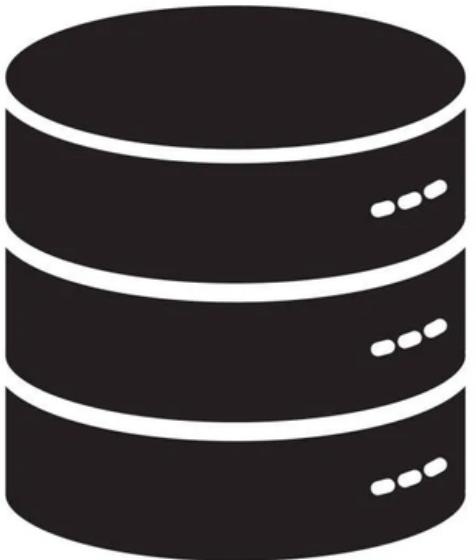
# Propuesta de proyecto

**Arquitectura del sistema**

*Kotlin Compose*

**Retrofit**

**Model–View–ViewModel**

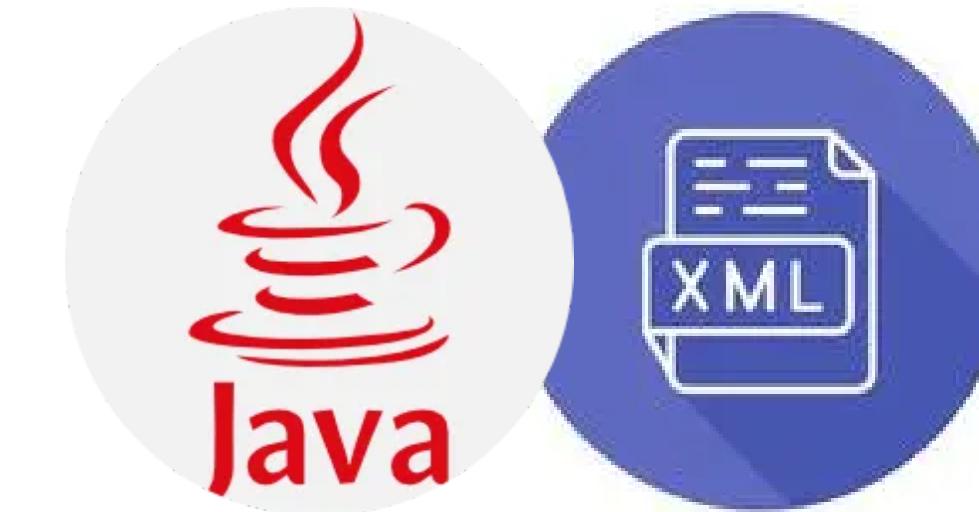


Postgre**SQL**

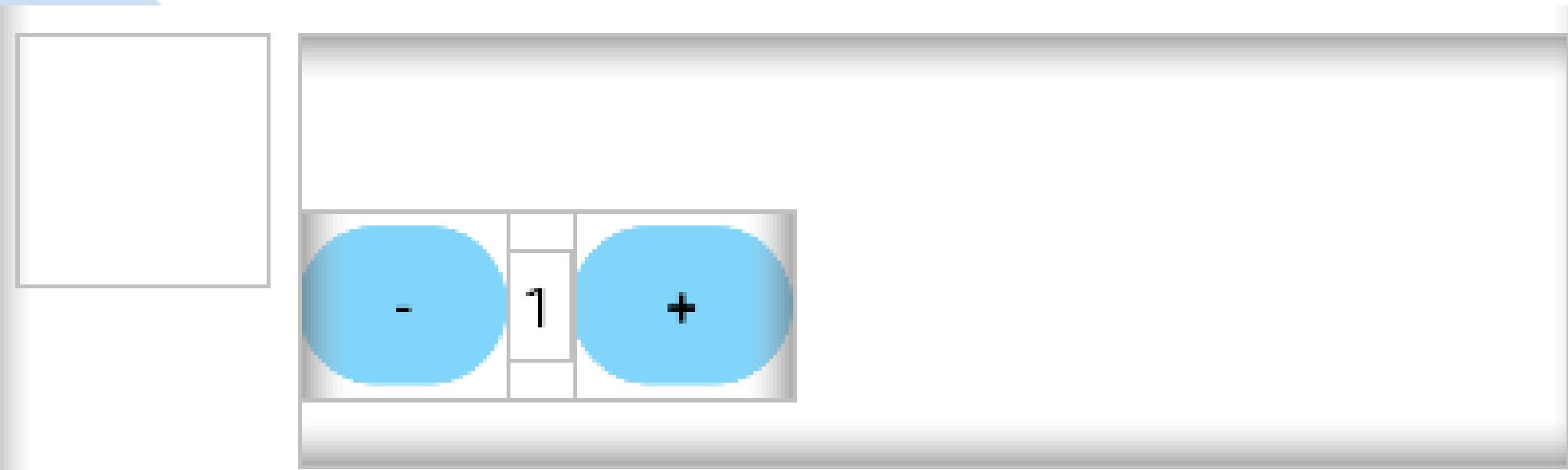
# Propuesta de proyecto

## Arquitectura del sistema

*Kotlin Compose*



**RecyclerView**



**Scaffold**

Buscar módulos

Facturación Electrónica  
Facturación

