UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA INFORMÁTICA EN
TECNOLOGÍAS DE LA INFORMACIÓN

DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN WEB PARA DOCUMENTACIÓN Y TOMA DE NOTAS

TRABAJO FIN DE GRADO

Junio 2025

AUTOR: Franc Tauste González

DIRECTOR: Manuel Quesada Martínez

RESUMEN

En la actualidad existen diferentes alternativas de aplicaciones para la toma de notas que ofrecen una funcionalidad diversa. Sin embargo, cuando hablamos de una alternativa real ante el papel nos encontramos con la pregunta, ¿existe una herramienta capaz de sustituirlo y crear un estándar al que los usuarios accedan de forma sencilla y rápida?

En este trabajo se realiza la implementación de una aplicación web para la toma de notas con un enfoque simple, pero incluyendo algunas de las funcionalidades más usadas entre los usuarios. Para ello, se analizan distintos trabajos académicos existentes donde se muestran evidencias del uso de estas funciones, así como del uso de otras nuevas opciones innovadoras como la Inteligencia Artificial o la toma de notas por voz. Además, durante la implementación se ha abordado el uso de tecnologías novedosas como Firebase, donde gestionamos los datos con bases de datos no relacionales, o React donde indagamos sobre el uso de una librería web para el desarrollo de aplicaciones del lado del cliente.

Tras la realización de este TFG, se ha creado un prototipo funcional de una aplicación de toma de notas cumpliendo los objetivos deseados. Como trabajo futuro podemos destacar la inclusión de nueva funcionalidad como: la mejora de la esquematización de la información, el uso de tablas Kanban para la gestión de proyectos, la mejora de la personalización que ofrecen las herramientas, añadir la posibilidad de trabajo colaborativo o desplegar la aplicación en un entorno real.

Palabras clave: toma de notas, react, aplicación web, firebase.

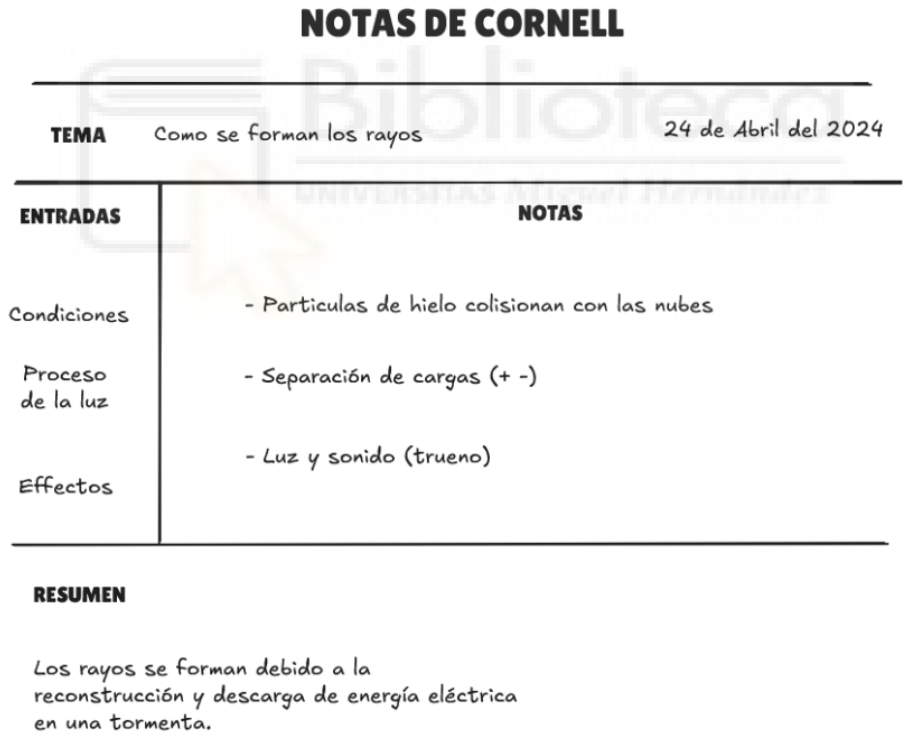
AGRADECIMIENTOS

Quiero agradecer a todas las personas que me han ayudado y apoyado para realizar este proyecto, como ha sido mi familia y el tutor Manuel Quesada Martínez. Además, quiero también agradecer a todos los profesores de la Universidad con los que he tenido la oportunidad de aprender nuevos conocimientos durante los últimos años.

ÍNDICE DE CONTENIDOS

1. INTRODUCCIÓN

La toma de notas en las actividades de nuestro día a día es una técnica imprescindible con las que podemos tomar apuntes, por ejemplo, durante la sesión de una asignatura, una reunión o cuando necesitamos representar una nueva idea de forma clara y concisa. Además, este tipo de aplicaciones no solo están presentes en ámbitos laborales o estudiantiles, sino que podemos aplicarlas para cualquier tipo tarea sobre la que se requiera realizar un buen seguimiento y organización de la información. Esto es debido a la cantidad de actividades que realizamos a lo largo del día, así como las pocas cosas que podemos llegar a recordar y tener en cuenta al mismo tiempo. Por ello, frecuentemente recurrimos a métodos externos como las notas para guardar información y usarla en otro momento.



NOTAS DE CORNELL

TEMA Como se forman los rayos 24 de Abril del 2024

ENTRADAS	**NOTAS**
Condiciones	- Particulas de hielo colisionan con las nubes
Proceso de la luz	- Separación de cargas (+ -)
Effectos	- Luz y sonido (trueno)

RESUMEN

Los rayos se forman debido a la reconstrucción y descarga de energía eléctrica en una tormenta.

Ilustración 1: Ejemplo de las notas de Cornell

En el origen, las personas usaban métodos rudimentarios y costosos para tomar notas, que más adelante se actualizarían con la invención del papel. El papel permitió evolucionar la escritura haciéndola más cómoda y accesible para todo el mundo [1].

Tal fue su éxito que con el tiempo aparecieron las primeras técnicas propuestas para el aprendizaje y organización de la información en la Universidad, como son las Notas de Cornell [2]. En la Ilustración 1 se muestra un ejemplo de los campos propuestos por Cornell, que como vemos es una forma estructurada de organizarnos.

