

Two thick, horizontal, slightly wavy bars. The top one is teal and the bottom one is yellow, both spanning the width of the page.

BUILDYOURBD

Autor: José Manuel Flores Marzo y Antonio Guerrero García

Tutor: Javier Martin Rivero

Fecha de entrega: 21/05/2025

Convocatoria: 2024 2025

Documentación TFG

Motivación

Pues la motivación de para realizar este proyecto viene desde tiempo atrás ya que soy un apasionado del mundo fitness y me encanta aprender de todo sobre el porqué no hacer que el fitness sea más cómodo y que llame más la atención a la gente menos apasionada del fitness pues BuidYourBD se encarga de hacer precisamente eso. Tras un largo tiempo usando aplicaciones de fitness vi que no existía ninguna que tuviera un todo en 1 con lo cual motivación extra para hacerlo y conseguir que la comunidad fitness crezca.

Objetivos

Los objetivos del proyecto son bastante sencillos: Conseguir una comunidad sana, estable y con ganas de mejorar tanto física como mentalmente.

Que la gente pueda subir sus entrenamientos o receta por ejemplo entrenamientos hit, recetas con pocas calorías, etc.

Que sobre todo para la gente nueva en este mundo que la app tenga acceso a tu ubicación y puedas meterte en un chat dependiendo del gimnasio que elijas cercano a tu ubicación para que puedas hacer amigos en esos gimnasios tanto desde fuera como desde dentro de ellos.

Que puedas trackear tus alimentos para poder llevar un conteo de tus calorías, grasas, hidratos y proteínas diarios. Serían los objetivos de este proyecto.

Metodología

Se ha decidido utilizar la metodología SCRUM en la que dividiremos el trabajo en “sprints”. En cada sprint se hace una planificación detallada de lo que se hace en cada uno. Que tendrán una duración aproximada de entre 1 semana y 2 semanas.

Backlog

En este caso nuestro Backlog esta dividido en varias partes:

1. Funcionalidades Básicas

- **Registro y Gestión de Perfil de Usuario**
 - o Crear formulario de registro (nombre, correo, contraseña).
 - o Página de inicio de sesión para usuarios existentes.
 - o Recuperación de contraseña.
 - o Edición de perfil de usuario (foto, datos personales).
 - o Visualización del historial de entrenamientos.
- **Modo Oscuro**
 - o Implementar un modo oscuro para la aplicación.
 - o Permitir al usuario alternar entre modo claro y oscuro según preferencia.
- **Seguimiento de Progreso**
 - o Registro de peso y calorías quemadas.
 - o Visualización de estadísticas de progreso en gráficos (peso, calorías, etc.).
 - o Establecer metas de progreso personalizadas.
- **Recordatorios**
 - o Implementar recordatorios para beber agua.
 - o Recordatorios para comer.
 - o Recordatorios para entrenar

2. **Rutinas y Recetas**

- **Rutinas Personalizadas**
 - o Permitir que el usuario cree sus propias rutinas de entrenamiento.
 - o Crear plantillas de rutinas prediseñadas.
 - o Recomendación de rutinas según objetivos del usuario.
- **Recetario de Cocina**
 - o Crear base de datos de recetas con información nutricional.
 - o Permitir que el usuario filtre recetas por tipo.
 - o Función que permite que el usuario guarde sus recetas favoritas.

3. **Funcionalidades Avanzadas**

- **Obtención de Gimnasios y Comunidad**
 - o Integración de APIs para obtener gimnasios cercanos a la ubicación del usuario.
 - o Crear una sección para mostrar la lista de gimnasios y sus detalles (ubicación, horarios, etc.)
 - o Implementación de una funcionalidad de chats en los gimnasios para interactuar con otros usuarios.
- **Comunidad Fitness y Contenido Compartido**
 - o Crear un recetario compartido donde los usuarios puedan subir y compartir recetas.
 - o Permitir que los usuarios compartan sus rutinas de entrenamiento en una sección comunitaria.
 - o Implementar un sistema de comentarios y valoraciones para recetas y rutinas de entrenamiento.