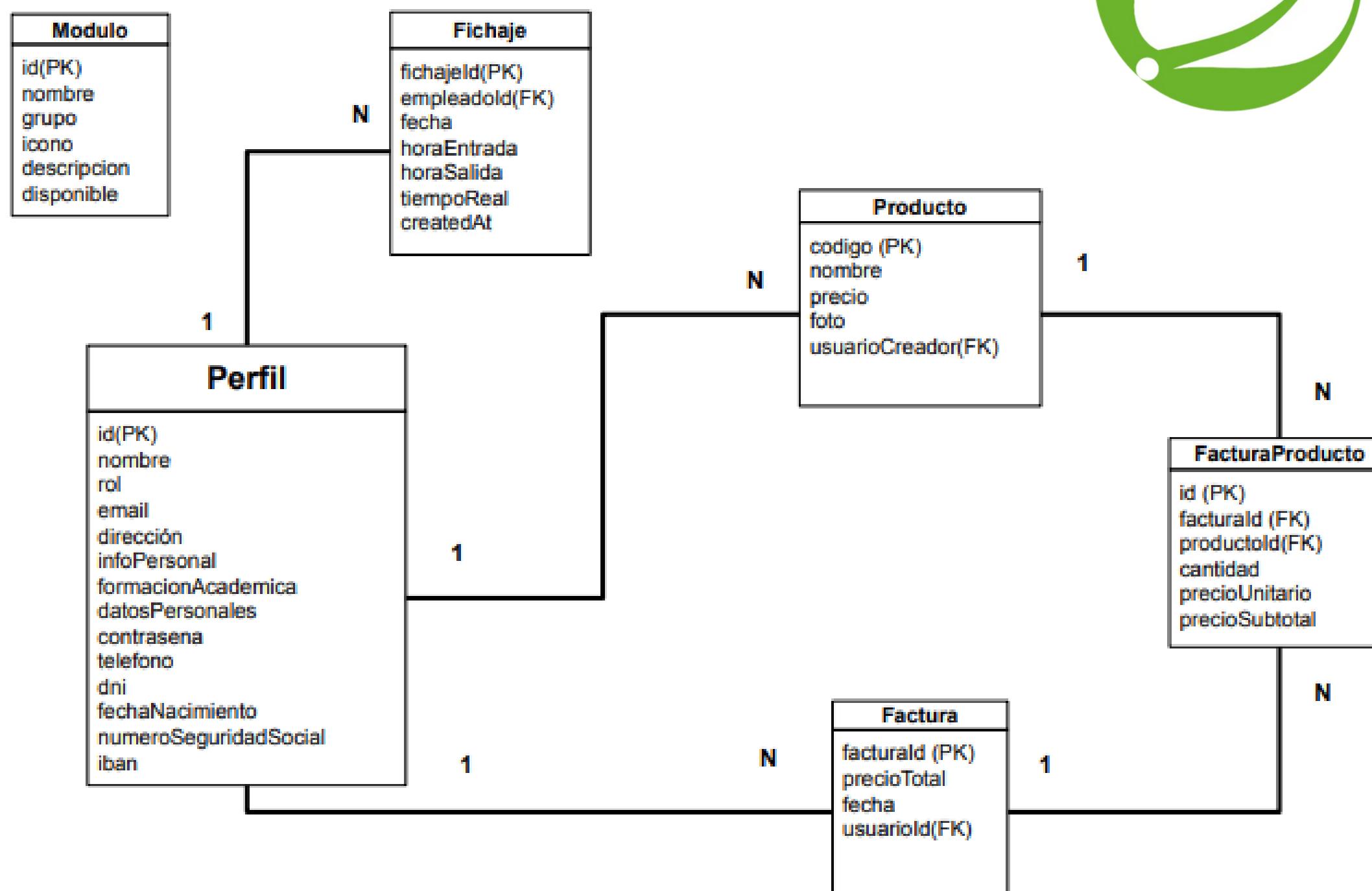
Gestión de Inventario  
Inventario

CRM de Clientes  
Clientes

Análisis de Ventas  
Ventas

# Propuesta de proyecto

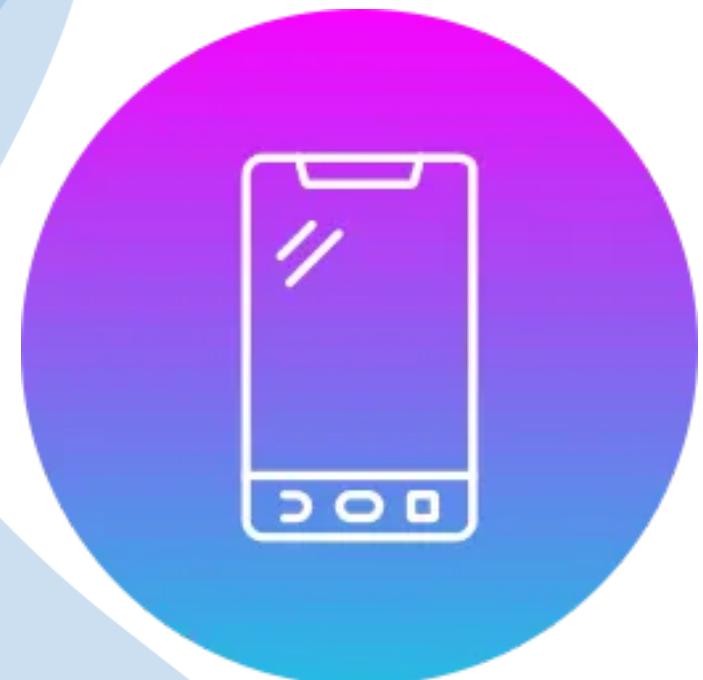
## Arquitectura del sistema



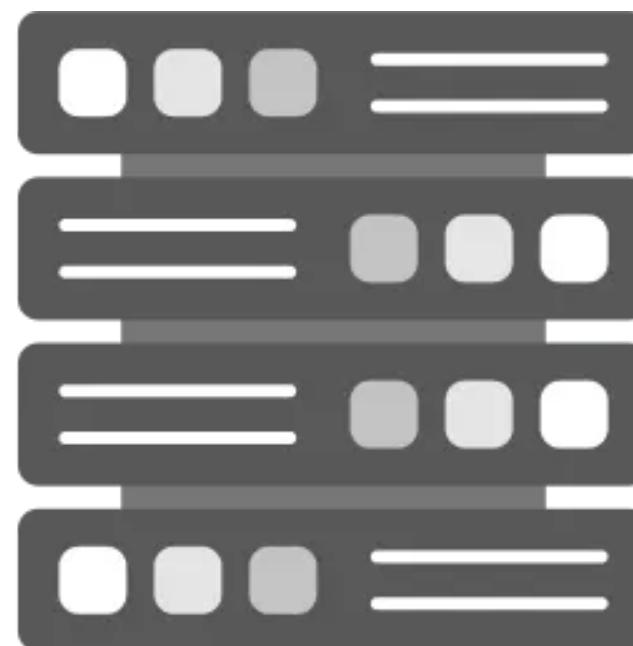
# Propuesta de proyecto

**Arquitectura del sistema - Comunicación entre componentes**

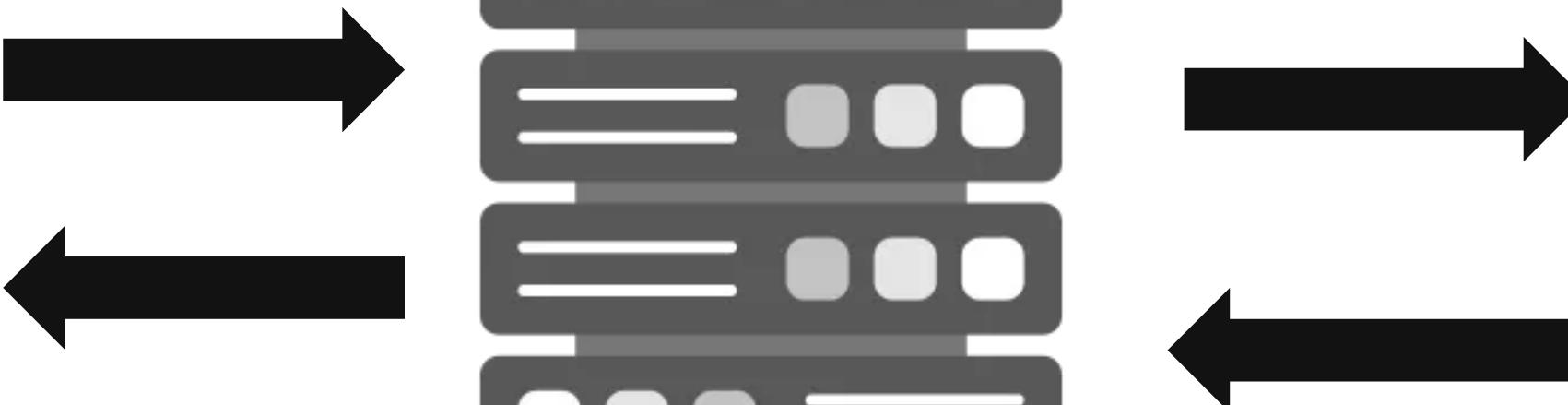