Si movemos el foco al ámbito de las aplicaciones, no sería hasta la invención de los ordenadores personales cuando se intentaría digitalizar el papel con el desarrollo de los primeros procesadores de texto. Con ello aparecerían las primeras aplicaciones para la toma de notas y texto plano como el Bloc de Notas de Microsoft para el Sistema Operativo MS-DOS, MacWrite para Mac OS o Lotus Agenda para DOS [3].

Actualmente todo ha evolucionado mucho y ya existen multitud de métodos para realizar diferentes tipos de anotaciones. Tomar notas puede realizarse desde la aplicación de notas de tu teléfono, hasta con paquetes de software más complejos con herramientas de todo tipo para la oficina. En los últimos tiempos, la gestión de notas incluso se ha trasladado a los propios recordatorios de los asistentes virtuales por voz.

Durante su desarrollo, muchas de estas aplicaciones adquieren su carácter diferenciador al enfocándose en solucionar uno o varios problemas específicos. De este modo, las aplicaciones se especializan en automatizar necesidades concretas que no siempre se acaban de amoldar a nuestras necesidades. Estos problemas, a veces, llevan a los usuarios finales a dejar de utilizarlas, y finalmente se acaba recurriendo al clásico papel y bolígrafo.

1.1. MOTIVACIÓN

La idea de este Trabajo Fin de Grado (TFG) gira en torno a la problemática descrita en el párrafo anterior. Por ejemplo, personalmente pese a haber probado distintas aplicaciones de toma de notas, siempre acabo recurriendo o la aplicación que tengo más a mano, como sería el bloc de notas del ordenador o del teléfono, o incluso al papel y bolígrafo. El motivo de elegir un simple editor de texto es que las funcionalidades ofrecidas por otras herramientas no terminan de resultarme útiles pese a su complejidad.

Por lo tanto, la motivación de este trabajo es llenar el vacío de funcionalidad de las actuales aplicaciones de toma de notas, dado que el hecho de tomar notas básicas es una tarea difícil de gestionar. Con la aplicación aquí desarrollada busco encontrar un punto intermedio entre un editor sencillo que podemos tener siempre a mano, pero que sea capaz de reflejar ideas más complejas.

1.2. OBJETIVOS

Teniendo en cuenta la motivación descrita, el objetivo principal de este TFG es:

- Crear una aplicación web para la toma de notas general del día a día, proporcionando las herramientas esenciales para el usuario con una interfaz y uso amigable.

Para lograr este objetivo, nos marcaremos los siguientes objetivos secundarios:

1. Implementar un sistema para la documentación más complejo que un simple editor de texto, y encontrar un punto intermedio entre las típicas aplicaciones de pizarra de dibujo y otras más enfocadas a la realización de documentos detallados.

2. Realizar una investigación sobre las aplicaciones en tendencia de toma de notas, con el fin de conocer mejor que funcionalidad incorporan y que herramientas son más destacadas para los usuarios. De esta forma podremos incorporar las más imprescindibles en el trabajo, e incluso poder implementar nuevas propuestas que permitan a nuestra aplicación hacerse un hueco entre ellas.

3. Integrar en la aplicación alguna funcionalidad relacionada con la Inteligencia Artificial (tendencia en los últimos años). Por ejemplo, con el uso de Inteligencia Artificial podríamos facilitar la toma de notas durante clases y o reuniones de trabajo realizando resúmenes sin tener que dejar de atender.

2. MATERIALES Y MÉTODOS

En este capítulo realizaremos un estudio del estado del arte, desde el análisis de aplicaciones relacionadas, hasta el estudio de tecnologías actuales que nos permitan implementar la aplicación web de toma de notas.

2.1. ANÁLISIS DE TRABAJOS RELACIONADOS

En este apartado vamos a analizar algunos trabajos y herramientas relacionadas con las aplicaciones de toma de notas que nos ayudará a identificar los requisitos y funcionalidades a incluir.

2.1.1. HERRAMIENTAS RELACIONADAS

Si queremos hacer que la aplicación que este los más a mano posible, un factor a tener en cuenta es su implementación en accesible desde el mayor número de dispositivos. Además, sería deseable conocer que características y funcionalidades son las que destacan en la toma de notas entre las aplicaciones ya existentes.

Por ejemplo, en el artículo *"Affordances of mobile devices and note-taking apps to support cognitively demanding note-taking"* [4] se realiza un estudio con varios estudiantes, que comparten sus experiencias usando varias aplicaciones de notas, destacando que no hay ninguna en concreto que satisfaga todas las necesidades. Además, en dicho trabajo también se destaca la importancia de los teléfonos móviles para organizar la información de manera eficiente en comparación con el uso del papel y bolígrafo.

Algunas de las funcionalidades que estos estudiantes destacan son:

- Gestión de documentos y carpetas.
- Uso de teclado para escribir.
- Reconocimiento de texto por dibujo.

- Dibujo en diapositivas PDF.
- Texto formateado.
- Inserción de imágenes o archivos multimedia.
- Compartir documentos.

