

Move Up

Miguel Angel Colmenero Palacios

Junio 2023

Introducción

Datos del proyecto

Nombre	Apellidos	Título	Ciclo	Año	Centro educativo
Miguel Angel	Colmenero Palacios	MoveUp	DAM	2023	IES Virgen del Carmen

Primeras ideas del proyecto

En un principio, estás fueron las primeras ideas que tuve:

- La idea principal de mi proyecto es desarrollar una aplicación híbrida con Ionic.
- La aplicación tratará de ser una red social, con una interfaz parecida a la de TikTok, Instagram BeReal... , en la que puedas agregar amigos, tener tus publicaciones, interactuar con amigos (la idea es la realización de un chat pero como todavía no sé la dificultad que puede llegar a tener esta idea se puede eliminar en un futuro), etc. La diferencia de mi aplicación en cuanto a las demás es que ésta estará enfocada a la realización de ejercicio físico.
- Se presentarán desafíos dentro de la aplicación que se deben realizar compartiendo un video de que se han realizado (por ejemplo se presenta un desafío de hacer 50 flexiones en los próximos 10 minutos). Tus amigos o personas externas pueden decidir si el desafío ha sido superado o no.
- Se obtendrán puntos dentro de la aplicación que serán acumulables y se podrán canjear por recompensas. Por ejemplo, si se tratara de una aplicación real y contara con el patrocinio de marcas como Myprotein, se podrían obtener recompensas como una camiseta o un suplemento al alcanzar un número de puntos dentro de la aplicación.
- Habrá un ranking semanal/mensual del que más puntos ha obtenido entre amigos y globalmente y los primeros obtendrán puntos dentro de la

aplicación.

- Se podrán compartir, además de publicaciones, rutinas y los demás usuarios podrán interactuar con ellas, escribir comentarios, compartirlas, etc. Además también se podrían publicar quedadas en un lugar concreto dentro de la aplicación.
- Se puede tener un seguimiento del progreso dentro de la aplicación (objetivos, peso, marcas). Esto podría realizarse o no, pero en caso de hacerlo sería de forma muy general ya que la aplicación no está orientado a esto.
- Se podrán realizar desafíos entre los usuarios de la aplicación en el que se notifique al otro usuario de que le han retado (puede haber recompensas de puntos por la victoria o derrota del desafío, aceptar o rechazar el desafío, un tiempo límite para realizarlo, etc).
- Al tratarse una red social, dispondrá de un login y de un menú inferior en el que se podrán acceder a todas las partes de la aplicación con un estilo parecido a aplicaciones como Youtube, Instagram, TikTok, etc. ya que he observado que actualmente todas las aplicaciones actuales cuentan con uno.

Descripción del proyecto

Cómo más adelante se puede apreciar claramente en los diagramas de casos de uso, las ideas que al final han sido definitivas para el desarrollo de mi aplicación han sido muchas menos que las que tenía al principio.

Mi aplicación consta de una interfaz bastante parecida a Instagram en la que se pueden realizar las siguientes funcionalidades: * La aplicación consta de un login y un registro contra Firebase Authentication y un usuario que se setea en el contexto una vez hecho login. * Cuenta con un menú de tabs en el que se pueden acceder a todas las partes de la aplicación entre las que se encuentran: * Home: Se visualizará todo el contenido que se suba en la aplicación * Explore: Se podrán buscar por el nombre de usuario todos los usuarios registrados en la aplicación * New: Se subirá contenido a través de la cámara o alguna imagen del dispositivo * Friends: Se mostrarán solo los posts de las personas que sigues y se podrá filtrar por cada una de ellas * Myprofile: Se podrá acceder al perfil del usuario logueado

- Se puede visitar el perfil de cualquier usuario que esté utilizando la aplicación
- Se puede comentar en el post de cualquier usuario
- Se puede añadir un like o “dar me gusta” a cualquier post
- Se pueden guardar los posts a los que posteriormente se tendrá acceso en el perfil de dicho usuario. También serán visibles para las demás personas que utilicen la aplicación.
- Se puede acceder a tus seguidores y seguidos y a los de los demás usuarios en el correspondiente perfil de cada uno
- Las publicaciones cuentan con un menú donde se puede acceder a una

nueva ventana y obtener información sobre la cuenta, además de que si son del propio usuario, se pueden eliminar.

Planificación

Esta sería aproximadamente la planificación del proyecto:

- Semana del 27 al 31 de marzo
 - Organización y fases del trabajo (planificación, recursos que podría necesitar, viabilidad...). Tecnologías que queremos usar...
 - Ideas iniciales sobre el mismo (los “deseos” o “características” de SCRUM)
 - Elementos del proyecto (qué pretendemos generar: un manual o tutorial, un producto software..., la memoria y la presentación)
 - Preparamos la plantilla del documento a entregar, o, en su defecto el repositorio vacío para empezar a documentar en Markdown
- Semana del 10 al 14 de abril
 - Título
 - Objetivos iniciales (convertimos los deseos o características en requisitos)
 - Breve resumen del mismo (esta semana debe estar subido al repositorio en nuestro GitLab)
 - Investigación de estudios y proyectos similares (lo que llamamos “estado del arte”)
- Semana del 17 al 21 de abril
 - Investigación de estudios y proyectos similares (plasmarlo en la documentación)
 - Introducción (generar este apartado en la documentación)
 - Objetivos definitivos (generar este apartado en la documentación), exactamente qué estamos haciendo
 - Material y recursos a utilizar (recoger detalladamente todos los recursos que se disponen y/o necesitarán en la documentación)
- Semana del 24 al 28 de abril
 - Métodos seguidos en el proceso, metodologías, tecnologías (ej. porque usar un lenguaje o framework concreto y no otro)
- Semana del 3 al 5 de mayo
 - Resultados iniciales (primeros “bocetos” del programa)
 - Analizar si hace falta cambiar algo de los requisitos o tecnologías inicialmente planificadas y explicar si hay algún cambio porque se ha hecho. Esto se plasma en la documentación
- Semana del 8 al 19 de mayo (dos semanas)
 - Resultados intermedios (demo funcional)
 - Primera revisión de la documentación para ver que estén todos los puntos necesarios
- Semana del 22 al 26 de mayo
 - Resultados finales (proyecto terminado: tutorial, aplicación...)

- Segunda revisión del documento donde ya estén todos los apartados necesarios
- Preparación de la presentación
- Semana del 29 de mayo al 9 de junio
 - Pulimos los posibles “bugs”
 - Entrega del documento final
- Semana del 12 al 16 de junio
 - Organización de la presentación
 - Entrega de la presentación para la exposición
- Semana del 19 al 22 de junio
 - Presentación de proyectos

Tecnologías

Ionic



Figura 1: Imágen de Ionic

Ionic es un framework de desarrollo de aplicaciones móviles híbridas que ofrece grandes ventajas. Estas son algunas de ellas:

- Desarrollo multiplataforma: Con Ionic, puedes crear una aplicación y ejecutarla en iOS, Android y la web sin tener que escribir código separado para cada plataforma. Esto ahorra tiempo y recursos.
- Fácil de usar: Ionic se basa en tecnologías web como HTML, CSS y JavaScript, lo que facilita su aprendizaje y uso, especialmente para aquellos familiarizados con el desarrollo web. Además, cuenta con una amplia documentación y una comunidad activa para brindar soporte.
- Diseño atractivo: Ionic ofrece una amplia biblioteca de componentes y estilos predefinidos que permiten crear interfaces de usuario atractivas y modernas. Puedes lograr una apariencia y experiencia de usuario consistentes en diferentes plataformas, siguiendo las mejores prácticas de diseño de aplicaciones móviles.

- Rendimiento optimizado: Ionic utiliza tecnologías como React, lo que le permite aprovechar al máximo el hardware del dispositivo y ofrecer un rendimiento rápido y fluido. Además, el uso de WebView nativo mejora el rendimiento en comparación con enfoques basados solo en HTML5.
- Amplia integración: Ionic se integra fácilmente con herramientas y servicios populares. Puedes utilizar Firebase para la autenticación y el almacenamiento en la nube, Cordova para acceder a características nativas del dispositivo y plugins que agregan funcionalidades adicionales a tu aplicación.
- Actualizaciones sencillas: Mantener tus aplicaciones actualizadas es fácil con Ionic. El framework ofrece un mecanismo de actualización rápido y sencillo, lo que te permite aprovechar las últimas características y correcciones de errores en un entorno tecnológico en constante evolución.

En resumen, Ionic es una solución completa y eficiente para el desarrollo de aplicaciones móviles híbridas. Gracias a esto, me permite desarrollar para múltiples plataformas, a la vez que ofrece un diseño atractivo, un rendimiento optimizado y una amplia integración con otras herramientas y servicios. Con Ionic, puedo una aplicación móvil de alta calidad de manera más rápida y sencilla que con otras tecnologías.

Cloud Firestore

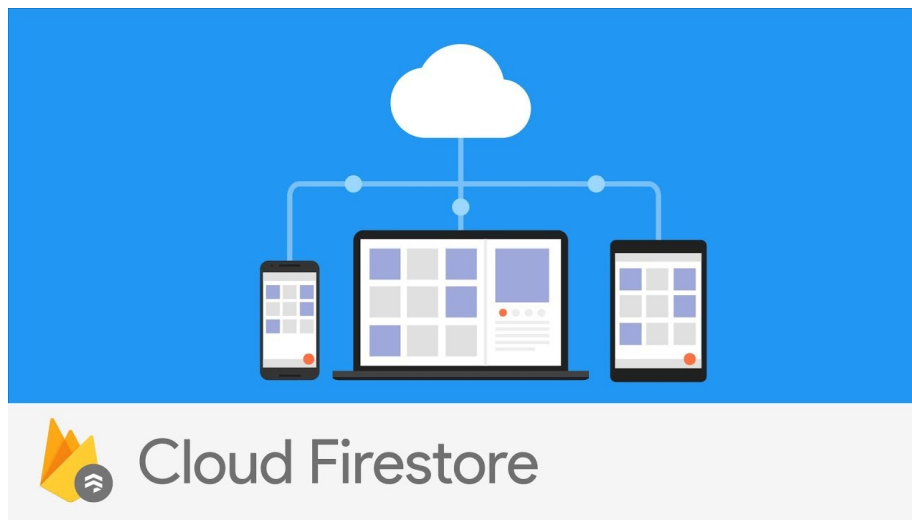


Figura 2: Imagen de Cloud Firestore

Cloud Firestore ofrece numerosas ventajas para el desarrollo de aplicaciones y la gestión de datos. Estas son algunas de las principales ventajas de utilizar Cloud Firestore:

- Escalabilidad y rendimiento: Cloud Firestore es una base de datos NoSQL en la nube altamente escalable. Puede manejar grandes volúmenes de datos y soportar una alta concurrencia de lecturas y escrituras sin degradar el rendimiento. Esto lo convierte en una excelente opción para aplicaciones con un crecimiento rápido y requerimientos de rendimiento exigentes.
- Sincronización en tiempo real: Cloud Firestore ofrece sincronización en tiempo real de datos. Esto significa que los cambios realizados en la base de datos se reflejan instantáneamente en todos los dispositivos conectados. Los clientes reciben actualizaciones automáticas sin necesidad de realizar consultas repetitivas al servidor. Esta funcionalidad es ideal para aplicaciones colaborativas y en tiempo real, como chats o paneles de control en vivo.
- Flexibilidad de datos estructurados: Cloud Firestore es una base de datos de documentos flexible que permite organizar los datos en colecciones y documentos. Puedes almacenar estructuras de datos complejas, como objetos anidados y matrices, sin necesidad de establecer un esquema rígido. Esto facilita el desarrollo y la evolución de la aplicación, ya que los cambios en la estructura de datos no requieren modificaciones extensas en el código.
- Seguridad y control de acceso: Cloud Firestore ofrece una amplia gama de opciones de seguridad para proteger tus datos. Puedes definir reglas de seguridad a nivel de documento y colección para controlar quién tiene acceso y qué operaciones pueden realizar. Además, Firestore se integra con la autenticación de Firebase, lo que te permite gestionar la identidad de los usuarios de manera segura y sencilla.
- Integración con ecosistema de Firebase: Cloud Firestore es parte del conjunto de herramientas de Firebase, lo que permite una fácil integración con otros servicios de Firebase. Puedes utilizar Firebase Authentication para la autenticación de usuarios, Firebase Cloud Functions para la lógica del servidor, Firebase Hosting para alojar tu aplicación web, entre otros. Esta integración proporciona una solución completa para el desarrollo de aplicaciones con todas las funcionalidades necesarias.
- Documentación y soporte: Cloud Firestore cuenta con una documentación completa y clara, así como una comunidad activa de desarrolladores que brindan soporte y comparten conocimientos. Además, Firebase ofrece servicios de asistencia técnica para ayudarte en caso de problemas o dudas.

En resumen, Cloud Firestore es una base de datos escalable, flexible y altamente sincronizada en tiempo real. Ofrece seguridad avanzada, integración con el ecosistema de Firebase y un excelente soporte. Al elegir Cloud Firestore, podré construir mi aplicación sin tener que preocuparme por la infraestructura subyacente.



Figura 3: Imagen de React

React

React es una biblioteca de JavaScript ampliamente utilizada en el desarrollo de interfaces de usuario. Sus numerosas ventajas hacen que sea una elección bastante adecuada para mi proyecto. Estas son algunas de las principales ventajas de utilizar React:

- **Eficiencia y rendimiento:** React utiliza un algoritmo de reconciliación virtual (Virtual DOM) que minimiza las actualizaciones y optimiza el rendimiento. Realiza cambios eficientes en el DOM, lo que resulta en una aplicación rápida y receptiva, incluso con grandes conjuntos de datos.
- **Componentes reutilizables:** React se basa en el concepto de componentes, que son bloques de construcción independientes y reutilizables. Puedes crear componentes personalizados y combinarlos para construir interfaces complejas. Esta modularidad facilita el mantenimiento del código y fomenta la reutilización, lo que ahorra tiempo y esfuerzo en el desarrollo.
- **Unidireccionalidad de datos:** React sigue el principio de flujo de datos unidireccional, lo que significa que los datos fluyen en una sola dirección. Esto facilita el seguimiento de los cambios y el mantenimiento del estado de la aplicación, lo que a su vez mejora la capacidad de depuración y la previsibilidad del flujo de datos.
- **Reactividad:** React ofrece una actualización automática y eficiente de la interfaz de usuario en función de los cambios en los datos. Cuando los datos cambian, React actualiza solo las partes necesarias de la interfaz de usuario, lo que mejora el rendimiento y la experiencia del usuario.

- **Amplio ecosistema y comunidad activa:** React cuenta con una amplia gama de bibliotecas y herramientas complementarias que facilitan el desarrollo, como React Router para el enrutamiento, Redux para la gestión del estado y Next.js para el desarrollo de aplicaciones web del lado del servidor. Además, tiene una comunidad activa de desarrolladores que comparten conocimientos, contribuyen a proyectos de código abierto y proporcionan soporte.
- **React Native para desarrollo móvil:** React Native, basado en React, permite desarrollar aplicaciones móviles nativas para iOS y Android utilizando JavaScript. Comparte la lógica de negocio y los componentes entre las plataformas, lo que acelera el desarrollo y facilita el mantenimiento de las aplicaciones móviles.

En resumen, React ofrece eficiencia, rendimiento, reutilización de componentes, unidireccionalidad de datos, reactividad y un ecosistema sólido. Todas las ventajas mencionadas anteriormente lo convierten en una opción sólida para el desarrollo de mi proyecto.