4. Optimización y Mejoras

- **Interfaz y Experiencia de Usuario (UX/UI)**

- o Diseño responsivo.

- o Mejora de la navegación y accesibilidad.

5. Seguridad Protección de Datos

- o Encriptación de contraseñas y datos sensibles.

- o Implementación de autenticación de dos factores(2FA).

6. Corrección de Errores Mantenimiento y Soporte.

- o Solución de errores en la carga de datos de entrenamiento.

- o Actualización de la base de datos para incluir nuevas recetas y rutinas.

Tecnologías y herramientas

Android Studio: Es el framework para poder hacer aplicaciones en Android.

MySQL: Se usará para hacer la base de datos de la aplicación.

Kotlin: Sera el lenguaje usado para poder hacer la aplicación en Android.

Google Maps API: Permitirá integrar mapas interactivos en mi aplicación.

FireBase Authentication: Es una herramienta muy útil para gestionar la autenticación de usuarios de manera rápida y segura.

Retrofit: Es una biblioteca de Android que facilita la conexión con APIs RESTful para enviar y recibir datos.

Firestore: Es una base de datos en tiempo real de Firebase. **WorkManager:** es una herramienta para ejecutar tareas en segundo plano, incluso si la app se cierra o el dispositivo se reinicia.

Google Fit API: La Google Fit API proporciona acceso a datos relacionados con la actividad física del usuario. En "BuildYourBD".

Hemos decidido usar estas herramientas por estas decisiones. Se decidió hacer un programa en kotlin para poder hacer un app para lo cual necesitamos Android studio, MySQL y firestore serán las bases de datos que se intentaran implementar porque MySQL es una base de datos normal y firestore es una base de datos en tiempo real, Google maps API es la mejor api para poder realizar la cuestión de obtener los gimnasios cerca de tu ubicación y además en parte es gratis , firebase Authentication es una de las herramientas que hemos visto más fiables para la autenticación de usuarios de manera rápida y segura, retrofit es una de las herramientas que vamos a usar para enviar y recibir datos porque al parecer está bastante optimizada, es muy completa y es gratis, WorkManager hemos visto que es una herramienta que no da mucho conflicto para implementarla en proyectos de Android studio , Y Google fit API porque era de las pocas API fitness que nos permitían usar la API de forma gratuita.

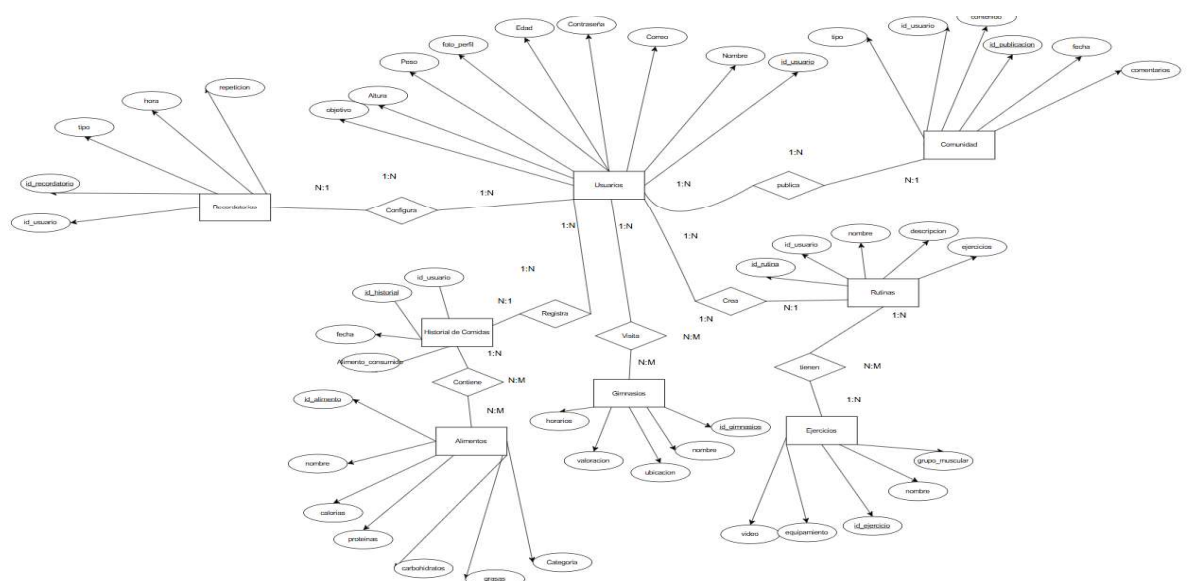
Mockup figma:

<https://www.figma.com/design/QyhJYpIXA8cLiqMLhLUOVL/Untitled?node-id=0-1&t=lwQ2TipgdHMdQMou-1>

Motivación (Ingles)

The motivation behind this project comes from my deep passion for the fitness world, which has grown stronger over time. I've always been curious to learn more about health, nutrition, and training. That passion sparked the idea for BuildYourBD—a fitness app designed to make the journey easier and more attractive, not only for enthusiasts like me, but especially for those who find it hard to stay consistent or get started. While exploring different fitness apps, I noticed that most of them are either too focused on a single aspect—like workouts or calorie tracking—or overly complex. None of them seemed to offer a truly complete, user-friendly, and motivating experience. That gap inspired me to create an all-in-one solution. BuildYourBD brings together essential fitness tools in one place: a food database for tracking calories and meals, personalized training routines, progress tracking features, hydration and workout reminders, and even community features to share recipes and workouts. It also includes dark mode and integration with APIs for gym locations and health data. My goal is to build not just another app, but a complete ecosystem that supports users in building long-lasting, healthy habits—helping them become the best version of themselves, both physically and mentally.

E-R




```

classDiagram
    class Gimnasio {
        + Id : string
        + Nombre : string
        + Direccion : string
        + Latitud : double
        + Longitud : double
    }
    class ChatGimnasio {
        + Id : string
        + GimnasioId : string
        + UsuarioId : string
        + Mensaje : string
        + FechaHora : string
    }
    class Progreso {
        + Id : string
        + UsuarioId : string
        + Fecha : date
        + Peso : double
        + CaloriasQue : double
        + Observaciones : string
    }
    class Usuario {
        + Id : string
        + Nombre : string
        + Email : string
        + Password : string
        + Edad : integer
        + Altura : double
        + PesoActual : double
        + Genero : string
        + Objetivo : string
        - registrar()
        - iniciarSesion()
        - actualizarProgreso()
    }
    class Recordatorio {
        + Id : string
        + UsuarioId : string
        + Titulo : string
        + Descripcion : string
        + ListaIngredientes : string
    }
    class HistorialAlimentacion {
        + Id : string
        + Fecha : date
        + UsuarioId : string
        - agregarAlimento()
        - calcularTotales()
    }
    class Alimento {
        + Id : string
        + Nombre : string
        + Calorias : double
        + Proteinas : double
        + Grasas : double
        + Carbohidratos : double
        + Categoria : string
    }
    class RecetaComunidad {
        + Id : string
        + UsuarioId : string
        + Titulo : string
        + Descripcion : string
        + ListaIngredientes : string
    }
    class RutinaPersonalizada {
        + Id : string
        + UsuarioId : string
        + Nombre : string
        - agregarEntrenamiento()
        - eliminarEntrenamiento()
    }
    class Entrenamiento {
        + Id : string
        + Titulo : string
        + Descripcion : string
        + DuracionMin : integer
        + Nivel : string
        + Tipo : string
    }

    Gimnasio "1" -- "*" ChatGimnasio : chatGimnasio
    ChatGimnasio "*" -- "1" Gimnasio : chatGimnasio
    ChatGimnasio "*" -- "1" Progreso : progreso
    Progreso "1" -- "*" ChatGimnasio : progreso
    Usuario "1" -- "*" Recordatorio : recordatorio
    Recordatorio "*" -- "1" Usuario : recordatorio
    Usuario "1" -- "*" HistorialAlimentacion : historialAlimentacion
    HistorialAlimentacion "*" -- "1" Usuario : historialAlimentacion
    Usuario "1" -- "*" RutinaPersonalizada : rutinaPersonalizada
    RutinaPersonalizada "*" -- "1" Usuario : rutinaPersonalizada
    RutinaPersonalizada "*" -- "*" Entrenamiento : entrenamiento
    Entrenamiento "*" -- "*" RutinaPersonalizada : entrenamiento
    HistorialAlimentacion "*" -- "*" Alimento : alimento
    Alimento "*" -- "*" HistorialAlimentacion : alimento
    RecetaComunidad "*" -- "*" Usuario : usuario
    Usuario "*" -- "*" RecetaComunidad : usuario
  