**App**



**Server**



**Database**



# Diseño de la aplicacion

## Paleta de colores

### Colores primarios

#1E88E5

#81D4FA

#1565C0

### Colores de fondo

#F5F5F5

#121212

# Diseño de la aplicación



**Usuario**

Correo electrónico

**Contraseña**

Contraseña

Selección de un rol

**Iniciar sesión**

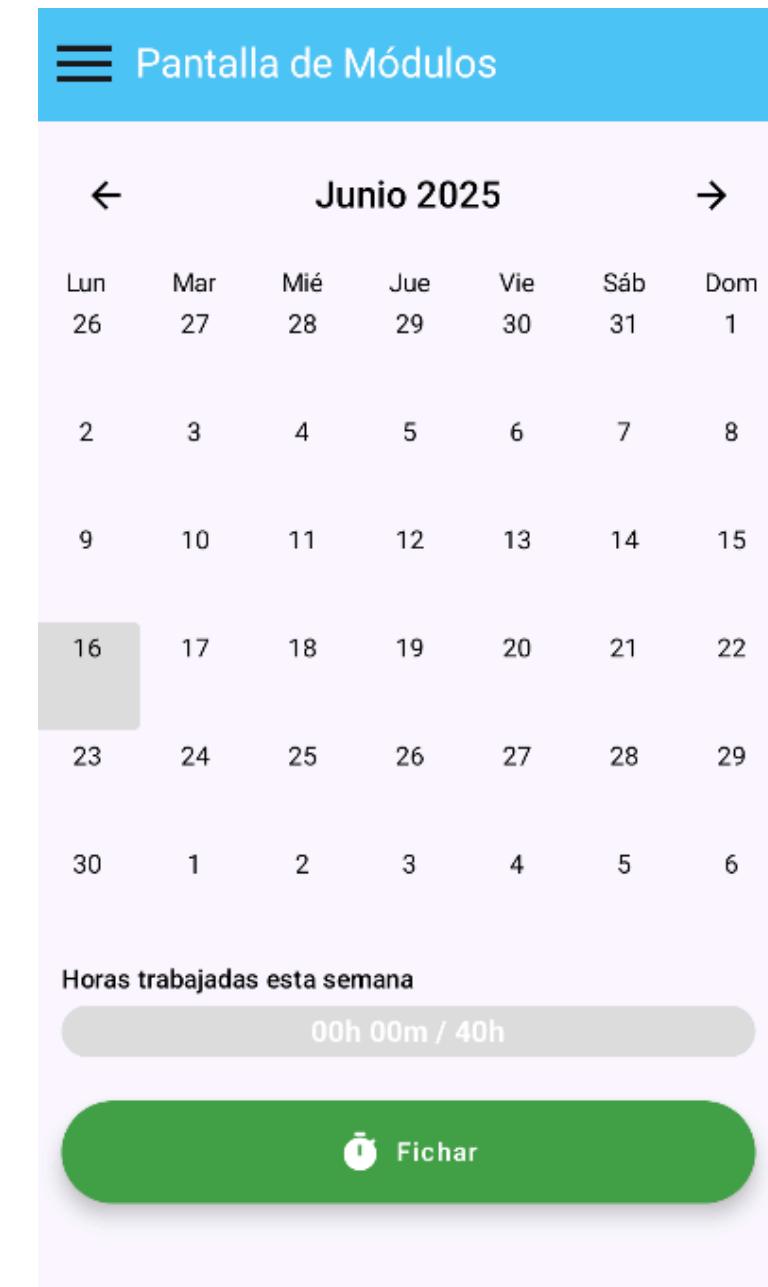
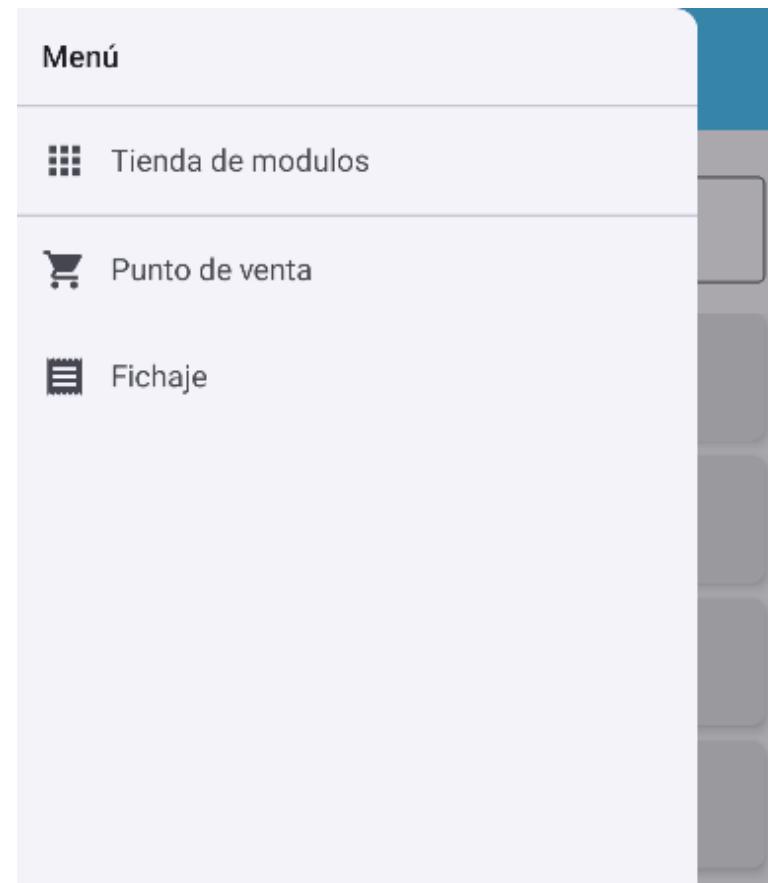
Aun no tiene una cuenta? [Registrar](#)

