

Temporizadores

Nombre del temporizador



Duración (en minutos)

CREAR TEMPORIZADOR



Temporis

Temporis:

Aplicación para la optimización de nuestro tiempo y para la plataforma Android

Alumno: Kevin Zamora Amela

Asignatura: Programación multimedia y dispositivos móviles

Curso: 2o de DAM (o Desarrollo de Aplicaciones Multiplataforma)

Entrega: Abril 2025



Olvidé mi contraseña

¿Aún no tienes cuenta?

ENTRAR

Entrar con Google

Prueba1

10 min

Inactivo

Creado el: 18/04/2025 16:29

09:48



ImageView

Email

Contra

Button

Índice

• Propuesta inicial	3
• Proceso de desarrollo	6
• Funcionalidades desarrolladas	8
• Problemas y dificultades durante el desarrollo	9
• Capturas de pantalla	10

Temporis

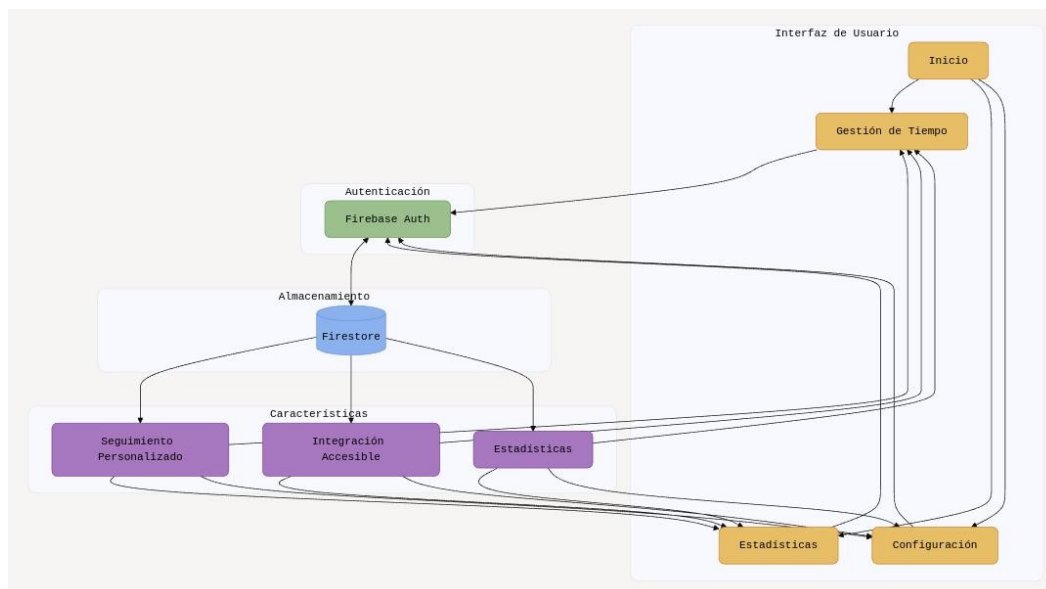
Aplicación de Gestión del Tiempo Personalizada

Descripción conceptual de la aplicación:

- Esta propuesta presenta una aplicación móvil para la gestión del tiempo operativo y por ende, también de nuestra productividad, pero diseñada de forma más accesible, permitiendo su adaptabilidad ante usuarios con diversidad funcional. Esta futura adaptabilidad nos permitirá poder enfocar el proyecto de cara hacia el TFG del presente ciclo formativo en Desarrollo de Aplicaciones Multiplataforma (o DAM).
- La funcionalidad básica de nuestra aplicación consistirá en la implementación de un sistema de contadores personalizados, integrado con seguimiento de tiempo. Este permitirá a los usuarios optimizar su productividad mediante la creación de objetivos personalizados y el seguimiento de actividades específicas, aunque con algunas funcionalidades complementarias que nos gustaría añadir posteriormente. Se ha tomado como referencia la aplicación de gestión de tiempo Toggl Track y nuestro objetivo de partida (o Mínimo Producto Viable) consistirá en desarrollar una aplicación similar a la citada, aunque puede que descartando o eliminando de primeras algunas funcionalidades.
- En consecuencia, objetivo principal consiste desarrollar una aplicación móvil que permita a los usuarios gestionar eficientemente su tiempo mediante un sistema flexible de contadores personalizados, integrado con características de accesibilidad y seguimiento de actividades. La aplicación debe ser intuitiva y adaptable a diferentes necesidades individuales.
- Para su desarrollo y posterior despliegue, utilizaremos los servicios de Google 'Firebase Auth' y 'Firebase Firestore'. Al menos esa es nuestra intención.
- Finalmente, se utilizará el lenguaje de programación Kotlin para su desarrollo.

