UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

BreathBank

Presentado por Eduardo García de Leániz Juarros
en Universidad de Burgos — 6 de abril de 2025
Tutor: Ana Serrano Mamolar

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. nombre tutor, profesor del departamento de nombre departamento, área de nombre área.

Expone:

Que el alumno D. Eduardo García de Leániz Juarros, con DNI 71482319K, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado BreathBank.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 6 de abril de 2025

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
D. nombre tutor	D. nombre co-tutor

Resumen

En este primer apartado se hace una **breve** presentación del tema que se aborda en el proyecto.

Descriptores

Palabras separadas por comas que identifiquen el contenido del proyecto Ej: servidor web, buscador de vuelos, android ...

Abstract

A **brief** presentation of the topic addressed in the project.

Keywords

keywords separated by commas.

Índice general

Índice de figuras

Índice de tablas

1. Introducción

Escribir Introducción. Posible apartado > casos de uso

Opcion 1

En un mundo marcado por el estrés, las preocupaciones cotidianas y un ritmo de vida cada vez más acelerado, la importancia de cuidar nuestra salud física y mental se ha vuelto más relevante que nunca. En este contexto, las prácticas de respiración, que han sido utilizadas desde tiempos antiguos en diversas culturas y disciplinas, han demostrado ser una herramienta poderosa para mejorar tanto la capacidad pulmonar como el bienestar general. Sin embargo, no siempre es fácil acceder a estas prácticas de manera rápida, constante y guiada.

La tecnología, con su constante avance, nos brinda una oportunidad única para integrar hábitos saludables en nuestra rutina diaria de una forma accesible y personalizada. Las aplicaciones móviles se han convertido en aliados de la salud, permitiendo a los usuarios ejercitar su cuerpo y mente desde la palma de su mano. Es aquí donde surge la idea de crear una aplicación centrada en ejercicios de respiración para mejorar la capacidad pulmonar y promover la relajación, dos aspectos clave para mantener una vida equilibrada y saludable.

Este proyecto busca no solo proporcionar una herramienta práctica para la mejora física y emocional, sino también demostrar el impacto positivo que la tecnología puede tener en el cuidado de la salud personal. A través de ejercicios de respiración guiados, la app pretende ayudar a los usuarios a reducir el estrés, aumentar su capacidad pulmonar y disfrutar de momentos de calma y concentración en medio de un entorno que cada vez demanda más de nosotros. Con esta propuesta, la tecnología no es solo un medio, sino una solución efectiva para fomentar el bienestar integral.

Opcion 2

En un mundo cada vez más acelerado, donde el estrés y la ansiedad se han convertido en compañeros constantes de muchas personas, la búsqueda de herramientas que favorezcan el bienestar personal se ha intensificado. La respiración, una de las funciones más fundamentales del cuerpo humano, ha demostrado ser una poderosa aliada no solo para mantenernos vivos, sino también para mejorar nuestra salud mental y física. Sin embargo, en medio de la rutina diaria, es fácil perder de vista su importancia y, en ocasiones, incluso olvidamos cómo respirar de manera eficiente.

Las técnicas de respiración, ampliamente utilizadas en disciplinas como el yoga, la meditación o la terapia respiratoria, son métodos efectivos para reducir el estrés, mejorar la concentración y aumentar la capacidad pulmonar. Sin embargo, muchas personas no saben por dónde empezar o cómo incorporar estas prácticas de manera accesible y práctica en su vida cotidiana. Es aquí donde la tecnología puede jugar un papel transformador.

Este trabajo de fin de grado presenta una solución innovadora: una aplicación móvil que ofrece ejercicios de respiración guiada con el objetivo de mejorar la capacidad pulmonar y promover la relajación. A través de una plataforma accesible y sencilla, los usuarios podrán incorporar rutinas de respiración adaptadas a sus necesidades y horarios. La importancia de este proyecto radica en su capacidad para conectar a las personas con herramientas simples pero poderosas que pueden marcar una diferencia significativa en su calidad de vida, promoviendo no solo el bienestar físico, sino también emocional.