Ya tiene una cuenta? [Iniciar sesión](#)

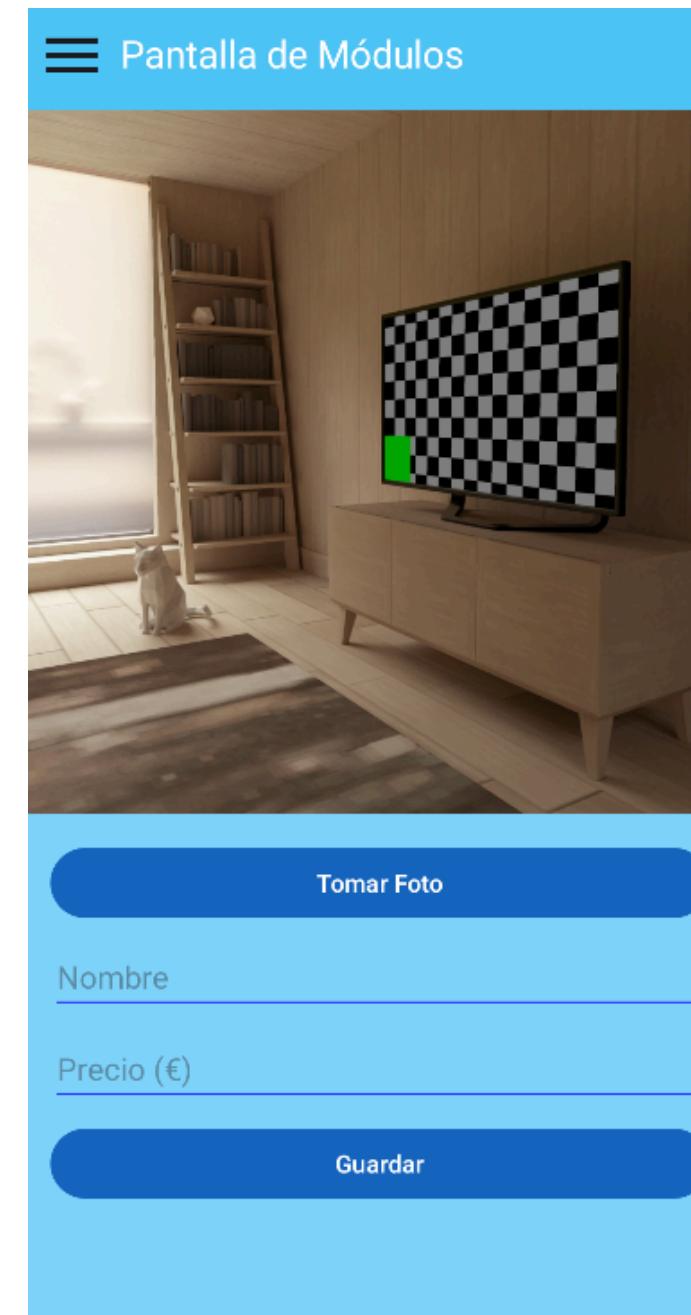
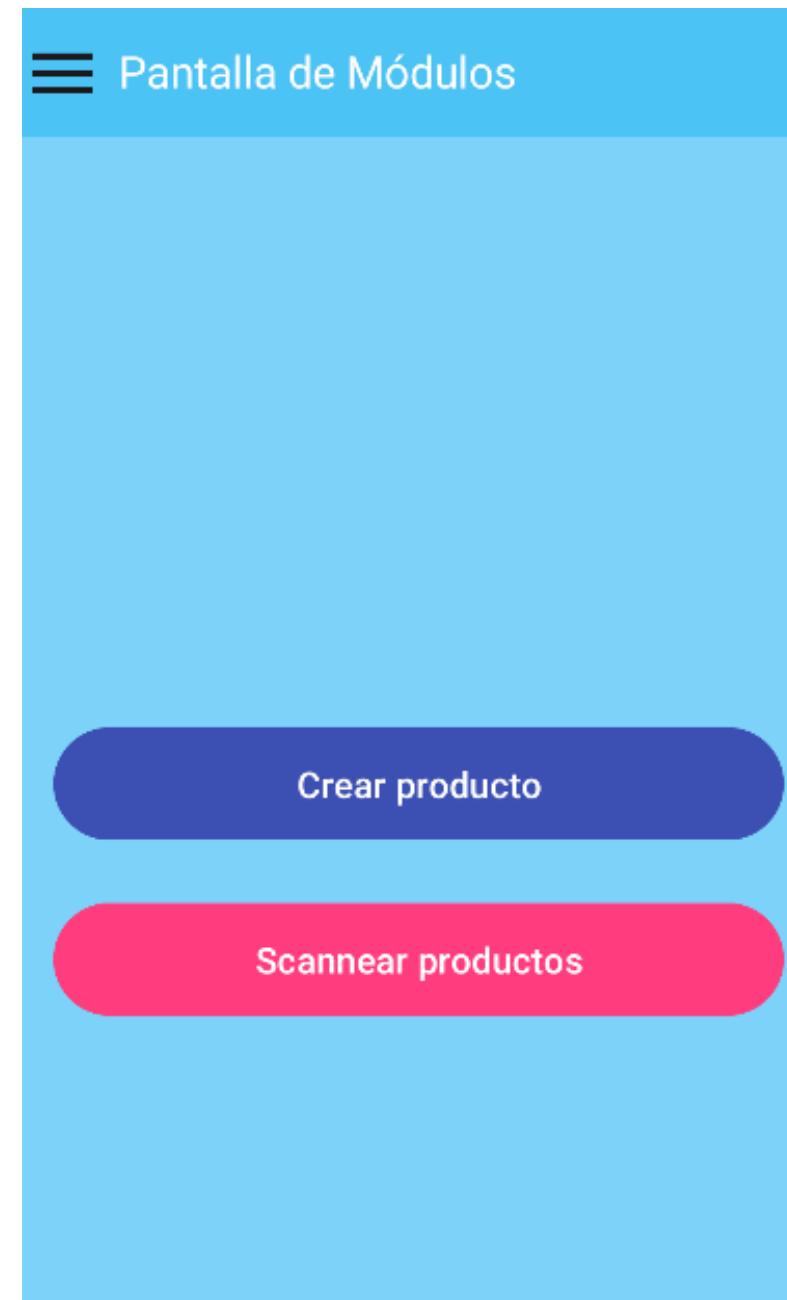
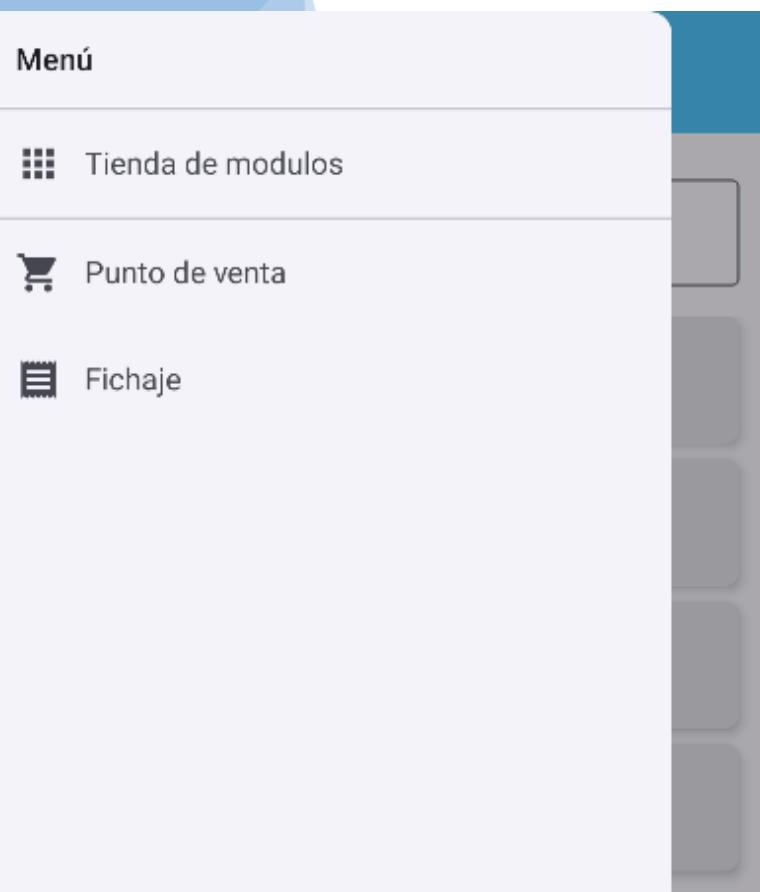
**Registrarse**