```

The diagram illustrates the relationships between various entities in a fitness application. The classes and their attributes are as follows:

- Gimnasio**: + Id : string, + Nombre : string, + Direccion : string, + Latitud : double, + Longitud : double
- ChatGimnasio**: + Id : string, + GimnasioId : string, + UsuarioId : string, + Mensaje : string, + FechaHora : string
- Progreso**: + Id : string, + UsuarioId : string, + Fecha : date, + Peso : double, + CaloriasQue : double, + Observaciones : string
- Usuario**: + Id : string, + Nombre : string, + Email : string, + Password : string, + Edad : integer, + Altura : double, + PesoActual : double, + Genero : string, + Objetivo : string, - registrar(), - iniciarSesion(), - actualizarProgreso()
- Recordatorio**: + Id : string, + UsuarioId : string, + Titulo : string, + Descripcion : string, + ListaIngredientes : string
- HistorialAlimentacion**: + Id : string, + Fecha : date, + UsuarioId : string, - agregarAlimento(), - calcularTotales()
- Alimento**: + Id : string, + Nombre : string, + Calorias : double, + Proteinas : double, + Grasas : double, + Carbohidratos : double, + Categoria : string
- RecetaComunidad**: + Id : string, + UsuarioId : string, + Titulo : string, + Descripcion : string, + ListaIngredientes : string
- RutinaPersonalizada**: + Id : string, + UsuarioId : string, + Nombre : string, - agregarEntrenamiento(), - eliminarEntrenamiento()
- Entrenamiento**: + Id : string, + Titulo : string, + Descripcion : string, + DuracionMin : integer, + Nivel : string, + Tipo : string

The relationships between these classes are defined by the following associations:

- Gimnasio** (1) is associated with **ChatGimnasio** (*).
- ChatGimnasio** (*) is associated with **Gimnasio** (1).
- ChatGimnasio** (*) is associated with **Progreso** (1).
- Progreso** (1) is associated with **ChatGimnasio** (*).
- Usuario** (1) is associated with **Recordatorio** (*).
- Recordatorio** (*) is associated with **Usuario** (1).
- Usuario** (1) is associated with **HistorialAlimentacion** (*).
- HistorialAlimentacion** (*) is associated with **Usuario** (1).
- Usuario** (1) is associated with **RutinaPersonalizada** (*).
- RutinaPersonalizada** (*) is associated with **Usuario** (1).
- RutinaPersonalizada** (*) is associated with **Entrenamiento** (*).
- Entrenamiento** (*) is associated with **RutinaPersonalizada** (*).
- HistorialAlimentacion** (*) is associated with **Alimento** (*).
- Alimento** (*) is associated with **HistorialAlimentacion** (*).
- RecetaComunidad** (*) is associated with **Usuario** (*).
- Usuario** (*) is associated with **RecetaComunidad** (*).