El creciente interés por la salud mental, la reducción del estrés y el autocuidado hace que la creación de este tipo de aplicaciones sea más relevante que nunca. En esta memoria se exploran las bases teóricas que sustentan la eficacia de los ejercicios de respiración, el diseño de la aplicación, sus funcionalidades y la manera en que puede contribuir de manera efectiva al bienestar integral de los usuarios.

1.1. Estructura

Descripción del contenido del trabajo y del estructura de la memoria y del resto de materiales entregados.

Párrafo 2: objetivo de crear una app para móviles + lenguaje + framework + para qué sirve la app

Párrafo 3: estructura de la memoria + características + buenas prácticas (ventajas)

Subapartado “Estructura” -> “Memoria” y “Anexos”

Estructura de la memoria

Estructura de la memoria

- Introducción
- Objetivos del proyecto
- Conceptos teóricos
- Técnicas y herramientas
- Aspectos relevantes del desarrollo del proyecto
- Trabajos relacionados
- Conclusiones y líneas de trabajo futuras

Estructura del anexo

Estructura del anexo

- Plan de Proyecto
- Requisitos
- Diseño
- Manual de Usuario
- Manual de Programador

2. Objetivos del proyecto

Este apartado explica de forma precisa y concisa cuales son los objetivos que se persiguen con la realización del proyecto. Se puede distinguir entre los objetivos marcados por los requisitos del software a construir y los objetivos de carácter técnico que plantea a la hora de llevar a la práctica el proyecto.

Objetivos del software + beneficios del usuario + objetivos técnicos

2.1. Objetivos del software

2.2. Beneficios

2.3. Objetivos técnicos

3. Conceptos teóricos

En aquellos proyectos que necesiten para su comprensión y desarrollo de unos conceptos teóricos de una determinada materia o de un determinado dominio de conocimiento, debe existir un apartado que sintetice dichos conceptos.

3.1. Conceptos sobre el tema de la aplicación

Inversor

Usuario de la aplicación BreathBank y persona que destina un recurso valioso, en este caso su tiempo, a la práctica de respiraciones conscientes con el objetivo de obtener beneficios medibles en su bienestar físico y mental. Este concepto se fundamenta en la idea de que la inversión de tiempo en esta actividad genera un retorno positivo en términos de calma, serenidad y paz mental, respaldado por estudios científicos.

Respiración consciente

Es aquella en la que nuestra mente y nuestros pensamientos están enfocados en la propia respiración. La intención puede ir desde únicamente ser consciente de nuestra respiración a modificarla también conscientemente aumentando o disminuyendo la intensidad y la duración de la inspiración y de la espiración o la duración de las pausas entre ellas.

Ciclos y series respiratorias

Ciclo respiratorio (C.R): realización de 3 respiraciones conscientes consecutivas.

Serie respiratoria (S.R): realización de 3 ciclos respiratorios (C.R) consecutivos.

Tipos de respiraciones según musculos implicados

- **Respiración abdominal**: Solo se contrae el músculo diafragma y se expande principalmente la mitad inferior o zona abdominal. Es la respiración más habitual y que hacemos de manera involuntaria.

- **Respiración torácica**: Además del diafragma, se contraen otros músculos accesorios que provocan una mayor expansión de la caja torácica, quedando la zona abdominal sin apenas expansión o incluso retrayéndose. (¿Incluir respiración torácica pasiva?)

- **Respiración global o integral**: Resulta de la unión de los dos tipos de respiración anteriores. Como resultado, se produce una expansión de ambas cavidades, abdominal y torácica, ya sea de forma simultánea o secuencial.

Tipos de inversores

como lo explico?

Niveles de práctica

Cada nivel viene determinado por la capacidad del inversor de respirar de forma más lenta, buscando aumentar el tiempo que puede inspirar y espirar de forma continuada sin que aparezca tensión en el cuerpo ni en la mente. Estos están diseñados para adaptarse progresivamente a las capacidades respiratorias de cada inversor, que irá subiendo de nivel en base a la experiencia que vaya adquiriendo y a la mejora en su práctica diaria.