# Diseño de la aplicación



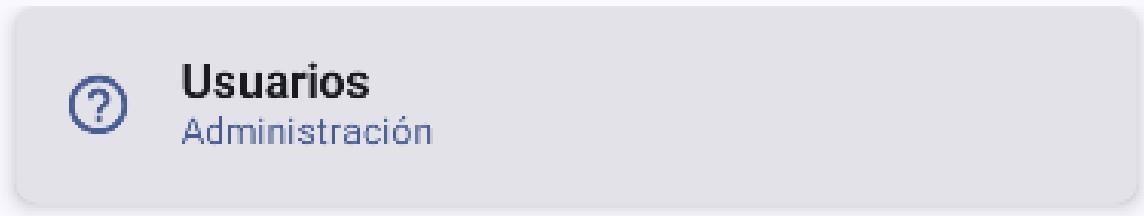
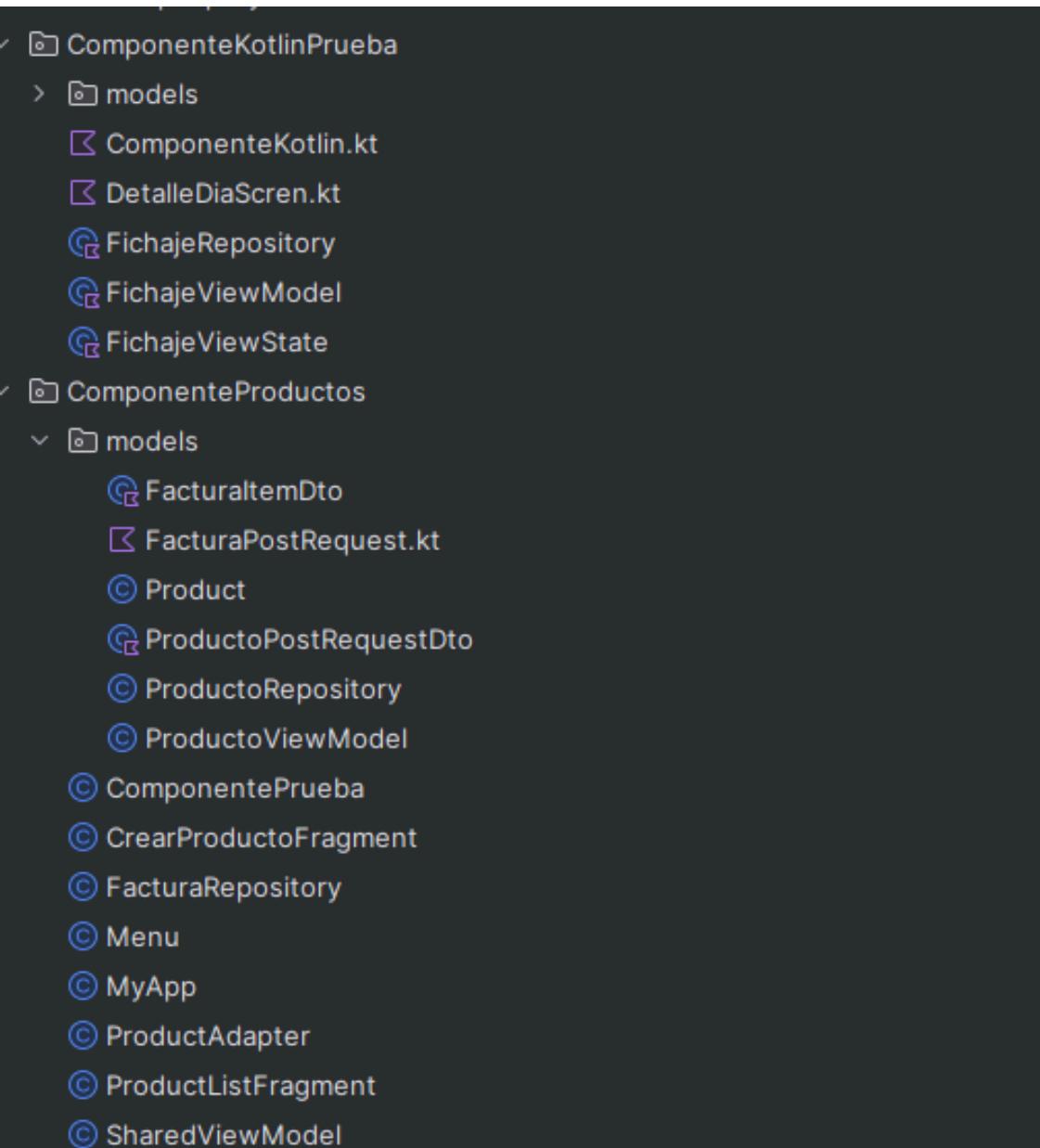
# Diseño de la aplicación