Boceto o diagrama de la estructura de la aplicación:

- Esquema de navegación (en versión preliminar):



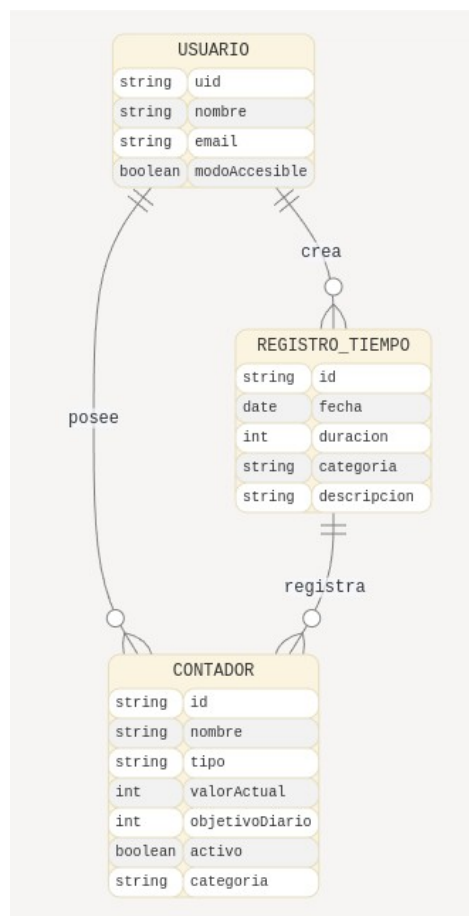
vin Zamora

En este esquema se muestra la funcionalidad que nos gustaría poder implementar, aunque no nos resulte posible desarrollarla/implementarla en su totalidad durante el desarrollo de la presente asignatura.

Las funcionalidades a implementar en una primera fase resultarán la gestión del acceso/registro de los usuarios y la implementación de la funcionalidad relacionada con la gestión del tiempo dedicado a cada tarea u objetivo.

Las funcionalidades de: configuración, estadísticas, seguimiento personalizado e integración accesible se implementarán en futuras versiones de nuestra aplicación, si resulta posible.

- **Esquema relacional entre entidades** (en versión preliminar)
- Un USUARIO puede crear múltiples REGISTROS_TIEMPO y CONTADORES.
- Los REGISTROS_TIEMPO pueden estar asociados con CONTADORES para el seguimiento y se guardarán “en la cuenta de cada usuario/a”, quedando enlazados con este/a.
- El campo modoAccesible permite personalizar la interfaz según necesidades individuales. Esta sería una funcionalidad a añadir posteriormente.



Priorización de Funcionalidades MVP (o Mínimo Producto Viable)

A continuación y para maximizar el impacto del proyecto propuesto, presento una nueva organización de las funcionalidades a desarrollar durante esta asignatura y posteriormente, también para el desarrollo del proyecto de final de grado. Las he organizado y distribuido en tres niveles de prioridad tomando como base el tiempo disponible, del que dispongo al compaginar el trabajo, el resto de tareas del curso y el desarrollo de la presente propuesta de aplicación. La planificación de funcionalidades propuesta es la siguiente:

Nivel 1: Funcionalidades Esenciales (MVP Básico)