[illegible]

Caso de Uso 1	Registrarse
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede registrarse
Descripción	Permite al usuario registrarse en la app para poder navegar por ella.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 2	Recuperar Contraseña
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede recuperar su contraseña.
Descripción	La app permite al usuario recuperar su contraseña.
Referencias	Registrarse (1)
Comentarios	Ningún comentario
Caso de Uso 3	Iniciar sesión
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede Iniciar Sesión.
Descripción	Permite al usuario iniciar sesión en la app con sus credenciales propias.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 4	Validar Credenciales
Alias	
Actores	Sistema
Requisitos Funcional	El sistema valida las credenciales del usuario.
Descripción	Permite al sistema validar las credenciales del usuario.
Referencias	Iniciar Sesión (3)
Comentarios	Ningún comentario

Caso de Uso 5	Ver alimentos
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede ver sus alimentos
Descripción	Permite al usuario ver sus alimentos
Referencias	
Comentarios	Ningún comentario

Caso de Uso 6	Filtrar alimentos por categoría
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede filtrar alimentos por categoría
Descripción	Permite al usuario filtrar alimentos por categoría.
Referencias	Ver alimentos (5)
Comentarios	Ningún comentario

Caso de Uso 7	Consultar historial de comidas
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede consultar su historial de comidas
Descripción	Permite al usuario consultar su historial de comidas.
Referencias	Filtrar alimentos por categoría (6), Ver alimentos (5)
Comentarios	Ningún comentario

Caso de Uso 8	Eliminar alimento del historial
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede eliminar alimentos de su historial
Descripción	Permite al usuario eliminar alimentos de su historial.
Referencias	Filtrar alimentos por categoría (6), Ver alimentos (5)
Comentarios	Ningún comentario

Caso de Uso 9	Introducir Peso
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede introducir su peso
Descripción	Permite al usuario introducir su peso.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 10	Evaluación del peso
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede evaluar su peso
Descripción	Permite al usuario evaluar su peso en el tiempo que lleve.
Referencias	Introducir Peso (9)
Comentarios	Ningún comentario

Caso de Uso 11	Ver progreso
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede ver su progreso
Descripción	Permite al usuario ver su progreso.
Referencias	Evaluación del peso (10), Introducir peso (9)
Comentarios	Ningún comentario

Caso de Uso 12	Gráfico de peso y calorías
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede ver un gráfico de su peso y calorías
Descripción	Permite al usuario ver un gráfico de su peso y calorías diario.
Referencias	Ver progreso (11)
Comentarios	Ningún comentario

Caso de Uso 13	Crear Recordatorio
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede crear un recordatorio
Descripción	Permite al usuario crear un recordatorio (comida, agua, etc.)
Referencias	
Comentarios	Ningún comentario

Caso de Uso 14	Editar Recordatorio
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede editar un recordatorio
Descripción	Permite al usuario editar un recordatorio que haya creado.
Referencias	Crear Recordatorio (13)
Comentarios	Ningún comentario

Caso de Uso 15	Eliminar Recordatorio
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede eliminar un recordatorio
Descripción	Permite al usuario eliminar un recordatorio que haya creado.
Referencias	Crear Recordatorio (13), Editar Recordatorio (14)
Comentarios	Ningún comentario

Caso de Uso 16	Crear Rutina
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede crear una rutina de entrenamiento
Descripción	Permite al usuario crear una rutina de entrenamiento.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 17	Editar Rutina
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede editar sus rutinas
Descripción	Permite al usuario editar sus rutinas de entrenamiento.
Referencias	Crear Rutina (16)
Comentarios	Ningún comentario

Caso de Uso 18	Eliminar Rutina
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede eliminar sus rutinas
Descripción	Permite al usuario eliminar sus rutinas de entrenamiento.
Referencias	Editar Rutina (17), Crear Rutina (16)
Comentarios	Ningún comentario

Caso de Uso 19	Ver Rutinas
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede ver sus rutinas
Descripción	Permite al usuario ver sus rutinas creadas
Referencias	Editar Rutina (17), Crear Rutina (16)
Comentarios	Ningún comentario

Caso de Uso 20	Subir Recetas
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede subir recetas
Descripción	Permite al usuario subir recetas.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 21	Ver recetas
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede ver recetas
Descripción	Permite al usuario ver recetas de la app
Referencias	Subir recetas (20)
Comentarios	Ningún comentario

Caso de Uso 22	Subir entrenamiento
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede subir entrenamientos
Descripción	Permite al usuario subir entrenamientos a la app.
Referencias	Ver recetas (21), Subir recetas (20)
Comentarios	Ningún comentario