# Desarrollo

## Estructura modular

```
Scaffold(  
    topBar = {  
        TopAppBar(  
            title = { Text(text = "Pantalla de Módulos") },  
            navigationIcon = {  
                IconButton(  
                    onClick = { scope.launch { drawerState.open() } }  
                )  
                Icon(Icons.Default.Menu, contentDescription = "Abrir menú", modifier = Modifier.size(40.dp))  
            }  
        ),  
        colors = TopAppBarDefaults.topAppBarColors(  
            containerColor = Color(color = 0xFF4FC3F7),  
            titleContentColor = Color(color = 0xFFFFFFFF)  
        )  
    )  
    ) { innerPadding ->  
        NavHost(  
            navController = navController,  
            startDestination = "tienda",  
            modifier = Modifier.padding(innerPadding)  
        ) {  
            composable(route: "tienda") {  
                TiendaScreen(  
                    onModuleClick = { modulo ->  
                        sharedVm.select(modulo)  
                        navController.navigate(route: "detalle_modulo")  
                    }  
                )  
            }  
            composable(route: "detalle_modulo") {  
                val seleccionado = sharedVm.seleccionado.collectAsState().value  
                if (seleccionado != null) {  
                    ModuloDetailScreen(  
                        modulo = seleccionado,  
                        onBack = { navController.popBackStack() }  
                    )  
                }  
            }  
        }  
    }  
}
```



DYNAMIC  
FEATURE  
MODELS

# Desarrollo

## Controlador de la base de datos

```
✓ controllers
J Controlador.java
J FacturaRestController.java
J FichajeRestController.java
J ModuloRestController.java
J PerfilRestController.java
J ProductoRestController.java
```

```
✓ services
J FacturaService.java
J FacturaServiceImpl.java
J FichajeService.java
J FichajeServiceImpl.java
J ModuloService.java
J ModuloServiceImpl.java
J PerfilService.java
J PerfilServiceImpl.java
J ProductoService.java
J ProductoServiceImpl.java
```