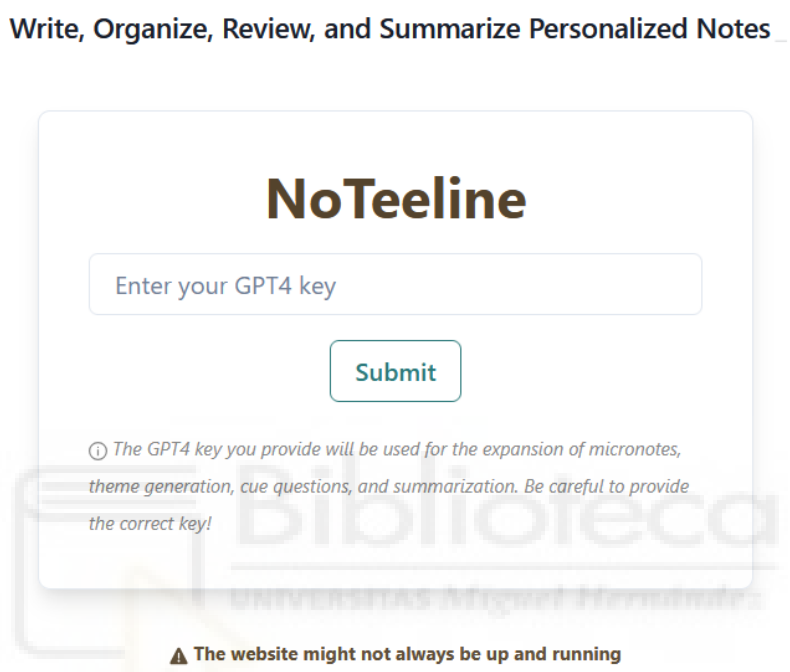


Ilustración 2: Aplicación NoTeeline

2.1.2. IA Y NOTAS DE CORNELL CON NOTEELINE

En el estudio [5] se desarrolla una aplicación, NoTeeline (ver Ilustración 2) para la toma de notas con el uso de Inteligencia Artificial. Esta aplicación permite mejorar en detalle las descripciones de las notas en tiempo real. Basándose en el formato de las notas de Cornell, la aplicación integra tres paneles principales:

- **Entradas:** en este panel se anotan todo tipo de palabras o conceptos clave, como preguntas sin respuesta, ideas o algunos puntos a destacar. La aplicación usa este apartado como referencia para la revisión de las notas.

- **Notas:** este panel contiene la mayoría de las descripciones más detalladas, como pueden ser ejemplos, esquemas, fórmulas matemáticas, etc. La aplicación usa este apartado para la expansión de notas.

- **Resumen:** este último panel se debe usar para resumir las partes más importantes que han aparecido durante la sesión donde hemos estado anotando las entradas y las notas.

Además, en dicho trabajo se realiza una investigación con 12 participantes que utilizan la herramienta durante un periodo de tiempo para calificar la calidad de las notas generadas por IA. De estos resultados se obtiene que la mayoría de los participantes se encuentran satisfechos con su uso, debido a que: las notas generadas son de buena calidad, se pueden tomar de una forma más eficiente y que el esfuerzo mental se reduce significativamente, llegando obtener resultados satisfactorios en el 93.2 % de las veces.

2.1.3. TOMA DE NOTAS POR VOZ

La Inteligencia Artificial es sin duda una herramienta a tener en cuenta para redactar notas con más detalle y menos esfuerzo que cuando estamos usando el teclado. Es destacable que según el dispositivo en el que nos encontremos el teclado puede no ser la alternativa más cómoda, por lo que recurrimos a otras posibilidades como las notas por dibujo o por voz.

Las notas por dibujo son más difíciles de tratar, pero para la voz contamos con sistemas de reconocimiento por IA que nos ayudan a transcribir el audio a texto sin ningún problema, situándolo como un sistema similar al teclado donde podemos tratar el texto a nuestro gusto.

Buscando en Internet nos encontramos con un artículo [6] que comparan el uso de teclado con el uso de la voz para la toma de notas y como afectan en la compresión conceptual y el aprendizaje. En el que llegamos a la conclusión de que no existe únicamente una herramienta que sea mejor o peor, pero sí que se recomienda la implementación de varias alternativas como la voz para fomentar su uso y llegar a ser más eficientes.

En resumen, algunas de las funcionalidades ofrecidas por las herramientas que son de especial relevancia son:

1. Gestión de notas en la nube.
2. Personalización.
3. Escritura por teclado.
4. Dibujo.
5. Asistencia para la toma de notas con IA.
6. Toma de notas por voz.

2.2. ANÁLISIS DE APLICACIONES RELACIONADAS

Aunque algunos de los trabajos anteriores habían desarrollado herramientas para la toma de notas, algunas de ellas eran prototipos funciones poco comerciales. En este apartado dirigimos nuestra atención a aplicaciones disponibles en tiendas populares.

2.2.1. EXCALIDRAW

Excalidraw [7] es una herramienta de código abierto del tipo pizarra, para dibujar y realizar diagramas de forma muy sencilla. En la Ilustración 3 se muestra su interfaz. Esta aplicación se encuentra disponible en Internet sin necesidad de usar una cuenta y se puede usar de forma local en el ordenador.

Algunas de las características a destacar son:

Ventajas

- Trabajo colaborativo.
- Incluye la importación de bibliotecas públicas para usar elementos predefinidos por otras personas.
- Buena personalización.

Desventajas

- No permite guardar los proyectos en la nube solo exportarlos.