Tipo de inversiones

Manual o guiada por audio.

Listón de la inversión

Explicar listón de la inversión

3.2. Conceptos sobre el software/IDE/Framework

4. Técnicas y herramientas

Esta parte de la memoria tiene como objetivo presentar las técnicas metodológicas y las herramientas de desarrollo que se han utilizado para llevar a cabo el proyecto. Si se han estudiado diferentes alternativas de metodologías, herramientas, bibliotecas se puede hacer un resumen de los aspectos más destacados de cada alternativa, incluyendo comparativas entre las distintas opciones y una justificación de las elecciones realizadas. No se pretende que este apartado se convierta en un capítulo de un libro dedicado a cada una de las alternativas, sino comentar los aspectos más destacados de cada opción, con un repaso somero a los fundamentos esenciales y referencias bibliográficas para que el lector pueda ampliar su conocimiento sobre el tema.

Android Studio como IDE de desarrollo del software. Cita: Cómo descargar Android Studio y App Tools - Android Developers. (s. f.). Android Developers. https://developer.android.com/studio?hl=es-419

Visual Studio Code como alternativa IDE de desarrollo del software. Cita: Visual Studio Code - Code editing. Redefined. (2021, 3 noviembre). https://code.visualstudio.com/

Flutter como SDK para la aplicación. Cita: Flutter - Build apps for any screen. (s. f.). https://flutter.dev/

Firebase como posible plataforma para autenticación de usuarios de la aplicación. Cita: Firebase. (s. f.). Firebase. https://firebase.google.com/?hl=es-419

4.1. Framework y lenguaje de programación

Flutter

En el proceso de desarrollo de BreathBank, se evaluaron diversas opciones de frameworks que permitieran construir la aplicación para múltiples plataformas de manera eficiente. Sin embargo, se decidió optar por **Flutter** por una serie de características y ventajas que se comentan a continuación.

- **Desarrollo Multiplataforma con un solo código base:** Una de las principales razones para elegir Flutter es su capacidad para desarrollar aplicaciones para múltiples plataformas (Android, iOS) con una sola base de código. Esto no solo reduce significativamente el tiempo de desarrollo, ya que no es necesario duplicar esfuerzos para adaptarse a diferentes sistemas operativos.

  Flutter permite escribir el código una vez y ejecutarlo en ambas plataformas sin comprometer la funcionalidad o la experiencia del usuario, lo que es una ventaja clara frente a soluciones que requieren mantener bases de código separadas para cada plataforma.

- **Alto rendimiento en desarrollo y compilación:** A diferencia de otros frameworks que dependen de un puente o capas intermedias para comunicar el código con los sistemas operativos nativos, Flutter compila directamente a código nativo. Esto significa que las aplicaciones desarrolladas con Flutter tienen un rendimiento muy cercano al de aplicaciones nativas. Además, funciona de manera bastante rápida a la hora de mostrar los cambios en el dispositivo o en el emulador que se esté probando, aunque esto se hablará también en el apartado de Hot Reload.

- **Control sobre la UI y widgets personalizados:** Flutter se basa en un sistema de widgets para la creación de interfaces de usuario. Cada componente de la interfaz, desde botones hasta animaciones, es un widget, lo que ofrece un control totalmente personalizado sobre el diseño y el comportamiento de la aplicación.

- **Hot Reload: Desarrollo Ágil y Eficiente:** Una de las características más valoradas por los desarrolladores de Flutter es el Hot Reload. Esta funcionalidad permite a los programadores ver los cambios en el código en tiempo real sin necesidad de reiniciar la aplicación. Esto acelera enormemente el proceso de desarrollo y permite realizar pruebas e

iteraciones rápidas sobre el diseño y la funcionalidad sin interrumpir el flujo de trabajo.

El Hot Reload mejora también la productividad en el proceso de depuración, ya que facilita la corrección de errores y la optimización de la aplicación de forma inmediata. Este enfoque ágil es ideal para el desarrollo iterativo de características y mejora de la experiencia de usuario.