Typescript



Figura 4: Imagen de Typescript

La utilización de TypeScript en mi proyecto Ionic con React ofrece numerosas ventajas para el desarrollo de aplicaciones móviles. Estas son algunas de las principales ventajas de utilizarlo:

- **Tipado estático:** TypeScript agrega un sistema de tipos estático a JavaScript, lo que permite detectar y prevenir errores comunes durante la etapa de desarrollo. Al tener un tipado más fuerte, se reducen los errores en tiempo de ejecución y se mejora la calidad del código.
- **Autocompletado y sugerencias inteligentes:** Gracias a la información de tipos proporcionada por TypeScript, los entornos de desarrollo como Visual Studio Code ofrecen autocompletado y sugerencias inteligentes. Esto acelera el desarrollo al mostrar opciones y facilitar la exploración de métodos y propiedades disponibles.
- **Mantenimiento y legibilidad del código:** TypeScript promueve una estructura más clara y legible en el código fuente. La adición de tipos

permite comprender rápidamente qué tipo de datos se esperan y cómo se utilizan en diferentes partes del proyecto. Esto facilita la colaboración en equipo y el mantenimiento a largo plazo del código.

- Refactorización segura: Gracias a su capacidad para inferir tipos y detectar cambios relacionados, TypeScript proporciona una refactorización segura y confiable. Puedes realizar cambios en el código con confianza, ya que TypeScript puede encontrar y actualizar automáticamente todas las referencias afectadas.
- Mayor productividad y detección temprana de errores: Al utilizar TypeScript, se detectan muchos errores en tiempo de compilación en lugar de tiempo de ejecución. Esto permite identificar y solucionar problemas potenciales antes de ejecutar la aplicación, lo que ahorra tiempo y evita problemas en producción.
- Soporte para características de ES6 y ES7: TypeScript es compatible con las últimas características de JavaScript, incluyendo las introducidas en ECMAScript 6 (ES6) y ECMAScript 7 (ES7). Puedes aprovechar las ventajas de estas nuevas funcionalidades mientras se mantiene la compatibilidad con versiones anteriores de JavaScript.

En resumen, utilizar TypeScript en mi proyecto Ionic con React me proporciona beneficios como el tipado estático, autocompletado inteligente, mantenimiento del código, refactorización segura, detección temprana de errores y soporte para características de JavaScript moderno. Estas ventajas mejoran la calidad de mi código, aceleran el desarrollo y brindan una mayor confianza en la estabilidad y funcionalidad de mi aplicación móvil desarrollada.

Análisis

Para la organización de ideas de mi aplicación, una de las primeras cosas que tuve que hacer fue desarrollar el diagrama de casos de uso con lo que al menos serían las primeras ideas que tuve para desarrollarlo.

Como he mencionado anteriormente, esas fueron mis primeras ideas. El diagrama que realmente correspondería con mi proyecto es el siguiente:

Implementación

Script de introducción de los datos en Firestore

Cualquier persona que haya trabajado con Cloud Firestore de manera gratuita, es decir, sin tener ningún plan de pago y sin obtener funciones especiales, sabrá a la perfección para la introducción de datos se debe hacer de uno en uno ya que no te permite importar un archivo json ni nada por el estilo. Para solucionar