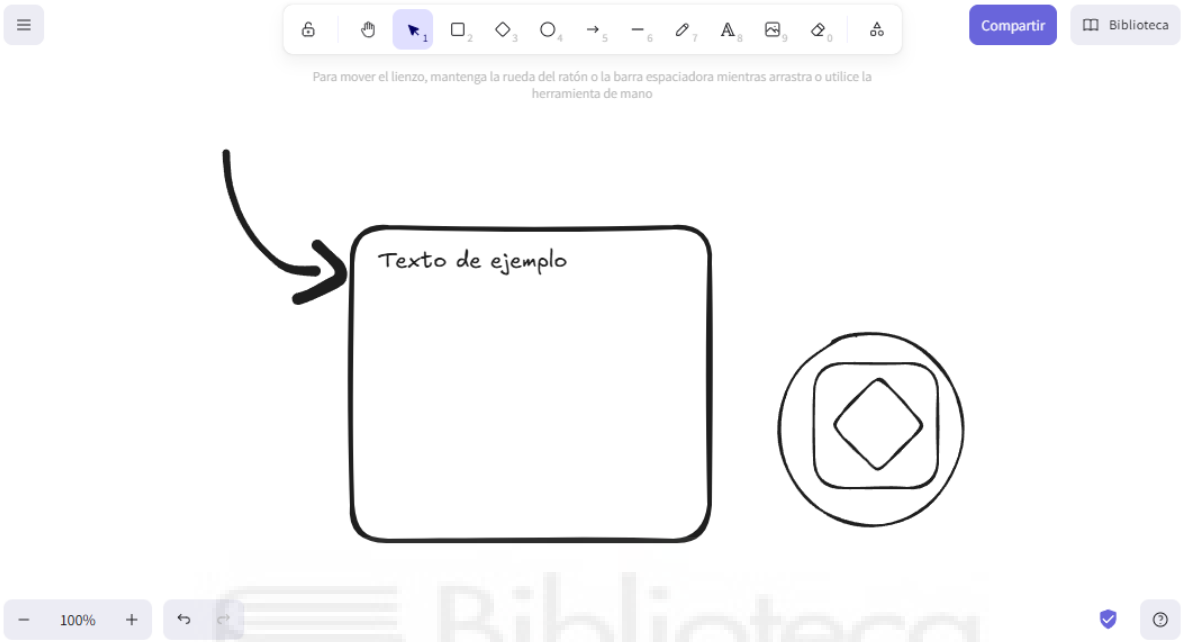


Ilustración 3: Ejemplo de uso de la aplicación Excalidraw

2.2.2. FREEFORM

Freeform [8] es una aplicación desarrollada por Apple (ver Ilustración 4). Es otro software para representar tus ideas en un tablero del tipo pizarra. Permite importar imágenes, dibujar, añadir notas, trabajo colaborativo, etc.

Ventajas

- Trabajo colaborativo.
- Mucha personalización

Desventajas

- Solo disponible para productos de Apple.
- Pocas herramientas diferentes disponibles.

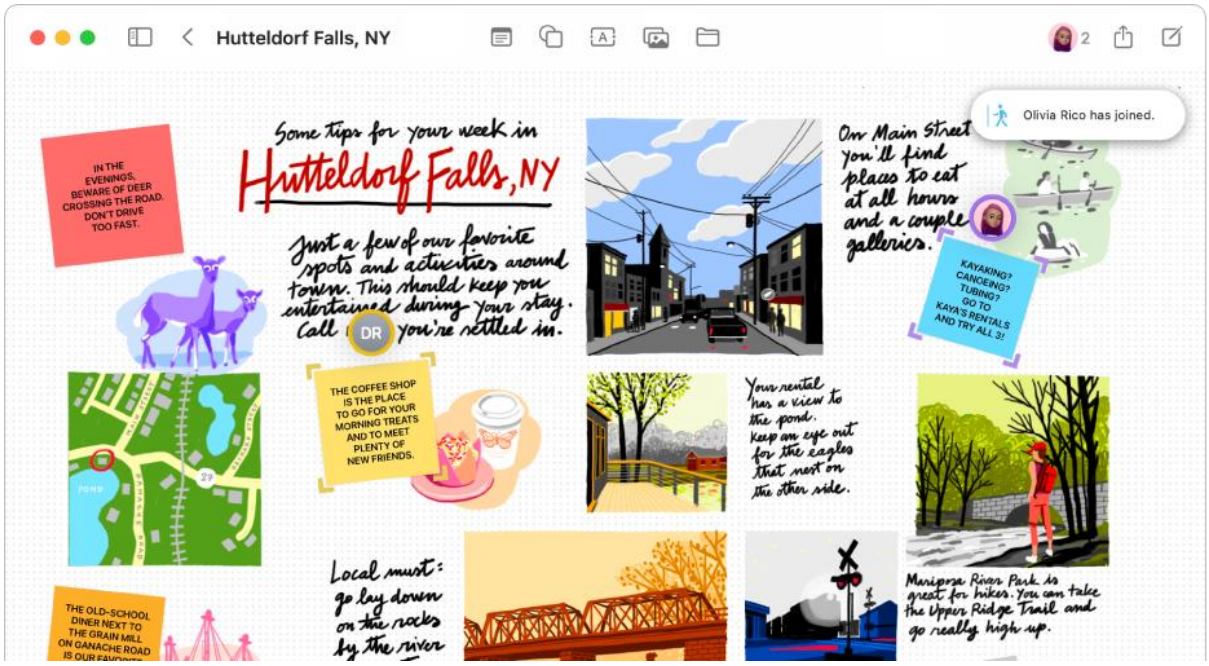


Ilustración 4: Ejemplo de uso de la aplicación Freeform

2.2.3. NOTION

Notion [9] es una aplicación web para la documentación y toma de notas, que se ha hecho destacar en los últimos años debido al paso de ofrecer un nuevo plan gratuito ilimitado para todos los usuarios y por su útil uso en la organización de proyectos.

En la Ilustración 5 se muestra su interfaz principal. Esta aplicación incluye además infinidad de herramientas para la gestión de tareas, proyectos colaborativos e integración con otras plataformas, que aumenta sus capacidades más allá de una aplicación de notas.