- Autenticación básica con Firebase Auth
- Sistema de contadores personalizados básico
- Registro de tiempo para actividades
- Interfaz básica de usuario

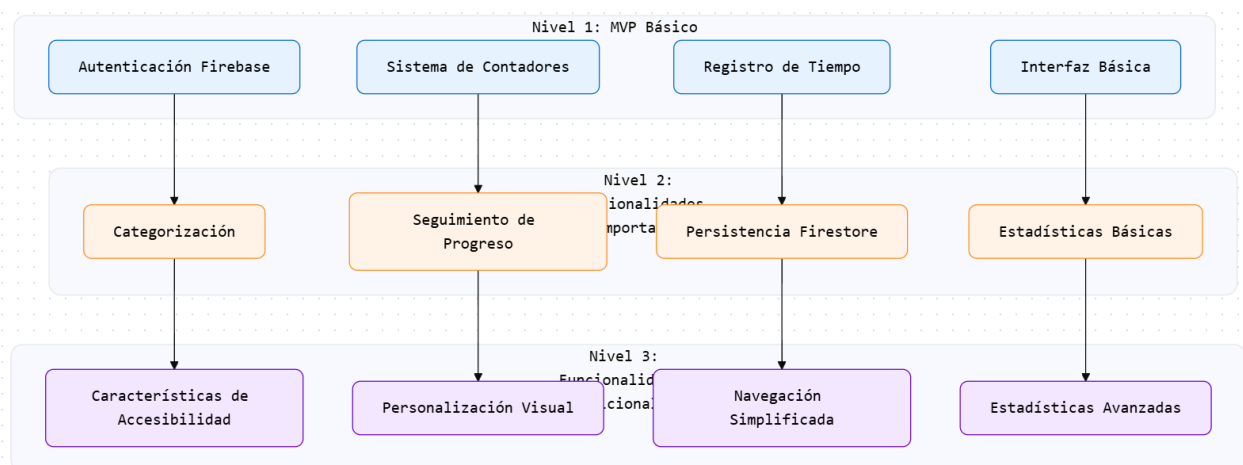
Nivel 2: Funcionalidades Importantes (Se añadirán al MVP según vayamos de tiempo y en orden decreciente)

- Categorización de actividades
- Seguimiento de progreso hacia objetivos
- Persistencia de datos con Firestore
- Estadísticas básicas de uso

Nivel 3: Funcionalidades Adicionales

- Características de accesibilidad
- Personalización de elementos visuales
- Navegación simplificada
- Estadísticas avanzadas

Para visualizar mejor la estructura y relaciones entre componentes, aquí está el diagrama de la planificación propuesta:



Leyenda:

- Los componentes en azul representan el MVP básico, esencial para el lanzamiento inicial.

- Los componentes en naranja son funcionalidades importantes que se implementarán posteriormente.
- Los componentes en morado son funcionalidades adicionales que enriquecerán la aplicación.
- Las flechas indican las dependencias entre componentes, mostrando el orden óptimo de implementación.

Proceso de desarrollo

Empezando con la misma creación del proyecto, hemos ido avanzando con su desarrollo, pese a las notables limitaciones de tiempo disponible, manifestadas por nuestro equipo de trabajo.

A lo largo del proceso, hemos ido realizando diferentes entregas parciales, en las que se mostraba al cliente final los avances realizados y a su vez, se trataba de añadir alguna funcionalidad y/o depurar y mejorar alguna de las ya existentes. Acto seguido se adjuntan los enlaces a los diferentes videos explicativos (mediante *Youtube*) y también hacia el repositorio de Github, en el cual se encuentra alojado nuestro proyecto y donde se puede apreciar el proceso y las diferentes fases o pasos por las que se ha ido pasando durante el proceso.