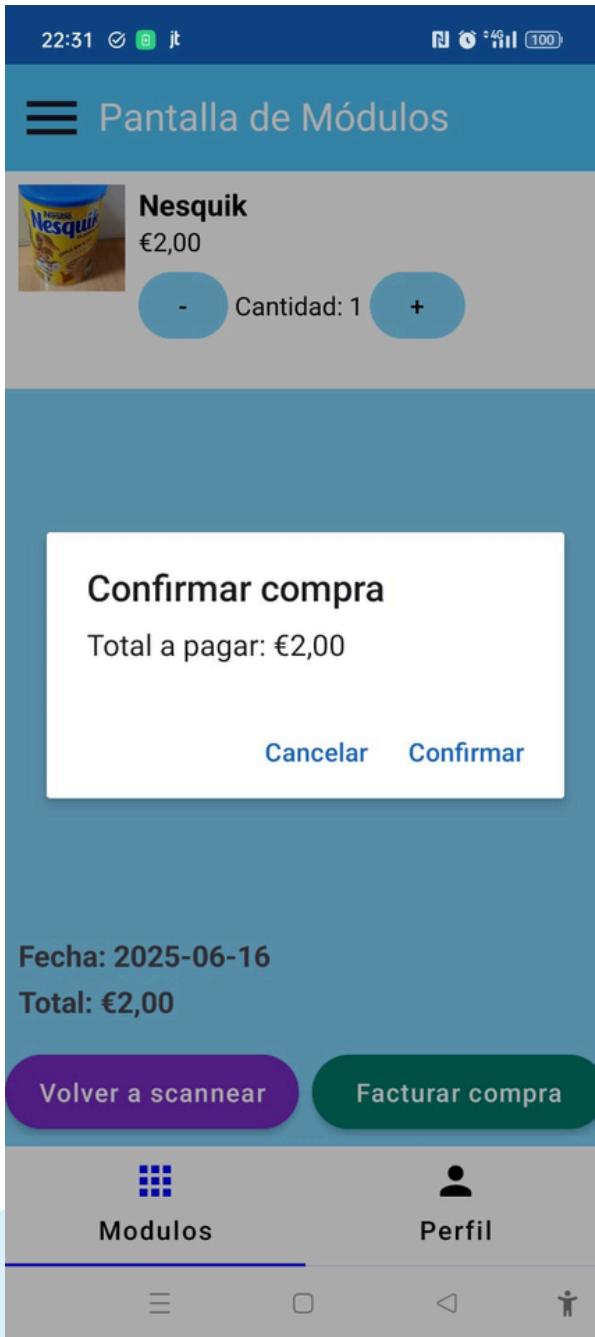
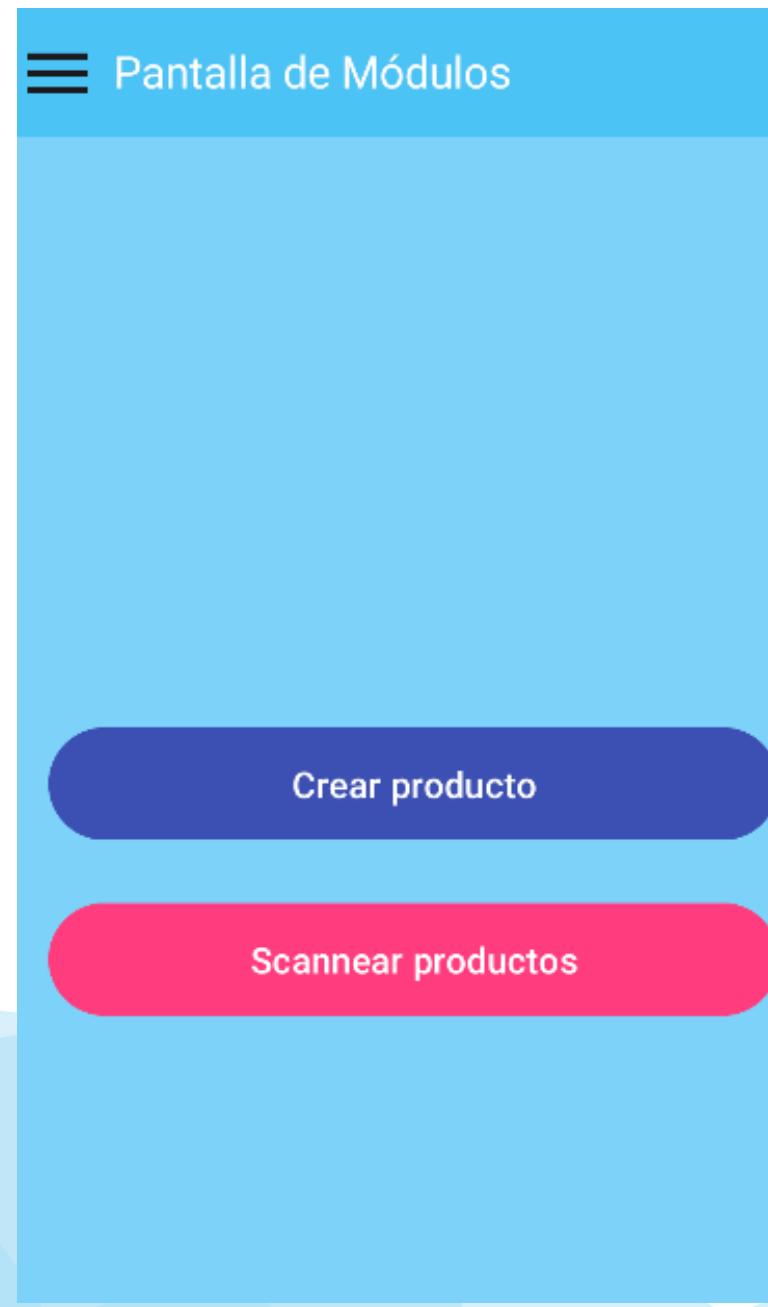
```
✓ mappers
J FacturaProductoRequestMapper.java
J FacturaRequestMapper.java
J FichajeRequestMapper.java
J ModuloRequestMapper.java
J PerfilMapper.java
J ProductoMapper.java
```

```
✓ repositories
J FacturaRepository.java
J FichajeRepository.java
J ModuloRepository.java
J PerfilRepository.java
J ProductoRepository.java
```

```
✓ models
✓ Dtos
✓ Requests
J FacturaItemDto.java
J FacturaPostRequestDto.java
J FacturaProductoPostRequestDto.java
J FichajePostRequestDto.java
J ModuloPostRequestDto.java
J PerfilPatchRequestDto.java
J PerfilPostRequestDto.java
J ProductoPostRequestDto.java
J PerfilDTO.java
J ProductoDto.java
J Factura.java
J FacturaProducto.java
J Fichaje.java
J Modulo.java
J Perfil.java
J Producto.java
```

# Desarrollo

## Modulo-Punto de Venta



# Desarrollo

## Modulo-Punto de Venta CameraX

```
public class MyApp extends Application implements CameraXConfig.Provider {  
    @Override 3 usages  
    public CameraXConfig getCameraXConfig() {  
        // Usa la configuración por defecto de Camera2 y log sólo errores  
        return CameraXConfig.Builder  
            .fromConfig(Camera2Config.defaultConfig())  
            .setMinimumLoggingLevel(Log.ERROR)  
            .build();  
    }  
}
```

```
<!-- Permiso para usar la cámara -->  
<uses-permission android:name="android.permission.CAMERA" />  
<uses-permission android:name="android.permission.INTERNET"/>  
  
<!-- Declaración de la característica de cámara (requerida para CameraX) -->  
<uses-feature android:name="android.hardware.camera.any"  
    android:required="false"/>  
  
<application  
    android:allowBackup="true"  
    android:name=".ComponenteProductos.MyApp"  
    android:dataExtractionRules="@xml/data_extraction_rules"  
    android:fullBackupContent="@xml/backup_rules"  
    android:icon="@drawable/modifeddos"  
    android:label="@string/app_name"  
    android:roundIcon="@drawable/modifedtre"  
    android:supportsRtl="true"  
    android:theme="@style/Theme.ProyectoBussinesOne"  
    android:usesCleartextTraffic="true"  
    tools:targetApi="31">  
    <activity  
        android:name=".MainActivity"
```