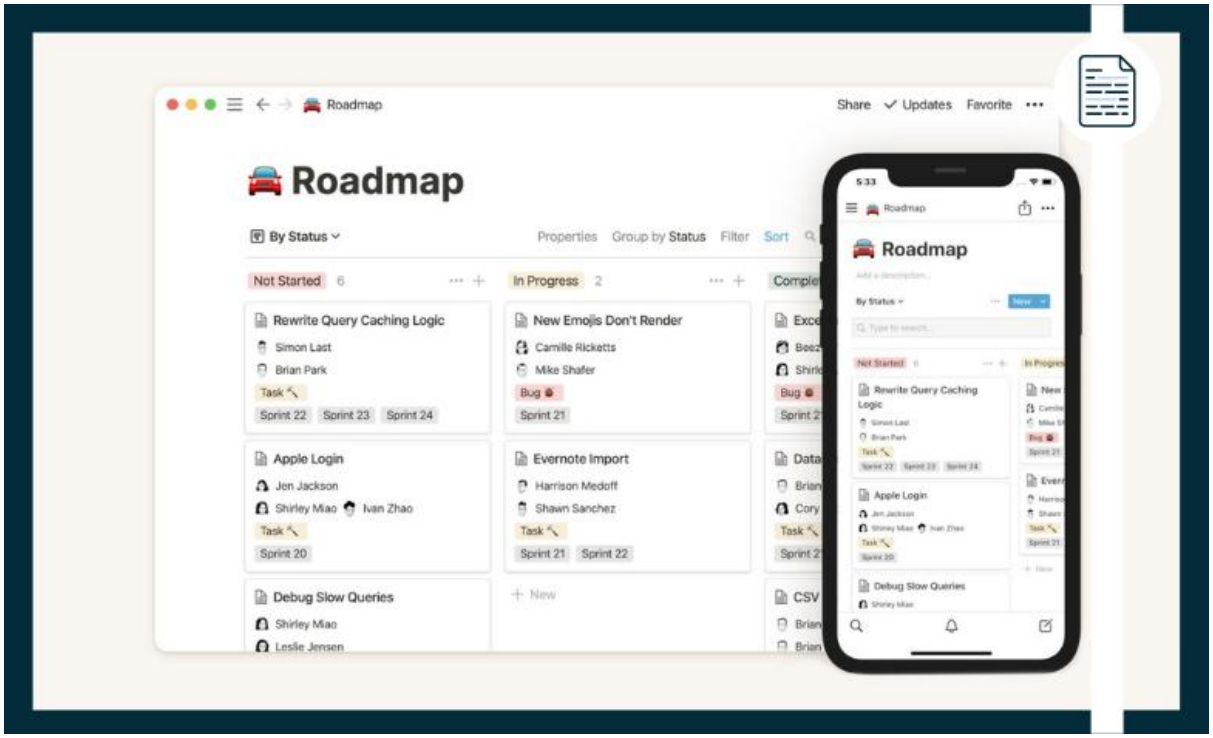


Ilustración 5: Ejemplo de uso de la aplicación Notion

Ventajas

- Documentación detallada con Markdown.
- Integración con Inteligencia Artificial.
- Trabajo colaborativo.
- Integración con otras plataformas.
- Gestor de correo.
- Calendario.
- Gestión de proyectos.

Desventajas

- No permite la toma de notas por dibujo.

2.2.4. ONENOTE

OneNote [10] es la aplicación para la toma de notas de Microsoft, que incluye en su paquete de software de oficina Office 365, de forma gratuita y de pago con funcionalidades ampliadas.

Esta aplicación nos permite usar notas en forma de pizarra para dibujar, o en forma de texto, además tiene la opción de añadir tablas y listas de tareas con mucha personalización (ver Ilustración 6). Estas funcionalidades la hacen una opción muy accesible para todo el mundo ofreciendo lo justo y necesario.

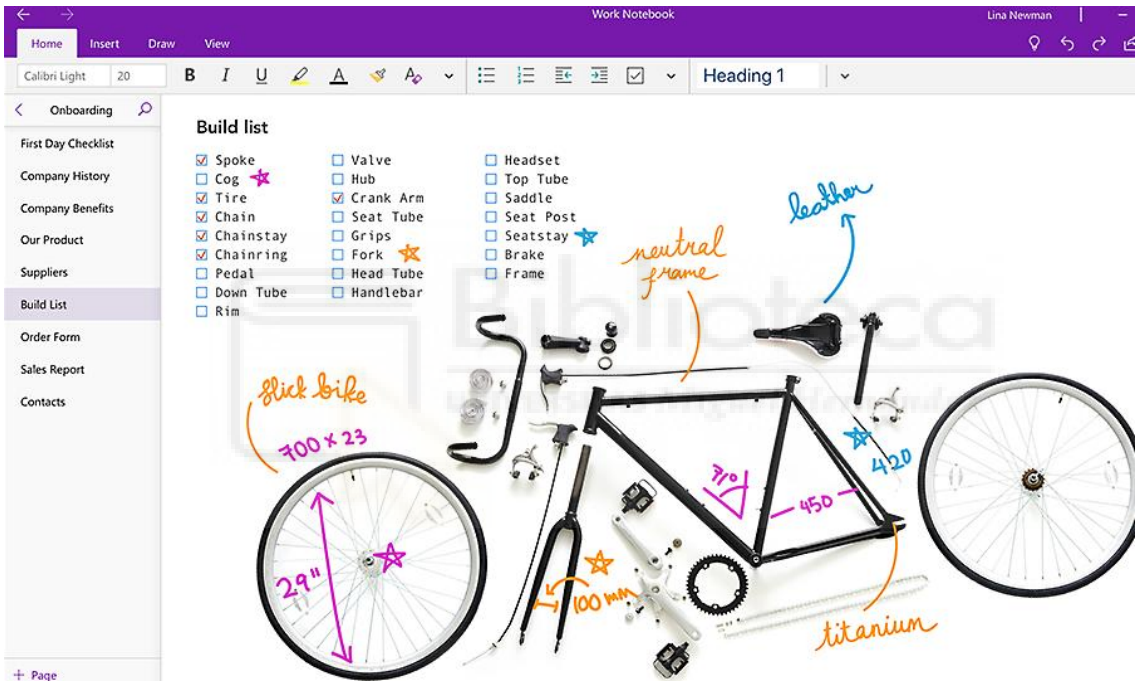


Ilustración 6: Ejemplo de uso de la aplicación OneNote

2.2.5. RESUMEN Y COMPARATIVA

En la Tabla 1 se muestra una tabla comparativa de las diferentes aplicaciones analizadas. En la primera columna de la tabla se muestran las características que se han analizado de todas ellas. Estas características son las que hemos definido como deseables de incluir en una aplicación de este tipo.

	Excalidraw	Freeform	Notion	OneNote
Insertar imágenes	Si	Si	Si	Si
Dibujo	Si	Si	No	Si
Documentación	No	No	Si	Si
IA	Si	No	IA de Notion	Copilot
Listas de tareas	No	No	Si	Si
Trabajo colaborativo	Si	Si	Si	Si
Disponibilidad	Gratis	Gratis	Gratis y pago	Gratis
Plataforma	Web	iOS, Mac OS	Web	PC, Móvil y Web

Tabla 1 Comparativa de las aplicaciones de toma de notas analizadas

Como podemos observar, ninguna de las opciones analizadas cubre todas las características deseables de ser implementadas. Aunque Notion y OneNote son aplicaciones muy avanzadas con cientos de funcionalidades adicionales que las hace actualmente las mejores del sector, ambas carecen de una buena combinación entre documentación y dibujo; este hecho aleja a los usuarios de cada bando, haciendo que el trabajo a desarrollar en este TFG sea una buena oportunidad para cubrir sus necesidades.