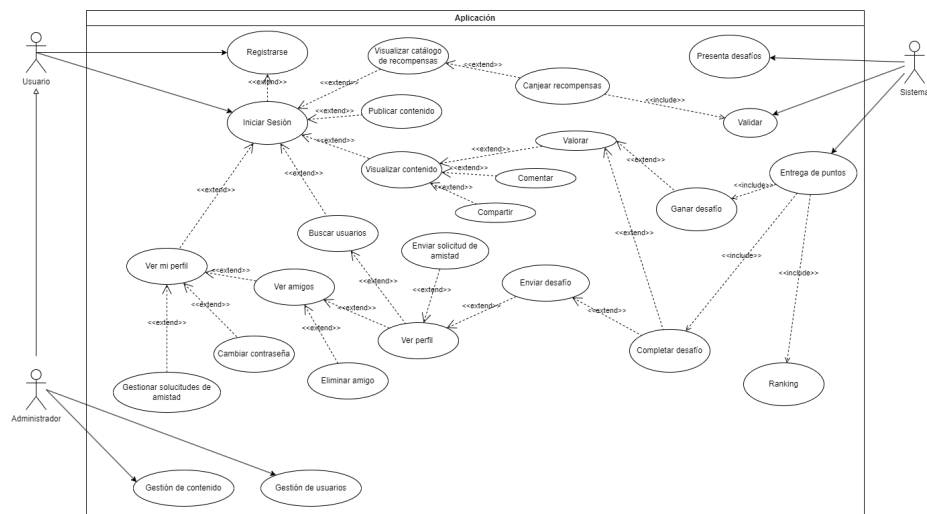


Figura 5: Imagen diagrama casos de uso

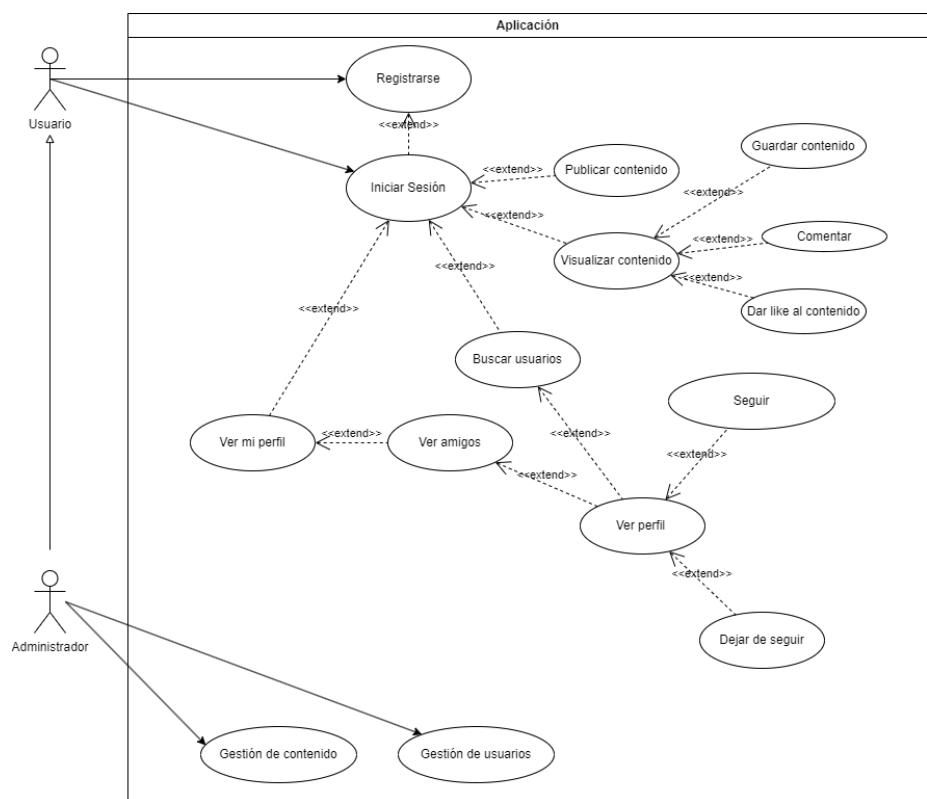


Figura 6: Imagen diagrama casos de uso

este problema, he realizado un script que me solucionara este problema. Dicho script se encuentra en la parte de helpers del proyecto.

He utilizado todos estos datos ficticios para comprobar cómo se vería realmente mi aplicación en funcionamiento pero, para su presentación, todos estos datos ficticios serán borrados ya que relentizarían bastante la aplicación.

Explicación de los arrays de datos

Lo primero que he tenido que hacer han sido los arrays con los datos de los campos que iban a tener cada usuario, entre ellos, nombre, apellido, descripción, etc. Este es el ejemplo de uno de ellos de apellidos españoles:

```
export const surname = ["García", "González", "Rodríguez", "Fernández", "López", "Martínez",
```

Todos estos arrays se exportaran del archivo **arrays.js** para ser importados en, ahora sí, el script de introducción de datos.

Explicación del script de introducción de datos

Lo primero que se ha hecho en el script es la configuración de firebase. Dicha configuración no es algo complicado ya que está todo incluido en la documentación y además en la parte de configuración de tu proyecto te lo proporciona.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
import { deleteDoc, doc, DocumentReference, getDoc, getFirestore } from "firebase/firestore";
import { collection, getDocs, addDoc } from "firebase/firestore";

// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
// For Firebase JS SDK v7.20.0 and later, measurementId is optional
const firebaseConfig = {
  apiKey: "AIzaSyCZ8I-PMSduWcObF6TSz4U2QOFwsgf8i_k",
  authDomain: "moveup-2ba70.firebaseio.com",
  projectId: "moveup-2ba70",
  storageBucket: "moveup-2ba70.appspot.com",
  messagingSenderId: "118564026428",
  appId: "1:118564026428:web:9ad69b11bf3075af3c2b39",
  measurementId: "G-24V496X6YY"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
const db = getFirestore(app);
const USERS_COLLECTION = "usuarios";
```

Ahora lo que he ido haciendo es, a partir de los arrays importados anteriormente y de la clase Math, generar todos los campos necesarios para cada usuario que va a ser alojado en la base de datos. * Generación de números aleatorios de seguidos:

```
const following = [];

for (let i = 0; i < 100; i++) {
  const randomNum = Math.floor(Math.random() * 900) + 100; // Genera un número aleatorio en
  following.push(randomNum);
}
```

- Para la generación de un número aleatorio de seguidores he utilizado una *template string* que debía de usarse cuando el número de seguidores fuera mayor de 4 cifras.