Videos explicativos e ilustrativos acerca del proceso de desarrollo:

Entrega nº 1: <https://youtu.be/BxLuISZNQYs>

Entrega nº 2: <https://youtu.be/giQe7k7nedg>

Entrega nº 3:




Enlace para acceder al repositorio donde se aloja nuestro proyecto:

https://github.com/kevinzamoraa/Temporis-AndroidApp-2oDAM-2T_PMDM

Diseño de la App

La aplicación **Temporis** está diseñada como una herramienta de gestión de tiempo enfocada en la creación y seguimiento de **temporizadores personalizados**. Inspirada en herramientas como *Toggl Track*, su interfaz es intuitiva y minimalista, facilitando al usuario el control de su tiempo en distintas tareas o actividades.


La estructura de la “app” está dividida en tres secciones principales, accesibles mediante una **barra de navegación inferior**:

-  **Inicio**: Muestra la lista de temporizadores creados y permite/habilita su creación, activación, edición o eliminación.
-  **Dashboard**: Muestra información acerca del/de la usuario/a con el/la que se haya iniciado sesión.
-  **Cerrar sesión**: Enlace destinado al cierre de sesión y desconexión del usuario autenticado.

La arquitectura sigue el patrón **MVVM** (en inglés “*model–view–viewmodel* ”), utilizando **Firebase Authentication** para la gestión de usuarios y **Cloud Firestore** como base de datos en tiempo real.

✅ **Funcionalidades Implementadas**

- 🛡️ **Registro e inicio de sesión con Firebase Authentication:** Los usuarios pueden registrarse e iniciar sesión de manera segura utilizando Firebase Authentication, con soporte para correo electrónico y redes sociales (por ejemplo, Google).
- 👤 **Gestión de usuarios autenticados:** La aplicación gestiona la autenticación del usuario, permitiendo personalizar la experiencia según el usuario autenticado.
- ⌚ **Creación de temporizadores personalizados:** Los usuarios pueden crear temporizadores personalizados para realizar un seguimiento de tareas y actividades específicas.
- ⌛ **Visualización de temporizadores activos con cuenta atrás en tiempo real:** Los temporizadores activos se muestran en tiempo real, con la cuenta atrás visible para que el usuario pueda ver cuánto tiempo queda en cada tarea.
- 🗑️ **Eliminación de temporizadores:** Los usuarios pueden eliminar temporizadores que ya no necesiten o que hayan terminado.
- 🔌 **Sincronización de datos en tiempo real con Firestore:** La app se mantiene sincronizada con Firestore, lo que permite que todos los cambios se reflejen al instante para todos los usuarios, sin necesidad de actualizar manualmente.
- 📱 **Interfaz responsive y adaptada para dispositivos móviles:** La aplicación se adapta automáticamente a diferentes tamaños de pantalla y orientaciones, asegurando una experiencia de usuario óptima en smartphones y tablets.
- 🔔 **Redirección automática al login si no hay sesión iniciada:** Si el usuario no está autenticado al abrir la app, se le redirige automáticamente a la pantalla de inicio de sesión para garantizar la seguridad.
- ⚙️ **Vista de perfil de usuario (en desarrollo):** El usuario puede ver su perfil con la información personal, aunque esta funcionalidad aún se encuentra en fase de desarrollo.
- 🌙 **Modo oscuro (dark mode):** Los usuarios pueden cambiar entre el modo claro y el modo oscuro, ofreciendo una experiencia de usuario más cómoda en entornos con poca luz. Esta funcionalidad puede activarse manualmente a través de los ajustes de la app, o adaptarse automáticamente a la configuración del sistema operativo del dispositivo.
- 🌞 **Modo claro (light mode):** Además del modo oscuro, la app también permite volver al modo claro, ideal para entornos bien iluminados y con menos carga visual.

-  **Sincronización en múltiples dispositivos** (en desarrollo): La aplicación mantiene los datos sincronizados en todos los dispositivos conectados al mismo usuario, gracias a la integración con Firestore.

Dificultades Encontradas y Cómo se Resolvieron

Durante el desarrollo se enfrentaron diversos desafíos técnicos, entre ellos:

1. Sincronización en tiempo real con Firestore

Se resolvió implementando **LiveData** y observadores para reaccionar a los cambios de Firestore, permitiendo que la lista de temporizadores se actualice al instante.