Queremos que la aplicación aquí desarrollada destaque entre usuarios con distintos propósitos o no enfocados en el sector de las tabletas digitales. Aunque las tabletas son los dispositivos que sacan mayor rendimiento a las aplicaciones de toma de notas (predominando el dibujo), en este trabajo queremos desarrollar una aplicación web que pueda ser utilizada en todo tipo de dispositivos, por ejemplo, sacándole también partido a los ordenadores donde predomina la escritura por teclado.

2.3. TECNOLOGÍAS Y HERRAMIENTAS

En este apartado se describirán las herramientas y tecnologías que deberán ser utilizadas para implementar la aplicación de toma de notas usando tecnologías web.

2.3.1. DESARROLLO DE APLICACIONES WEB

A la hora de desarrollar una aplicación web debemos elegir el lenguaje que más se ajuste a nuestros objetivos y que optimice el desarrollo de este. Dentro de los lenguajes de programación web tenemos dos categorías, una son los lenguajes del lado del servidor, en los que el código se ejecuta en el propio servidor antes de ser enviado y mostrar la página en el cliente. Ejemplos de estos lenguajes son PHP [11] y ASP [12]. Por otro lado, tenemos los lenguajes del lado del cliente, cuyo código lo ejecuta el propio sistema del usuario a través del navegador web. Por ejemplo, JavaScript [13] sería uno de los lenguajes más utilizados en este contexto.

No siempre se trata de elegir uno u otro, porque ambos tipos cumplen funciones específicas y dependiendo de la aplicación puede requerir el uso de ambos a la vez. Sin embargo, para nuestra aplicación el peso de la programación residirá en el *frontend*, por ello nos bastará con el uso de un lenguaje del lado del cliente y en concreto de JavaScript. Esto es debido a que los lenguajes del lado del servidor suelen utilizarse cuando hay que gestionar mucha información con una base de datos y para esta aplicación no va a ser crítico. Enfocar la mayor parte de los esfuerzos al desarrollo de un *frontend* sí va a ser necesario, ya que vamos a necesitar que la web sea lo más dinámica posible, de forma que parezca que estamos usando una aplicación de escritorio.

2.3.2. FRAMEWORKS DE DESARROLLO FRONTEND: REACT

Para conseguir que una página web parezca una aplicación de escritorio y podamos actualizar cualquier elemento en el *frontend* de forma asíncrona, necesitamos JavaScript. Aunque JavaScript es prácticamente el único lenguaje web de los orientados al cliente, no es el único método que tenemos para realizar el desarrollo.

Existen innumerables *frameworks* que facilitan el desarrollo y mejoran significativamente la aplicación. Entre los *frameworks* actualmente disponibles destacamos los siguientes:

- **Angular** [14]**:** es un *framework* desarrollado por Google de código abierto, que usa la variante de JavaScript, llamada TypeScript para el tipado de variables, que se organiza en componentes. Angular se basa en el modelo-vista-controlador, y destaca por su potencia en el desarrollo web, enfocándose en aplicaciones medianas y grandes.

- **Vue** [15]**:** es otro *framework* centrado en el desarrollo *frontend* y que se caracteriza por su implementación progresiva, que permite añadir funcionalidades de forma sencilla conforme se van requiriendo en el desarrollo.

- **React** [16]: aunque estemos hablando de *frameworks*, React se define como una biblioteca para construir interfaces, que incluye un sistema para desarrollar componentes reactivos. Hoy en día, y tras varios años, React es la librería más popular y usada del mundo para JavaScript, tiene detrás una gran comunidad que la mantiene y la hace una gran opción para el desarrollo de aplicaciones pequeñas, medianas o grandes.

Para el desarrollo de la aplicación he decidido usar React, ya que tiene una curva de aprendizaje más sencilla que otros *framework* como Angular, e incluye una gran flexibilidad y escalabilidad más que suficiente para lo que vamos a hacer.

Vue es otra buena opción que cumple perfectamente con nuestras necesidades, pero hasta hace poco no tenía conocimiento de él como de React. Personalmente, React es una de las tecnologías con las que quería profundizar debido a su popularidad entre las aplicaciones web.

En la Ilustración 7 se muestra un ejemplo de código con esta tecnología. Tras aprender un poco como funcionaba dediqué este trabajo a conocerla y utilizar en mayor detalle. Además, este proyecto se ajustaba perfectamente a ella.

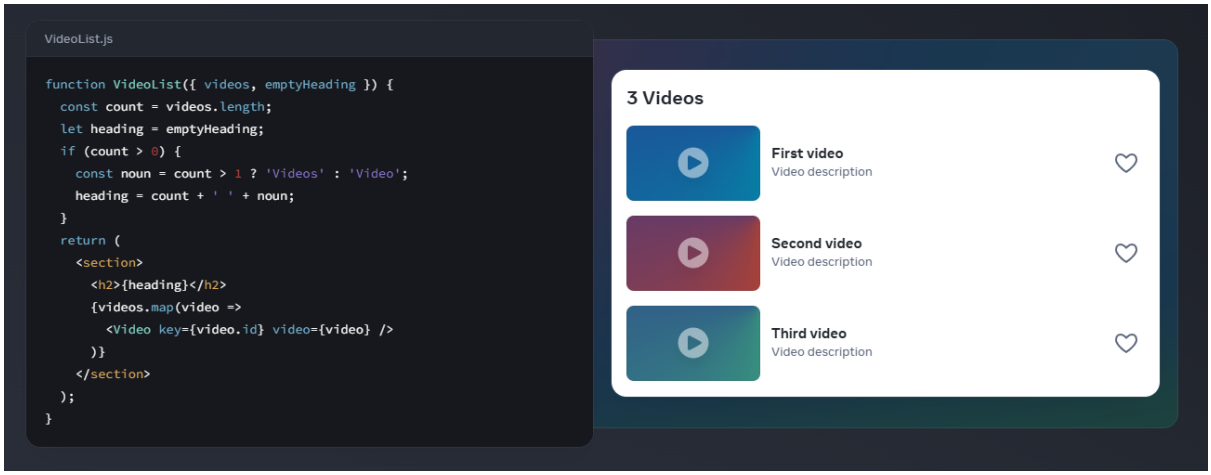


Ilustración 7: Ejemplo de código en React

Entre las principales características a destacar de React encontramos:

- **Componentes reutilizables**: los componentes son funciones JS con bloques de código HTML, que pueden ser usados en cualquier parte de la aplicación.
- **JSX**: es el tipo de archivo que utiliza React, que permite combinar HTML y JS en la misma página.
- **Estados**: los estados son variables que podemos implementar en el código HTML. React las usa para actualizar únicamente la parte del código que ha sido modificada debido al cambio de valor de esta, sin tener que actualizar toda la página. Los hooks son componentes que guardan un estado.
- **Buena documentación y comunidad activa:** cuenta en su página oficial con una mucha documentación donde se explica todas sus funciones al detalle y con ejemplos.