```
const followers = [];

for (let i = 0; i < 50; i++) {
  const randNum = Math.floor(Math.random() * (9999 - 1000 + 1)) + 1000;
  const randStr = `${Math.floor(Math.random() * 100)},${Math.floor(Math.random() * 9) + 1}K`;
  followers.push(Math.random() < 0.5 ? randNum : randStr);
}
```

- Generación de IDs aleatorios válidos para Firestore:

```
const letters = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';
const ids = [];

for (let i = 0; i < 50; i++) {
  let combination = '';

  for (let j = 0; j < 20; j++) {
    const randomIndex = Math.floor(Math.random() * letters.length);
    const randomLetter = letters[randomIndex];
    combination += randomLetter;
  }
  ids.push(combination);
}
```

Había que crear el método para que realizar el post a la base de datos:

```
const addUsers = async (users) => {
  try {
    //Definiendo la colección
    const cursesCol = collection(db, USERS_COLLECTION);
    const docRef = await addDoc(cursesCol, users);
    console.log("Insertado con éxito. ID: " + docRef.id)
  } catch (e) {
```

```

        console.error("Error en la insercción")
    }
}

```

Una vez hecho esto, ya solo quedaba recorrer los arrays a través de un bucle y asignar a cada usuario creado un campo de dicho array. Para la creación del nombre de usuario he vuelto a utilizar una *template string* en la que el nombre de usuario estuviera compuesto por las tres primeras letras del nombre, las tres primeras letras del apellido y un número de dos cifras aleatorio.

```

// Creación de datos de mujeres
for (let i = 0; i < avatar_women.length; i++) {
    const user = new Object();
    user.avatar = avatar_women[i];
    user.bio = bio [i];
    user.firstname = firstname_women[i];
    user.followers = followers[i];
    user.following = following[i];
    user.link = link[i];
    user.posts = [];
    user.surname = surname[i];
    user.title = title_female[Math.floor(Math.random() * title_female.length) + 1];
    user.username = `${firstname_women[i].substring(0,3)}${surname[i].substring(0,3)}${Math.f

// Añadiendo cada usuario
addUsers(user);
}

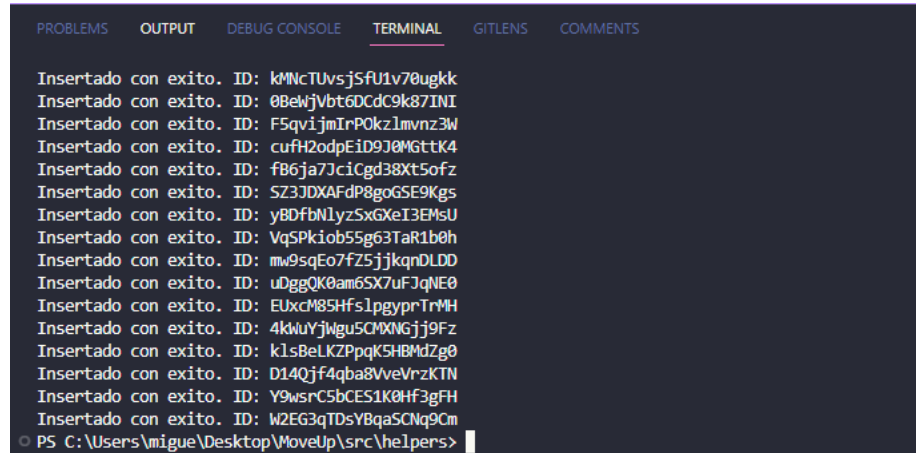
// Creación de datos de hombres
for (let i = 0; i < avatar_men.length; i++) {
    const user = new Object();
    user.avatar = avatar_men[i];
    user.bio = bio [i];
    user.firstname = firstname_men[i];
    user.followers = followers[i];
    user.following = following[i];
    user.link = link[i];
    user.posts = [];
    user.surname = surname[i];
    user.title = title_men[Math.floor(Math.random() * title_men.length) + 1];
    user.username = `${firstname_men[i].substring(0,3)}${surname[i].substring(0,3)}${Math.f

// Añadiendo cada usuario
addUsers(user);
}

```

Para finalizar, tan solo había que ejecutar dicho script con nodejs y esperar a

que todos los datos se introdujeran correctamente en la base de datos:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL GITLENS COMMENTS
Insertado con éxito. ID: kMNCtUvvsj5fU1v70ugkk
Insertado con éxito. ID: 0BewjVbt6DCdC9k87INI
Insertado con éxito. ID: F5qvijmIrP0kzLmvnz3W
Insertado con éxito. ID: cufH2odpEiD9J0MGttK4
Insertado con éxito. ID: fB6ja7JciCgd38Xt5ofz
Insertado con éxito. ID: SZ3JDXAFdP8goGSE9Kgs
Insertado con éxito. ID: y8DfbNlyzSxGxI3EMsU
Insertado con éxito. ID: VqSPkiob55g63TaR1b0h
Insertado con éxito. ID: mw9sqEo7fZ5jjkqnDLDD
Insertado con éxito. ID: uDggQK0am6SX7uFJqNE0
Insertado con éxito. ID: EUxcM85HfslpgyprTrMH
Insertado con éxito. ID: 4kluYjWgu5CMXNGjj9Fz
Insertado con éxito. ID: k1sBeLK7PpqK5HBMDzg0
Insertado con éxito. ID: D14Qjf4qba8VveVrzKTN
Insertado con éxito. ID: Y9wsrC5bCES1K0HF3gFH
Insertado con éxito. ID: W2EG3qTDsYBqaSCNq9Cm
PS C:\Users\miguel\Desktop\MoveUp\src\helpers>
```

Figura 7: Captura de la ejecución del script

Modificación del archivo vite.config.ts

Una modificación importante que he tenido que realizar en el archivo *vite.config.ts* ha sido la implementación del siguiente código:

```
optimizeDeps: {
  exclude: [`@ionic/pwa-elements/loader`],
}
```