# Desarrollo

## Modulo-Punto de Venta CameraX

```
private void startCamera() { 2 usages
    if (cameraStarted) return;
    cameraStarted = true;
    ListenableFuture<ProcessCameraProvider> cameraProviderFuture =
        ProcessCameraProvider.getInstance(requireContext());
    cameraProviderFuture.addListener(() -> {
        try {
            ProcessCameraProvider cameraProvider = cameraProviderFuture.get();
            //Desvincula cualquier caso de uso existente
            cameraProvider.unbindAll();

            // Construye el preview
            Preview preview = new Preview.Builder()
                .setTargetResolution(new android.util.Size( width: 640, height: 480))
                .build();
            preview.setSurfaceProvider(binding.previewView.getSurfaceProvider());

            //Configura imageCapture con una resolución reducida
            imageCapture = new ImageCapture.Builder()
                .setTargetResolution(new android.util.Size( width: 640, height: 480))
                .build();

            //Selecciona cámara trasera
            CameraSelector cameraSelector = CameraSelector.DEFAULT_BACK_CAMERA;

            //Vincula sólo 2 casos de uso: Preview e ImageCapture
            cameraProvider.bindToLifecycle(
                getViewLifecycleOwner(),
                cameraSelector,
                preview,
                imageCapture
            );
        } catch (ExecutionException | InterruptedException e) {
            Log.e( tag: "CameraX", msg: "Error al inicializar la cámara", e);
        } catch (IllegalArgumentException e) {
            Log.e( tag: "CameraX", msg: "No se pudo vincular: demasiados casos de uso", e);
            Toast.makeText(requireContext(), text: "No se pudo iniciar la cámara: combinación no compatible.", Toast.LENGTH_LONG).show();
        }
    }, ContextCompat.getMainExecutor(requireContext()));
}
```

```
private boolean cameraStarted = false; 3 usages
private FragmentCrearProductoBinding binding; 19 usages
private static final int REQUEST_CAMERA_PERMISSION = 101; 2 usages
private ImageCapture imageCapture; 4 usages
private long timestamp; 1 usage
```

```
private void takePhoto() { 1 usage
    String timestamp = new SimpleDateFormat(pattern: "yyyyMMddHHmmss", Locale.US)
        .format(System.currentTimeMillis());
    photoFile = new File(requireContext().getCacheDir(), child: "IMG_" + timestamp + ".jpg");

    ImageCapture.OutputFileOptions options =
        new ImageCapture.OutputFileOptions.Builder(photoFile).build();
    imageCapture.takePicture(options,
        ContextCompat.getMainExecutor(requireContext()),
        new ImageCapture.OnImageSavedCallback() {
            @Override 5 usages
            public void onImageSaved(@NonNull ImageCapture.OutputFileResults outputFileResults) {
                try {
                    Bitmap corrected = rotateBitmapIfRequired(photoFile.getAbsolutePath());
                    binding.capturedImage.setImageBitmap(corrected);

                    // Generar código de barras numérico aleatorio de 9 dígitos
                    int randomCode = (int) (Math.random() * 900000000) + 100000000; // asegura 9 dígitos
                    barcode = String.valueOf(randomCode);

                    Bitmap barcodeBmp = generateBarcode(barcode);
                    binding.barcodeImage.setImageBitmap(barcodeBmp);
                    int paddingPx = (int)(4 * getResources().getDisplayMetrics().density);

                    // aplica fondo blanco y padding
                    binding.barcodeImage.setBackgroundColor(Color.WHITE);
                    binding.barcodeImage.setPadding(paddingPx, paddingPx, paddingPx, paddingPx);
                } catch (IOException e) {
                    Log.e( tag: "PhotoCapture", msg: "Exif rotation failed", e);
                    // en última instancia, muestra el original:
                    binding.capturedImage.setImageBitmap(
                        BitmapFactory.decodeFile(photoFile.getAbsolutePath())
                    );
                }
                Toast.makeText(requireContext(), text: "Foto capturada y código generado", Toast.LENGTH_SHORT).show();
            }
        }
    );
    @Override
    public void onError(@NonNull ImageCaptureException exception) {
        Log.e( tag: "PhotoCapture", msg: "Error taking photo", exception);
        Toast.makeText(requireContext(), text: "Error capturando foto", Toast.LENGTH_SHORT).show();
    }
}
```

# Desarrollo

## Modulo-Fichaje

≡ Pantalla de Módulos

← Junio 2025 →

Lun	Mar	Mié	Jue	Vie	Sáb	Dom
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	1	2	3	4	5	6

Horas trabajadas esta semana  
00h 00m / 40h

Fichar

≡ Pantalla de Módulos

← Mayo 2025 →

Lun	Mar	Mié	Jue	Vie	Sáb	Dom
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1
2	3	4	5	6	7	8

Horas trabajadas esta semana  
00h 00m / 40h

Desfichar (00h:00m)

≡ Pantalla de Módulos

← 2025-05-19

00 06 12 18 23

Desfichar Solicitar fichaje

Fichajes de 19/5/2025

09:00	17:30
19:00	22:00

# Conclusiones

El proyecto ha resultado tan ambicioso como me advirtieron: un ERP completo implica una complejidad y dimensión muy superiores a las de este desarrollo, por lo que su alcance no puede compararse con el de una solución empresarial real. No obstante, a pesar de la envergadura del reto, los objetivos esenciales se han cumplido satisfactoriamente. Este proyecto constituye un valioso añadido a mi portfolio.

# Bibliografía I

Repositorio del proyecto : <https://github.com/JorgeJose05/Proyecto-TFG-BussinesOne>

Definicion ERP: [https://es.wikipedia.org/wiki/Sistema\\_de\\_planificaci%C3%B3n\\_de\\_recursos\\_empresariales](https://es.wikipedia.org/wiki/Sistema_de_planificaci%C3%B3n_de_recursos_empresariales)

Boletín Oficial del Estado sobre la ley de protección de datos personales:

<https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673>

Jetpack compose:

<https://developer.android.com/compose>

Spring Boot:

<https://docs.spring.io/spring-boot/>

PostgresSQL:

<https://www.postgresql.org/docs/>

AWS Elastic beanstalk:

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg>Welcome.html>

Retrofit:

<https://square.github.io/retrofit/> + Github(<https://github.com/square/retrofit>)

# Bibliografía II

Librería de camera:

[https://developer.android.com/media/camera/camerax?utm\\_source=chatgpt.com&hl=es-419](https://developer.android.com/media/camera/camerax?utm_source=chatgpt.com&hl=es-419)

Libreria ZXing : <https://github.com/zxing/zxing>

Libreria retrofit : <https://square.github.io/retrofit/>

Dynamic Feature Models: [https://developer.android.com/guide/playcore/feature-delivery?utm\\_source=chatgpt.com&hl=es-419](https://developer.android.com/guide/playcore/feature-delivery?utm_source=chatgpt.com&hl=es-419)

Repositorio del proyecto : <https://github.com/JorgeJose05/Proyecto-TFG-BussinesOne>

Arquitectura Model View ViewModel:

<https://developer.android.com/topic/architecture?hl=es-419>

**Muchas  
Gracias**