2.3.3. GESTIÓN DE DATOS: FIREBASE

Firebase [17] es una de las plataformas de Google, y se enfoca en ofrecer un gran número de herramientas para el desarrollo de aplicaciones móviles, aunque actualmente también permite el desarrollo en otros dispositivos y plataformas como son las aplicaciones web.

Algunas de las herramientas que integra y usaremos en este proyecto son:

- **Cloud Storage:** permite el almacenamiento de archivos en la nube como son fotos y video, pero en este caso es muy útil para almacenar el tipo de archivo que guardará la aplicación.

- **Realtime Database:** base de datos NoSQL en formato JSON que permite alojar datos de la aplicación en la nube y sincronizar los cambios en tiempo real con todos los usuarios.

Como vemos, el uso de esta tecnología también nos libera de tener que implementar la parte de persistencia de datos en el *backend*. En este caso, se delega su almacenamiento en esta librería de terceros simplificando el mantenimiento de nuestro sistema.

2.4. LIBRERÍAS DE TERCEROS

Las librerías de terceros son fragmentos de código ya programados por un desarrollador que permiten ser integrados en los proyectos de otros programadores, facilitando la implementación de una función o varias funciones y ahorrar en tiempo y líneas de código.

2.4.1. REACT MARKDOWN

Esta librería [18] se integra con los componentes de React y ofrece básicamente la opción de transformar texto simple al formato Markdown (ver Ilustración 8). Esto nos permite que un simple cuadro de texto pase a ser un documento con muchas posibilidades de personalización. Algunos ejemplos serían la inclusión de listas de tareas, código formateado con soporte de múltiples lenguajes, enlaces, listas, formateo de texto en negrita, cursiva, títulos, etc. El uso de esta librería ha sido clave en el desarrollo del proyecto, ya que se simplifica el uso de componentes nativos en la GUI, a la vez que los dota de mayor integración por usar un lenguaje, Markdown, estandarizado y ampliamente utilizado por la comunidad.

Markdown es un lenguaje de marcado ligero que se ha vuelto muy popular en los últimos años. Aplicaciones como GitHub lo integran en su página para la edición de texto en los archivos README o en las incidencias que tanto lo caracterizan. Pero GitHub no es el único proyecto que lo contiene, debido a su gran potencial y simpleza para realizar documentos con el uso de símbolos, que definen el formato de una línea o palabra. Esto lo hace una buena opción para integrar en este TFG.

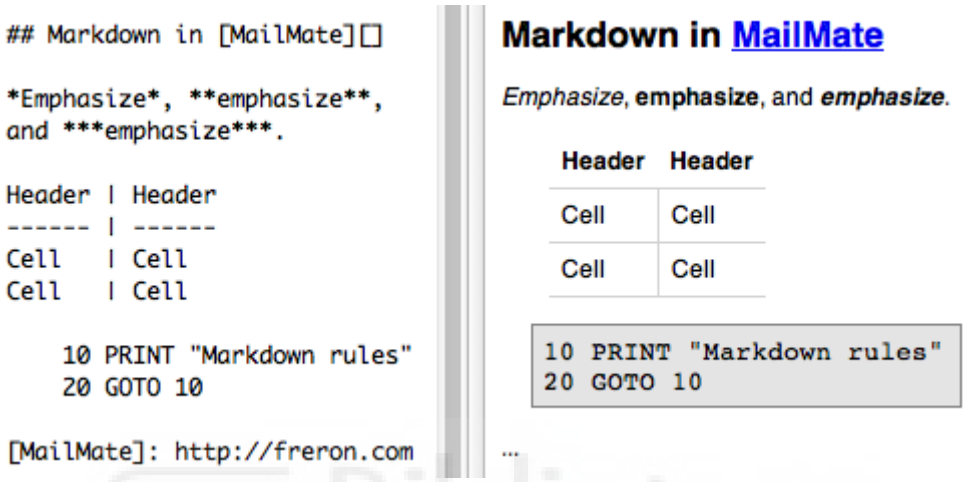


Ilustración 8: Ejemplo de uso del lenguaje Markdown

2.4.2. REACT SPEECH RECOGNITION

React Speech Recognition [19] es una librería específica para React que proporciona una serie de funciones para convertir a texto el audio que recibe desde el micrófono del dispositivo que lo ejecuta. Además, soporta la ejecución de comandos por voz y su uso con múltiples lenguajes, como es el español, que lo usaremos en este proyecto para realizar una herramienta de seguimiento de reunión. Esta nos permitirá grabar toda una sesión en el trabajo o en la universidad y convertirla a texto.

2.4.3. IONICONS

Ionicons [20] es una librería del grupo Ionic que proporciona un gran número de iconos (ver ilustración 9) para su integración en cualquier proyecto de programación de forma gratuita y sencilla, debido a que al realizar la importación de la librería solo requiere

del uso de un nuevo elemento llamado *ion-icon* y el nombre del icono para mostrarlo en la página.