- **Fácil Integración con Firebase:** Flutter destaca también por su fácil integración con Firebase, plataforma la cual he elegido para guardar los datos de la aplicación. Firebase ofrece múltiples herramientas para gestionar la autenticación, bases de datos en tiempo real, almacenamiento de archivos y funciones en la nube, se verán en más detalle en el apartado de Bases de Datos. La integración con Firebase en Flutter se realiza de manera sencilla gracias a los paquetes oficiales proporcionados por el equipo de Flutter, lo que permite a los desarrolladores incorporar funcionalidades avanzadas sin tener que gestionar una infraestructura de backend compleja.

Dart

Dart es un lenguaje de programación optimizado para aplicaciones móviles, web y de escritorio. Fue desarrollado por Google con el objetivo de proporcionar una plataforma eficiente para la creación de aplicaciones modernas. Dart es conocido por ser el lenguaje que utiliza el framework Flutter para el desarrollo de aplicaciones.

- **Sintaxis moderna y familiar:** Dart tiene una sintaxis que se asemeja a otros lenguajes como JavaScript, C++ y Java, lo que facilita su aprendizaje para programadores con experiencia en estos lenguajes.

- **Orientado a objetos:** Dart es un lenguaje orientado a objetos, lo que significa que utiliza clases y objetos para organizar el código y la lógica de las aplicaciones.

- **Asincronía y Concurrencia:** Dart proporciona soporte para programación asincrónica mediante async y await, facilitando la escritura de código no bloqueante y optimizando el rendimiento en tareas concurrentes. Esto es muy útil a la hora de implementar sistemas de autenticación o en operaciones de lectura o escritura en base de dastos.

- **Tipado estático:** Dart permite un tipado estático opcional, lo que ayuda a detectar errores en tiempo de compilación y mejora la mantenibilidad del código.

4.2. IDE, Bibliotecas y Extensiones

La elección del IDE ha cambiado a lo largo del desarrollo de este proyecto. Primero comencé utilizando Android Studio ya que pensé que quizá estaría un poco más integrado con el trabajo con aplicaciones móviles. Sin embargo, el no tener experiencia utilizándolo y el bajo rendimiento que daba en mi equipo con el emulador y a la hora de hacer debug, me hizo cambiar de opinión e inclinarme por **Visual Studio Code**.

Este editor de código fuente me funcionaba mucho mejor, además de que ya tenía experiencia trabajando con él en proyectos anteriores. Además, para trabajar con Flutter y Firebase solo eran necesarias instalar 3 extensiones: **Flutterr, Dart y Awesome Flutter Snippets**. Otras partes de la configuración como es la instalación de Flutter y Firebase, y la integración del proyecto de Firebase en Flutter si que fueron bastante más complejas, pero eso se detallará en su apartado correspondiente.

4.3. Base de datos

Uno de los pilares fundamentales dentro de el desarrollo de aplicaciones es **la gestión y el guardado de los datos** que vaya a utilizar la misma. Por ello es importante estudiar las opciones que hay e identificar cual se adapta mejor a cada caso en función de sus ventajas e inconvenientes.

Comparación entre distintos SGBD

En el mundo del desarrollo de aplicaciones, elegir el sistema de gestión de bases de datos (SGDB) adecuado es crucial para garantizar un rendimiento óptimo y una buena experiencia de usuario. A día de hoy existen múltiples opciones a valorar en función de su arquitectura, escalabilidad y otras características que se han tenido en cuenta para elegir el gestor adecuado.