Caso de Uso 23	Ver entrenamientos
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede ver entrenamientos
Descripción	Permite al usuario ver los entrenamientos de la app.
Referencias	Subir entrenamiento (22), Ver recetas (21), Subir recetas (20)
Comentarios	Ningún comentario

Caso de Uso 24	Cambiar Modo Oscuro
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede cambiar a modo oscuro
Descripción	Permite al usuario cambiar la app a modo oscuro
Referencias	
Comentarios	Ningún comentario

Caso de Uso 25	Editar perfil de usuario
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede editar su perfil de usuario
Descripción	Permite al usuario editar su perfil de usuario.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 26	Eliminar Cuenta de Usuario
Alias	
Actores	Usuario
Requisitos Funcional	El usuario puede eliminar su cuenta
Descripción	Permite al usuario eliminar su cuenta de usuario.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 27	Guardar los datos
Alias	
Actores	Sistema
Requisitos Funcional	El sistema guarda los datos
Descripción	El sistema guarda los datos del usuario.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 28	Cálculo de las calorías
Alias	
Actores	Sistema
Requisitos Funcional	El sistema calcula las calorías
Descripción	Permite al sistema el cálculo de las calorías del usuario
Referencias	
Comentarios	Ningún comentario

Caso de Uso 29	Enviar Recordatorios
Alias	
Actores	Sistema
Requisitos Funcional	El sistema envía recordatorios
Descripción	Permite al sistema enviar recordatorios al usuario.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 30	Gráfico de peso y Calorías
Alias	
Actores	Sistema
Requisitos Funcional	El sistema proporciona un gráfico de peso y calorías
Descripción	Proporciona al usuario un gráfico de peso y calorías.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 31	Cargar Interfaz
Alias	
Actores	Sistema
Requisitos Funcional	El sistema carga la interfaz de la pp
Descripción	El sistema carga la interfaz de la app para la visualización del usuario.
Referencias	
Comentarios	Ningún comentario

Caso de Uso 32	Asociar rutina o comida con el usuario
Alias	
Actores	Sistema
Requisitos Funcional	El sistema asocia la rutina o comida el usuario
Descripción	El sistema carga asocial la rutina o comida al usuario.
Referencias	
Comentarios	Ningún comentario

Prototipo

```
// lógica de inicio de sesión
private fun login() {
    val email = loginEmail!!.text.toString()
    val password = loginPass!!.text.toString()

    if (email.isEmpty() || password.isEmpty()) {
        Toast.makeText(context, this, text "Por favor ingresa todos los campos", Toast.LENGTH_SHORT)
        return
    }


    // Aquí se puede agregar la lógica para validar el email y la contraseña
    if (email == "usuario@dominio.com" && password == "123456") {
        Toast.makeText(context, this, text "Bienvenido", Toast.LENGTH_SHORT).show()
        // Redirigir a la pantalla principal o al dashboard
    } else {
        Toast.makeText(context, this, text "Credenciales incorrectas", Toast.LENGTH_SHORT).show()
    }
}

// lógica para iniciar sesión como invitado
private fun loginAsGuest() {
    Toast.makeText(context, this, text "Iniciando sesión como invitado", Toast.LENGTH_SHORT).show()
    // Aquí iría la lógica para iniciar sesión como invitado
}

// lógica para iniciar sesión con Google
private fun loginWithGoogle() {
    Toast.makeText(context, this, text "Iniciando sesión con Google", Toast.LENGTH_SHORT).show()
    // Aquí iría la lógica para la autenticación de Google
}


// lógica para ir a la pantalla de registro
private fun goToRegister() {
    Toast.makeText(context, this, text "Redirigiendo a registro", Toast.LENGTH_SHORT).show()
    // Aquí puedes redirigir a una actividad de registro
}
}
```


En este caso la lógica para iniciar sesión en el que básicamente trabajamos el que no haya algún campo vacío o que se intente insertar algo fuera de los campos puestos en el código.