Hecho esto y, siguiendo el tutorial de como implmentar la cámara y la subida de archivos desde el dispositivo adjuntado en la bibliografía, he podido conseguir que funcionara. Este punto me parecía crítico ya que, en el propio tutorial, no se mencionaba y he tenido que buscar bastante información para solucionarlo. El archivo *vite.config.ts* ha quedado así:

```
/// <reference types="vitest" />
import { defineConfig } from 'vite'
import react from '@vitejs/plugin-react';

export default defineConfig({
  optimizeDeps: {
    exclude: [`@ionic/pwa-elements/loader`],
  },
  plugins: [react()],
  test: {
    globals: true,
    environment: 'jsdom',
    setupFiles: ['./src/setupTests.ts'],
  },
});
```

```
}  
});
```

Reglas Cloud Firestore y Storage

Para lanzar tu proyecto en modo producción en Firebase, deben de haber unas reglas que se deben configurar sobre qué y quienes pueden leer y escribir datos. En el caso de aplicación, para su uso, se necesitaba un logueo previo por lo que estas reglas en sí en *modo de pruebas* (modo en el que todos pueden leer y escribir datos) no tenían mucha repercusión. Pero lo ideal, tal y como debe de hacerse, es definiendo una serie de reglas que, en mi caso, es que para escribir y leer los datos de mi aplicación solo lo podrán hacer los que se hayan logueado contra la autenticación de firebase, también usada en mi aplicación.

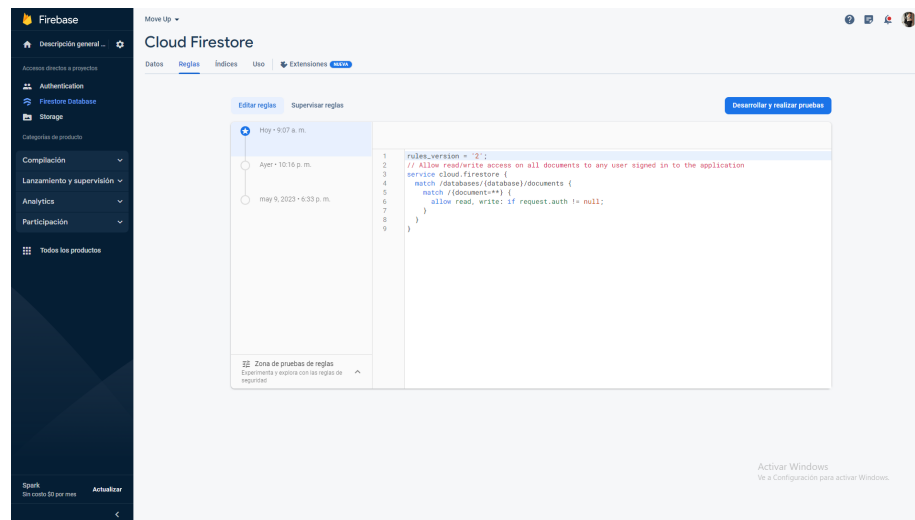


Figura 8: Captura de las reglas de Cloud Firestore

Al igual que en Cloud Firestore, para utilizar el Storage de Firebase pasa exactamente lo mismo. Para su uso se deben definir una serie de reglas. En el caso de mi aplicación, utilizo el Storage de Firebase para almacenar todo el contenido que se sube y alojarlo en una url que después consumo. Tal y como definí anteriormente que solo los usuarios logueados pudieran leer y escribir los datos, en este caso hice lo mismo.

Generación proyectos nativos android e ios

Para la generación de los proyectos nativos de android e ios, una de las formas que hay para hacerlo y la que yo personalmente he utilizado es capacitor.

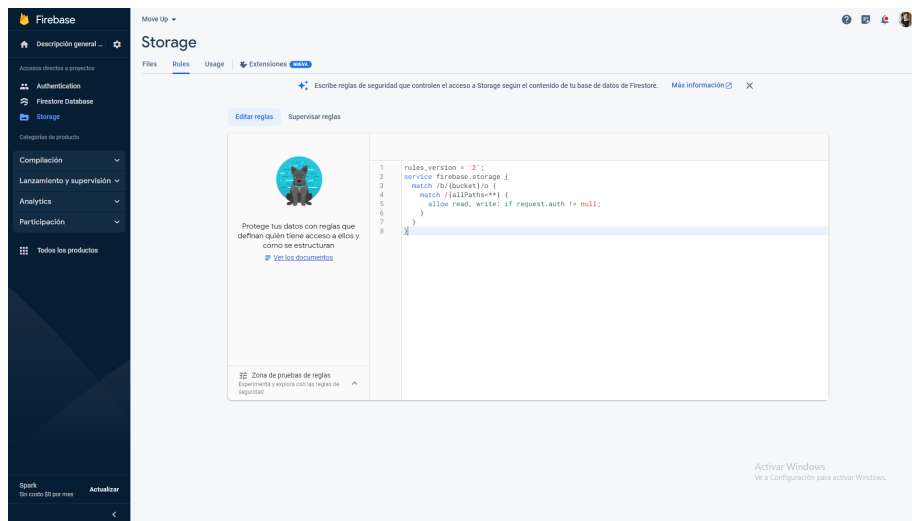


Figura 9: Captura de las reglas de Cloud Firestore

Generación del proyecto android

Se debe añadir la dependencia al proyecto de @capacitor/android:

```
npm install @capacitor/android
```

Para generar la carpeta android con el proyecto:

```
npx cap add android
```

Para abrir un IDE, en este caso, Android Studio:

```
npx cap open android
```

Una vez hecho esto se habrá desplegado nuestro IDE con la aplicación generada que podremos correr en un emulador.