2. Persistencia de sesión al reiniciar la app

Utilizando `FirebaseAuth.getInstance().currentUser`, se logró detectar si el usuario ya estaba autenticado y redirigirlo automáticamente al `MainActivity`.

3. Interacción entre Fragmentos y ViewModels

Se utilizaron **SharedViewModel** y **Navigation Component** para facilitar la navegación y el traspaso de datos entre vistas.

4. Error al cargar y actualizar los datos de usuario desde Firestore

Un error persistente se presentó al intentar cargar o actualizar los datos del perfil de usuario desde Cloud Firestore. El mensaje de error indica **"permission_denied"**, lo que sugiere un problema con las reglas de seguridad de Firestore o con la forma en que los datos están siendo gestionados. Esto impide que los usuarios puedan ver o actualizar sus datos correctamente. Este problema sigue pendiente de solución y será abordado en una futura actualización.

Capturas de Pantalla

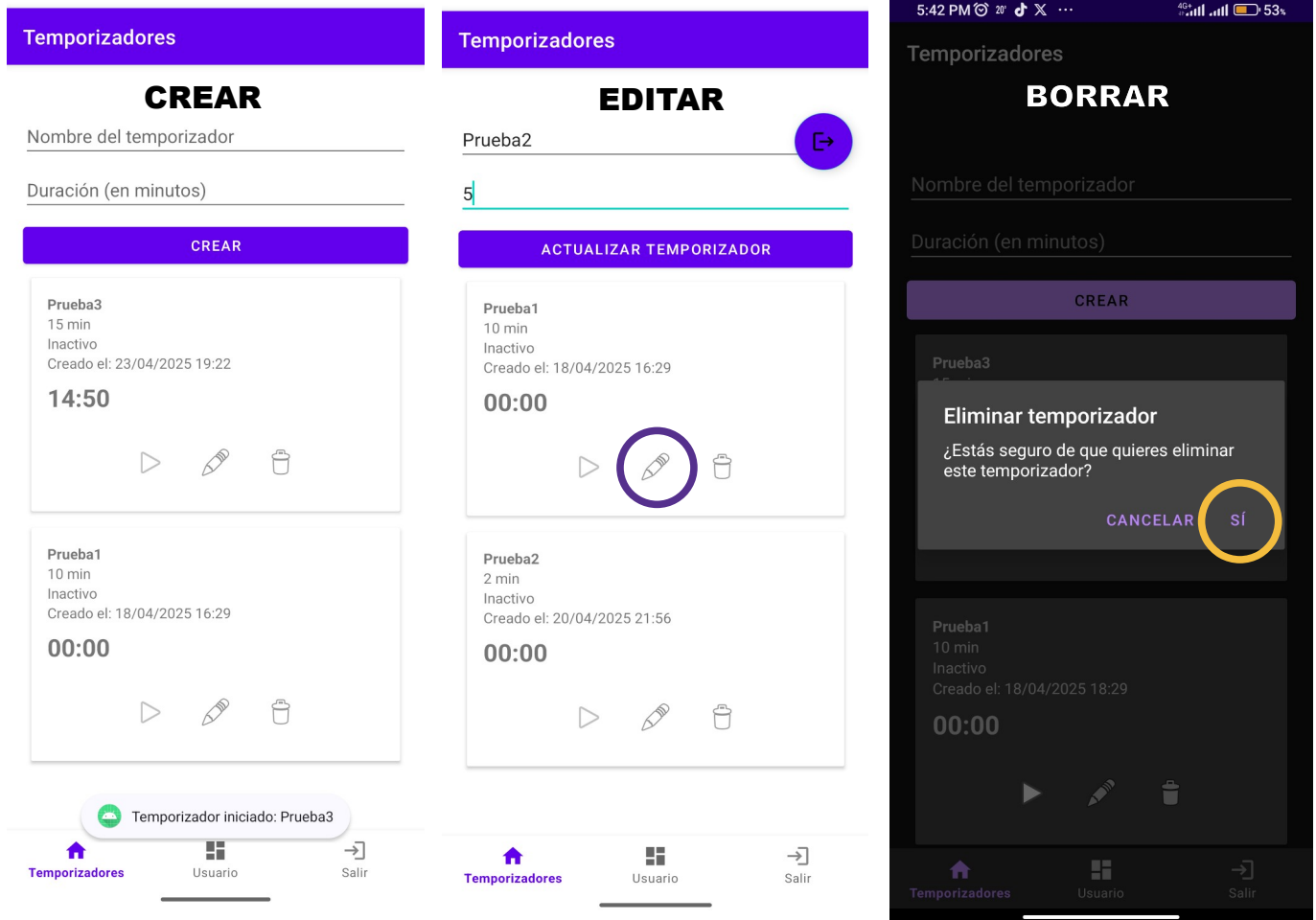
A continuación se muestran algunas pantallas funcionales de la aplicación:

1. Vista principal (Inicio)

Muestra temporizadores activos e inactivos, con opciones para eliminar o activar.

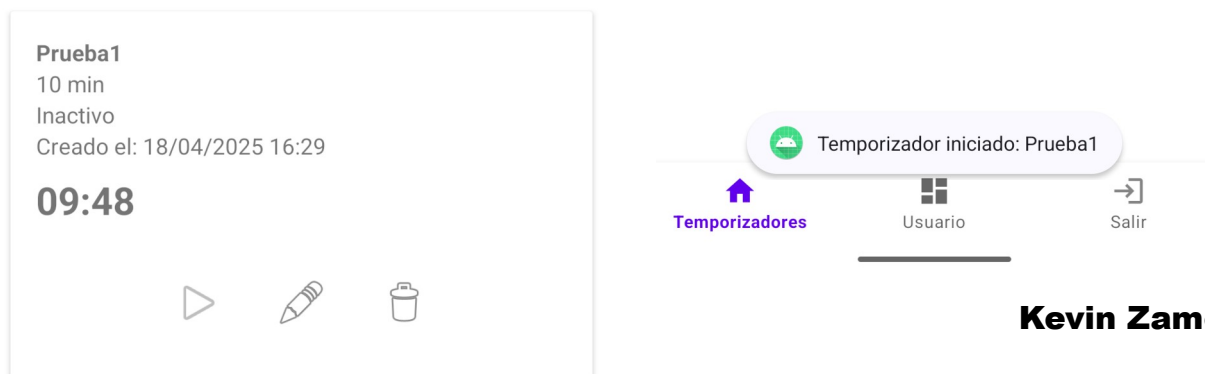
2. Formulario de nuevo temporizador

Permite definir nombre, duración y tipo de temporizador.



3. Cuenta atrás activa



Cada temporizador activo muestra una cuenta atrás en tiempo real dentro de una CardView.