INICIAR SESIÓN

Usuario o correo electrónico

 tunombre@gmail.com

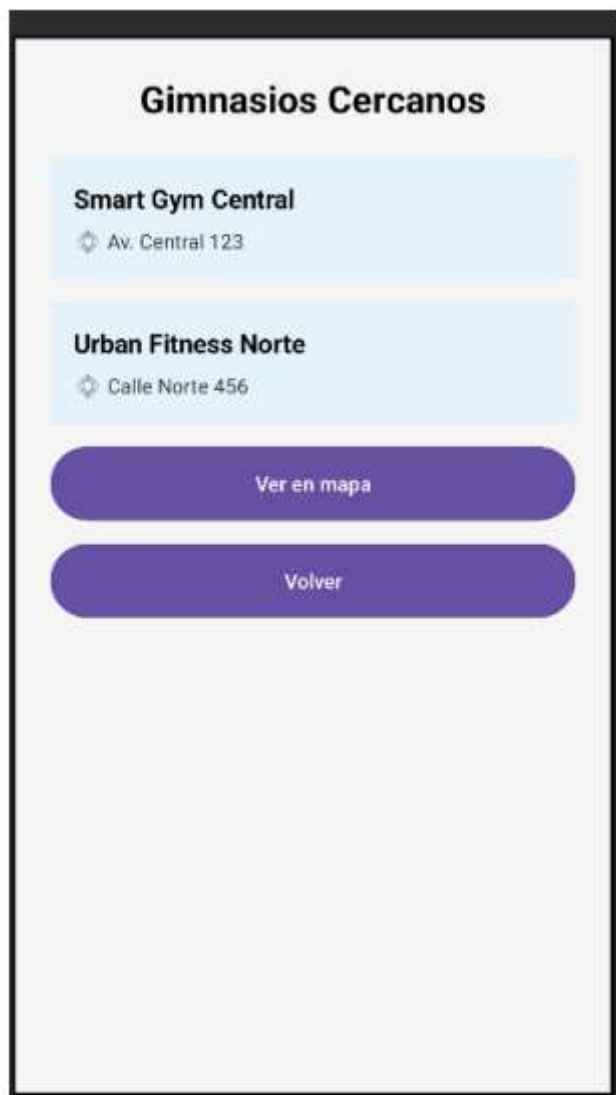
 Contraseña

Entrar

Google >
¿No tienes cuenta? Regístrate

Invitado

Con la página de inicio de sesión



Búsqueda de gimnasios según tu ubicación. En este caso necesitaremos la implementación de varias cosas.

```
implementation 'com.google.android.gms:play-services-maps:18.1.0'  
implementation 'com.google.android.gms:play-services-location:21.0.1'
```

La implementación para la configuración de Google maps, añadir estas dos líneas para los permisos de ubicación:

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

Y no menos importante nuestro api key de Google maps.

```
package com.example.buildyourbd

import android.content.Context
import android.location.Location
import com.google.android.gms.location.LocationServices
import com.google.android.gms.tasks.OnSuccessListener

class @innasio(private val context: Context) {

    private val fusedLocationClient = LocationServices.getFusedLocationProviderClient(context)

    fun obtenerUbicacion@usuario(callback: (Double, Double) -> Unit) {
        if (androidx.core.content.ContextCompat.checkSelfPermission(
            context, android.Manifest.permission.ACCESS_FINE_LOCATION
        ) == android.content.pm.PackageManager.PERMISSION_GRANTED
        ) {
            fusedLocationClient.getLastLocation().addOnSuccessListener { location: Location? ->
                location?.let {
                    callback(it.latitude, it.longitude)
                }
            }
        }
    }
}
```

Y este sería nuestro código para obtener la ubicación del usuario.