Generación del proyecto ios

En el caso de la aplicación ios no he podido darle los últimos retoques como si que he hecho con la aplicación android ya que, para hacerlo, se necesita el editor de código Xcode y, en su defecto, un ordenador con sistema operativo de Apple.

Se debe añadir la dependencia al proyecto de @capacitor/ios:

```
npm install @capacitor/ios
```

Para generar la carpeta android con el proyecto:

```
npx cap add ios
```

Para abrir un IDE, en este caso, Android Studio:

`npx cap open android`

Una vez hecho esto se habrá desplegado nuestro IDE con la aplicación generada que podremos correr en un emulador.

Conclusiones

Durante el desarrollo de mi proyecto final, he tenido la oportunidad de adquirir nuevos conocimientos y experiencias que han sido fundamentales para mi formación académica. A lo largo de este proceso, he trabajado en el desarrollo de una aplicación Ionic con React y Firebase como herramientas principales.

Una de las principales lecciones que he aprendido es la importancia de la planificación y el diseño previo antes de iniciar cualquier desarrollo. Si hubiera realizado una mejor planificación y organización del proyecto habría solucionado desde un principio errores a los que me he enfrentado como la mala elección de las tecnologías, entre ellas, firebase. Para la realización de mi proyecto, una tecnología que hubiera sido mucho más adecuada para mis funcionalidades hubiera sido, por ejemplo, mongo. Si hubiera realizado un análisis exhaustivo de los requisitos y la definición clara de los objetivos, habría logrado establecer una base sólida para mi proyecto y esto me hubiera permitido optimizar mis esfuerzos y minimizar los obstáculos encontrados durante el desarrollo.

Sin embargo, he adquirido un amplio conocimiento sobre las tecnologías utilizadas en mi proyecto, como Ionic y React. Estas plataformas me han brindado la flexibilidad necesaria para crear una aplicación con una interfaz de usuario intuitiva y atractiva. A través de la implementación de componentes reutilizables y la utilización de librerías especializadas, he logrado mejorar la eficiencia de mi desarrollo y proporcionar una experiencia de usuario de calidad.

Firebase, por otro lado, a pesar de los problemas que ha podido llegar a darme, me ha ofrecido una plataforma sólida para gestionar la base de datos y la autenticación de usuarios en mi aplicación. Su integración con Ionic y React ha facilitado enormemente el proceso de almacenamiento y recuperación de datos, así como la implementación de características de seguridad y acceso restringido. Esta combinación de tecnologías me ha permitido desarrollar una aplicación escalable, segura y bastante cómoda ya que no tengo que desplegar ningún contenedor docker con mi base de datos, backend, etc.

A lo largo de este proyecto, también he tenido la oportunidad de enfrentarme a desafíos y obstáculos que me han brindado valiosas lecciones. He aprendido la importancia de ser bastante autodidacta con los problemas que pueda encontrarme así como la necesidad de adaptarme a los cambios y tomar decisiones ágiles. Estas habilidades serán extremadamente útiles a lo largo de mi carrera laboral.

Mirando hacia el futuro, tengo la intención de seguir mejorando mi aplicación y, sobre todo, desarrollar toda la lluvia de ideas que había tenido en un principio

además explorar nuevas oportunidades para su implementación. Quiero realizar pruebas adicionales, recopilar comentarios de la gente de mi entorno y continuar añadiendo características y funcionalidades.

En conclusión, este proyecto me ha proporcionado un conjunto diverso de habilidades y conocimientos, desde la planificación y el diseño hasta la implementación y la gestión de un proyecto bastante más grande de lo que había trabajado hasta ahora. He aprendido a utilizar eficientemente las herramientas de desarrollo como Ionic, React y Firebase, y he adquirido una comprensión más profunda de la importancia de una buena planificación. Estoy contento por las ganas de seguir aprendiendo que me ha proporcionado este proyecto y me siento bastante preparado para enfrentar cualquier desafío que se me presente.

Bibliografía

Esta parte constará webs, tutoriales, vídeos que he visto y que me han ayudado con el desarrollo de mi proyecto:

- Realización del diagrama de casos de uso: <https://app.diagrams.net/>.
- Documentación Ionic oficial: <https://ionicframework.com/>
- Extracción de una plantilla similar a Instagram: <https://ionicreacthub.com/ionic-instagram-clone>
- Documentación de Firebase oficial: <https://firebase.google.com/docs>
- Consultas variadas de información: <https://chat.openai.com/>
- Tutorial para la implementación de la captura y subida de imágenes: <https://ionicframework.com/docs/react/your-first-app>
- Resolución de problema con la cámara: <https://forum.ionicframework.com/t/problem-following-the-photo-gallery-tutorial/232236/6>