Las **bases de datos relacionales** ofrecen muchas ventajas, tienen una estructura muy definida, permiten realizar consultas más complejas de manera eficiente y ofrecen una mayor integridad y consistencia de los datos. Sin embargo, estas son características que en el contexto de este proyecto no se iban a aprovechar al máximo. Por ello, ciertos SGDB como MySQL,

Característica	**BD Relacionales**	**BD No Relacionales**
Escalabilidad	↓	↑
Consistencia de datos (ACID)	↑	↓
Flexibilidad de esquema	↓	↑
Rendimiento en consultas simples	↓	↑
Consultas complejas (JOIN, GROUP BY, etc.)	↑	↓
Soporte para datos jerárquicos	↓	↑
Transacciones complejas	↑	↓
Manejo de datos offline (sin conexión)	↓	↑
Rendimiento en grandes volúmenes de datos	↓	↑
Manejo de relaciones complejas	↑	↓
Fácil de usar para desarrolladores	↑	↑
Manejo de datos en tiempo real	↓	↑
Costo de infraestructura	↓	↑

Tabla 4.1: BD Relacionales VS BD No Relacionales

PostgreSQL, Oracle, SQLite fueron descartados como opciones para crear la base de datos de BreathBank.

Por otro lado, las **bases de datos no relacionales** ofrecían algunas ventajas que sí podían ser de mayor provecho para el contexto de desarrollo de una aplicación para móviles: mayor flexibilidad en el modelo de datos y facilidad para manejarlo, mayor capacidad de adaptación a cambios en los datos, mejor manejo de operaciones de lectura y escritura en tiempo real, mayor tolerancia a fallos y disponibilidad, etc. Pero sobre todo, la mayor ventaja que ofrecían en este caso, era una mayor facilidad a la hora de integrarlas con la aplicación y de conectarlas con las herramientas de desarrollo que estaba utilizando.

En concreto **Firebase** es una plataforma que ofrecía todas las herramientas para cubrir las necesidades de backend que requería la aplicación: autenticación de usuarios, almacenamiento de datos, comunicación entre frontend y backend, notificaciones, etc. y además está muy integarada con Flutter, el entorno de desarrollo que estoy utilizando

Firebase

Firebase es una plataforma de desarrollo de aplicaciones móviles y web ofrecida por Google, que proporciona una variedad de servicios backend para

ayudar a los desarrolladores a construir, gestionar y escalar aplicaciones de manera eficiente.

Servicios de Firebase Revisar!

- **Firestore y Realtime Database:** Bases de datos NoSQL escalables que permiten almacenar y sincronizar datos en tiempo real entre todos los usuarios de la aplicación.
- **Autenticación:** Servicio sencillo para gestionar el registro, inicio de sesión y autenticación de usuarios mediante múltiples proveedores como correo electrónico, Google, Facebook, etc.
- **Cloud Functions:** Funciones serverless que permiten ejecutar lógica personalizada sin necesidad de gestionar servidores.
- **Notificaciones Push:** Envío de notificaciones en tiempo real a los usuarios.
- **Firebase Analytics:** Herramienta de análisis para comprender el comportamiento de los usuarios dentro de la aplicación.

Firestore DB vs Realtime DB

Tanto Realtime Database como Firestore Database son **bases de datos NoSQL en tiempo real** proporcionadas por **Firebase**, lo que significa que ambas permiten una sincronización instantánea de datos entre el cliente y el servidor. Ambas están optimizadas para aplicaciones móviles y web, con una infraestructura gestionada que facilita la **escalabilidad**. En cuanto a su estructura, ambas ofrecen almacenamiento flexible de datos (JSON en Realtime Database y documentos en Firestore), y permiten la **actualización en tiempo real de los datos**, lo que es fundamental para aplicaciones de este tipo. Además, las dos bases de datos están integradas dentro del ecosistema de Firebase, lo que facilita su utilización con otros servicios de la plataforma, como la autenticación y las notificaciones.