Pruebas valores límite

```
import org.junit.Assert
import org.junit.Test

class LoginTest {
    // Prueba para un correo minios de longitud válida (7 caracteres)
    @Test
    fun testValidEmailMinLength() {
        val email = "a@a.com" // Minimo 7 caracteres
        val result = isValidEmail(email)
        Assert.assertTrue(message = "El correo debería ser válido", result)
    }

    // Prueba para un correo con 100 caracteres (máximo permitido)
    @Test
    fun testValidEmailMaxLength() {
        val email = "a".repeat(94) + "@domain.com" // 100 caracteres
        val result = isValidEmail(email)
        Assert.assertTrue(message = "El correo debería ser válido", result)
    }

    // Método de validación simple para el correo
    private fun isValidEmail(email: String?): Boolean {
        return email != null && email.matches("^[\\w-\\.]+@([\\w-]+\\.){2,4}[\\w-]+\\.?$".toRegex())
    }
}

class LoginTest2 {
    // Prueba para una contraseña minima válida (6 caracteres)
    @Test
    fun testValidPasswordMinLength() {
        val password = "123456" // Minimo 6 caracteres
        val result = isValidPassword(password)
        Assert.assertTrue(message = "La contraseña debería ser válida", result)
    }
}
```

```
class LoginTest2 {
    // Prueba para una contraseña minima válida (6 caracteres)
    @Test
    fun testValidPasswordMinLength() {
        val password = "123456" // Minimo 6 caracteres
        val result = isValidPassword(password)
        Assert.assertTrue(message = "La contraseña debería ser válida", result)
    }

    // Prueba para una contraseña máxima válida (20 caracteres)
    @Test
    fun testValidPasswordMaxLength() {
        val password = "12345678901234567890" // Máximo 20 caracteres
        val result = isValidPassword(password)
        Assert.assertTrue(message = "La contraseña debería ser válida", result)
    }

    // Método de validación simple para la contraseña
    private fun isValidPassword(password: String?): Boolean {
        return password != null && password.length >= 6 && password.length <= 20
    }
}
```

Requisitos Funcionales (RF)

Son las características que la aplicación debe cumplir para su funcionamiento.

1. Gestión de calorías y alimentación

- Permitir a los usuarios registrar alimentos consumidos.
- Contar con una base de datos de alimentos con información nutricional.
- Calcular el total de calorías ingeridas diariamente.

2. Seguimiento del progreso

- Registrar el peso del usuario a lo largo del tiempo.
- Mostrar estadísticas de calorías quemadas y consumidas.
- Guardar métricas corporales (IMC, grasa corporal, etc.).

3. Recordatorio y notificaciones

- Configurar alertas para beber agua, comer o entrenar.
- Notificar al usuario sobre sus metas diarias.

4. Rutinas y Entrenamientos

- Permitir a los usuarios crear y personalizar rutinas de entrenamiento.
- Proporcionar planes de ejercicios recomendados.

5. Recetario y Comunidad

- Permitir a los usuarios compartir recetas saludables.
- Implementar un foro/chat donde la comunidad pueda interactuar.

6. Integración con APIs Externas

- Google Fit API: Sincronización de datos de actividad física.
- Google Maps API: Mostrar gimnasios cercanos.

Requisitos No Funcionales (RNF)

Son los atributos de calidad del sistema:

1. Seguridad

- Uso de Firebase Authentication para registro y acceso seguro.
- Protección de datos de usuarios con Firestore y encriptación.

2. Escalabilidad y Rendimiento

- Base de datos en Firestore para manejar un gran volumen de datos.
- Uso de WorkManager para tareas en segundo plano.

3. Usabilidad y accesibilidad

- Diseño responsive y modo oscuro.
- Interfaz fácil de usar con opciones de personalización.

Diagrama de Gantt



Conclusiones

Se han alcanzado los objetivos previstos del proyecto de manera no del todo acertada a como se quería acabar, pero se ha hecho lo máximo posible que es lo importante. En mi opinión siento que ha sido un proyecto que no hemos tenido tanto tiempo para desarrollar por temas ajenos a la institución, pienso que es un proyecto bastante bueno y con una muy buena proyección.

Vías Futuras

Implementaciones y funcionalidades que se podrán añadir o se quieren añadir a la aplicación.

Una mejora de la interfaz, añadir los chats dependiendo del gimnasio, base de datos con los alimentos información nutricional (grasas, hidratos, etc) ya predefinidos en la app.