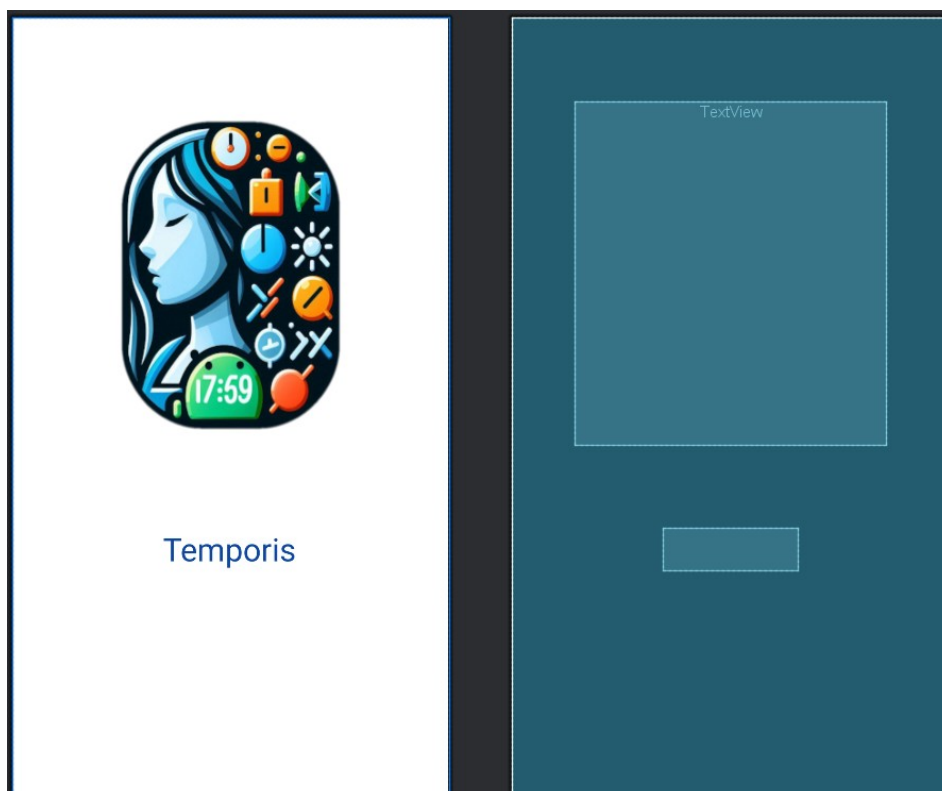
Kevin Zamora



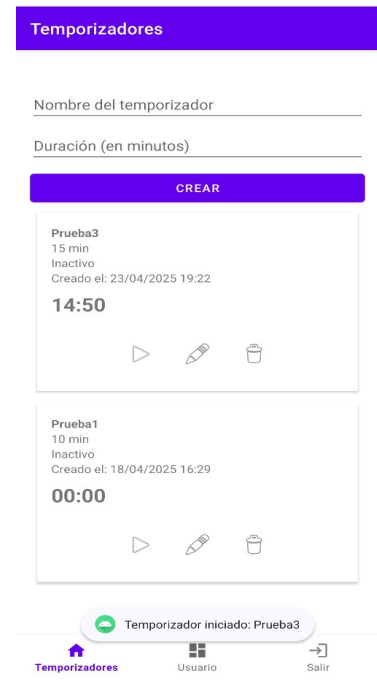
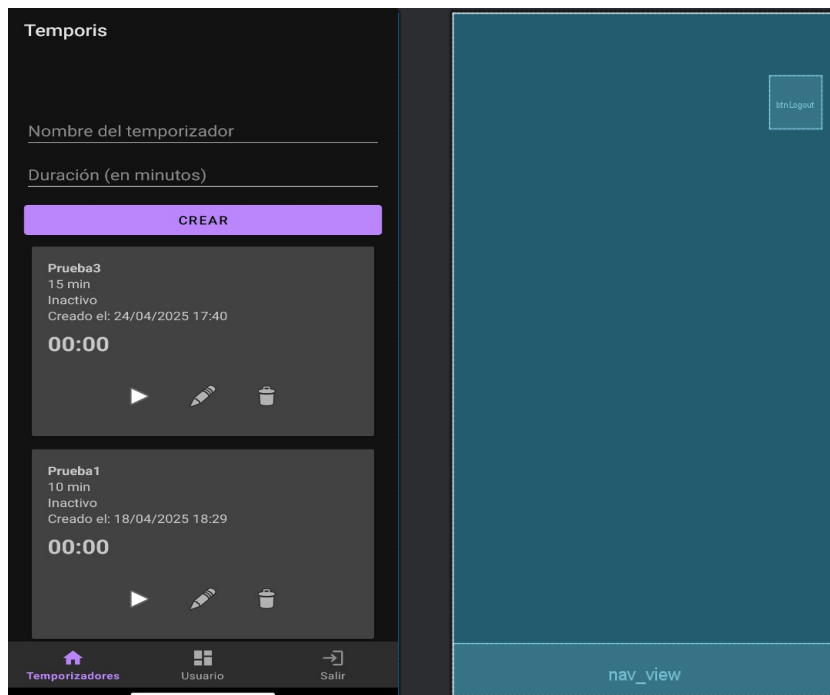
- **Login / Inicio de sesión y Registro:**