Las **principales diferencias** entre ambas radican en varias áreas clave. Realtime Database organiza la información en forma de un **árbol JSON**, mientras que Firestore emplea una estructura de datos basada en **colecciones y documentos**, lo que proporciona una organización más flexible y escalable. Respecto a las consultas, Firestore permite realizar búsquedas complejas y optimizadas a través de índices, lo que permite mayor eficiencia, mientras que Realtime Database dispone de un sistema más básico y limitado

Aspecto	**Firestore DB**	**Realtime DB**
Arquitectura	Único árbol JSON	Colecciones y documentos
Consultas	Simples, sin índices complejos	Avanzadas con índices automáticos
Escalabilidad	Menor	Mejor rendimiento a gran escala
Sinc. en Tiempo Real	Menos eficiente	Más eficiente
Manejo de Datos Offline	Soporte básico	Sinc. automática al reconectar
Latencia	Mayor (con vol. grandes)	Menor
Seguridad	Reglas globales	Reglas por documento y colección
Rendimiento general	Medio	Alto

Tabla 4.2: Firestore DB VS Realtime DB

para este tipo de operaciones. En términos de actualización en tiempo real, aunque ambos servicios soportan la sincronización instantánea de datos, Firestore maneja de manera más eficiente conexiones en segundo plano y operaciones con grandes volúmenes de datos, aunque esto no es de gran importancia en este caso ya que la aplicación no va a llegar a un volumen de datos tan grande como para que esto sea un factor relevante. Además, en cuanto a la **capacidad de funcionar sin conexión**, Firestore ofrece un manejo más robusto de datos cuando el dispositivo está sin conexión, permitiendo que la aplicación se **sincronice automáticamente cuando se restablezca la conexión**. Por último, Firestore ofrece un control de acceso más detallado, permitiendo **reglas de seguridad específicas** para cada documento y colección, mientras que Realtime Database trabaja con **reglas de acceso más generales**.

Teniendo en cuenta todo esto, he considerado que Firestore era más adecuado para este tipo de aplicación debido a la flexibilidad que me ofrecía en el modelo de datos, la fácil organización de los mismos, y la mayor capacidad de realizar consultas y devolver datos de manera más rápida. Además, su capacidad de trabajar de manera eficiente en modo offline permitiendo que la aplicación siga funcionando sin conexión y se sincronice de manera automática cuando se restablezca la conexión me parecía de gran utilidad.

Ventajas de usar Firebase

- Múltiples herramientas integradas en una
- Servicio de autenticación

- Bases de datos flexibles y en tiempo real
- Fácil integración con Flutter
- Permite trabajar con múltiples plataformas
- Escalabilidad automática

5. Aspectos relevantes del desarrollo del proyecto

Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, comentados por los autores del mismo. Debe incluir desde la exposición del ciclo de vida utilizado, hasta los detalles de mayor relevancia de las fases de análisis, diseño e implementación. Se busca que no sea una mera operación de copiar y pegar diagramas y extractos del código fuente, sino que realmente se justifiquen los caminos de solución que se han tomado, especialmente aquellos que no sean triviales. Puede ser el lugar más adecuado para documentar los aspectos más interesantes del diseño y de la implementación, con un mayor hincapié en aspectos tales como el tipo de arquitectura elegido, los índices de las tablas de la base de datos, normalización y desnormalización, distribución en ficheros3, reglas de negocio dentro de las bases de datos (EDVHV GH GDWRV DFWLYDV), aspectos de desarrollo relacionados con el WWW... Este apartado, debe convertirse en el resumen de la experiencia práctica del proyecto, y por sí mismo justifica que la memoria se convierta en un documento útil, fuente de referencia para los autores, los tutores y futuros alumnos.

15/03/2025 He decidido cambiar el IDE de desarrollo de la app de Android Studio a Visual Studio Code, ya que me parece una herramienta mucho más sencilla, intuitiva y eficiente para trabajar en un proyecto a largo plazo.

Creación del proyecto en Visual Studio Code: View -> Command Template -> Flutter: New Project

Creación template para el emulador en Visual Studio Code: View -> Command Template -> Flutter: Launch Emulator

6. Trabajos relacionados

Explicar trabajos relacionados.

Trabajos BreathBank años anteriores

7. Conclusiones y líneas de trabajo futuras

Explicar conclusiones y líneas de trabajo futuras.

Conclusiones

Lineas de trabajo futuras

Añadir idiomas, interfaz oscuro/claro, nuevos ejercicios de respiración, más metodos de acceso/registro de cuentas, dispositivos de medición

Bibliografía