Temporis	Temporis
	
Email	Usuario
Contraseña	Email
Olvidé mi contraseña	Contraseña
¿Aún no tienes cuenta?	Confirmar contraseña
ENTRAR	REGISTRAME
Entrar con Google	

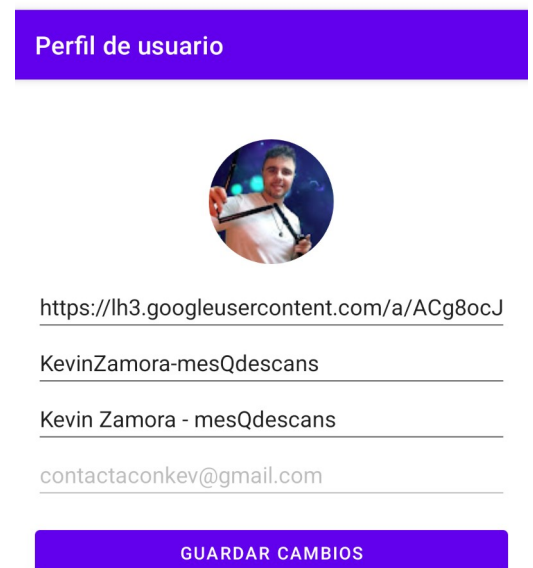
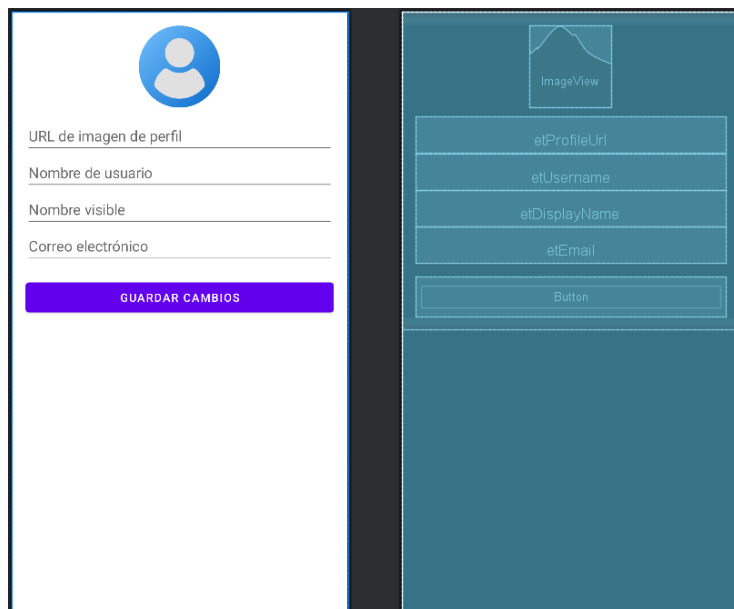
- **Splash Screen / Pantalla de bienvenida:**



- Inicio / Vista principal



- Dashboard / Perfil de usuario:



Conclusión

Temporis representa una base sólida para una app de productividad orientada al tiempo, con una estructura robusta y sincronización efectiva con Firebase. A pesar del obstáculo técnico con la función de cierre de sesión, el resto de funcionalidades clave han sido implementadas con éxito y están completamente operativas. En cuanto a seguir ampliando con más

Kevin Zamora

funcionalidades o seguir depurando y mejorando las actuales, estas quedan a la espera de poderlas seguir implementándolas y desarrollándolas en un futuro más o menos cercano; por ejemplo durante el desarrollo del proyecto final del presente ciclo formativo de grado superior.