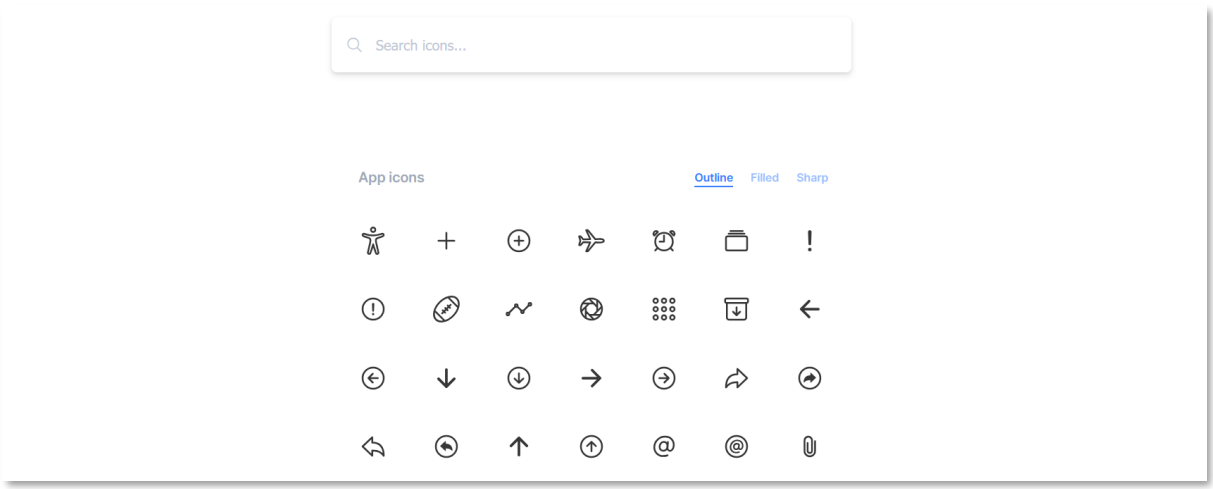


Ilustración 9: Página con iconos seleccionables de ionic (Ion Icons)

2.4.4. PUTER.JS

Puter.js es una librería de JavaScript que proporciona de forma gratuita la integración con un servicio *chatbot* de inteligencia artificial como es GPT-4, sin la necesidad de *backend* ni configuraciones adicionales. La librería es compatible con muchos otros modelos como es GPT-4o mini (ver ilustración 10).

Chat with GPT-4o mini

```
<html>
<body>
    <script src="https://js.puter.com/v2/"></script>
    <script>
        // Chat with GPT-4o mini
        puter.ai.chat(`What is life?`).then(puter.print);
    </script>
</body>
</html>
```

Ilustración 10: Ejemplo prompt de chatbot con Puter.js

2.5. OTRAS HERRAMIENTAS UTILIZADAS

Durante el desarrollo del proyecto requerimos de una buena gestión de nuestros ficheros y carpetas, así como de un editor de texto lo suficientemente capaz para adaptarse a las tecnologías que estamos usando y mejorar la eficiencia a la hora de programar.

Para este proyecto necesitamos algo más que un editor de texto ligero, ya que estos pueden llegar a ser caóticos en estos casos, y están más enfocados a la edición de scripts simples. Por ello, nos encontramos con los IDE o Entornos de Desarrollo Integrados, que aumentan las posibilidades en el desarrollo de software.

Entre las diferentes opciones, destacamos:

- **Reactide [21]:** es un IDE enfocado exclusivamente para React, permite la gestión sencilla del proyecto, y además integra un navegador con un servidor Node para previsualizar que estas programando en todo momento.

- **Visual Studio Code [22]:** es un IDE multipropósito desarrollado por Microsoft, que ofrece de código abierto y gratuito, para la edición de código en todo tipo de lenguajes, debido a la opción de poder incorporar infinidad de extensiones desde una tienda integrada en la propia aplicación, permitiendo la total libertad de diseño a gusto del usuario. En la Ilustración 11 se muestra su interfaz.

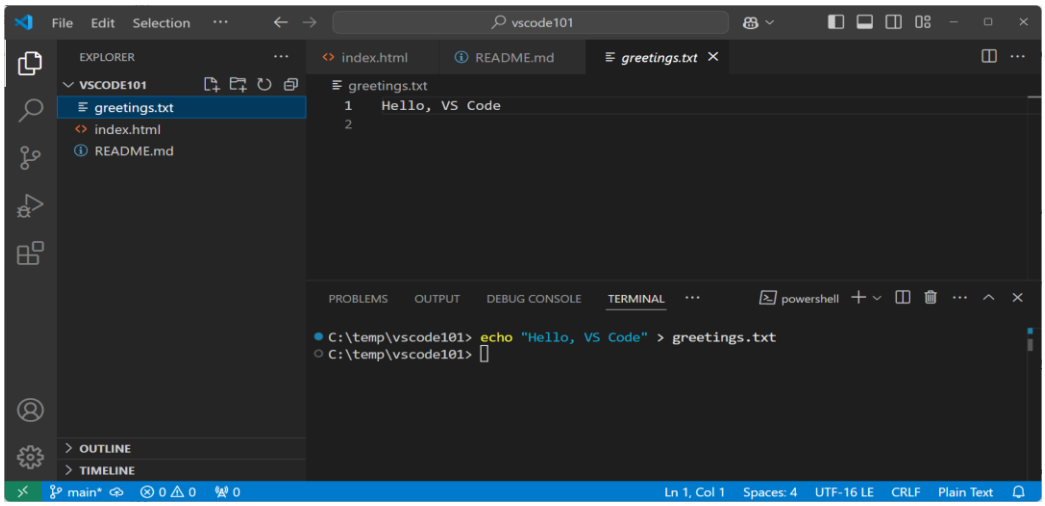


Ilustración 11: Vista de la interfaz del IDE Visual Studio Code

2.5.1. VISUAL STUDIO CODE

Aunque el uso de Reactide nos dejarían tener un entorno de desarrollo más simplificado y enfocado a la tecnología de desarrollo seleccionada, con VS Code es suficiente y es el que hemos decidido utilizar. Además, VS Code contiene *plugins* especializados para cubrir las necesidades que haya o puedan surgir, dotándola de un nivel de personalización alto.

Otro de los motivos a favor de la elección de Visual Studio Code es que, actualmente, es el editor más usado para programación en general. Es una herramienta muy recomendada para el desarrollo web y su integración con React. Por ello, creemos que se adapta perfectamente a las necesidades de este proyecto.

2.5.2. FIGMA

Figma [23] es una herramienta de diseño para la creación de prototipos de aplicaciones más enfocadas a entornos web, aunque puede usarse perfectamente en otros entornos debido a su naturaleza como editor vectorial para realizar todo tipo de diseños.

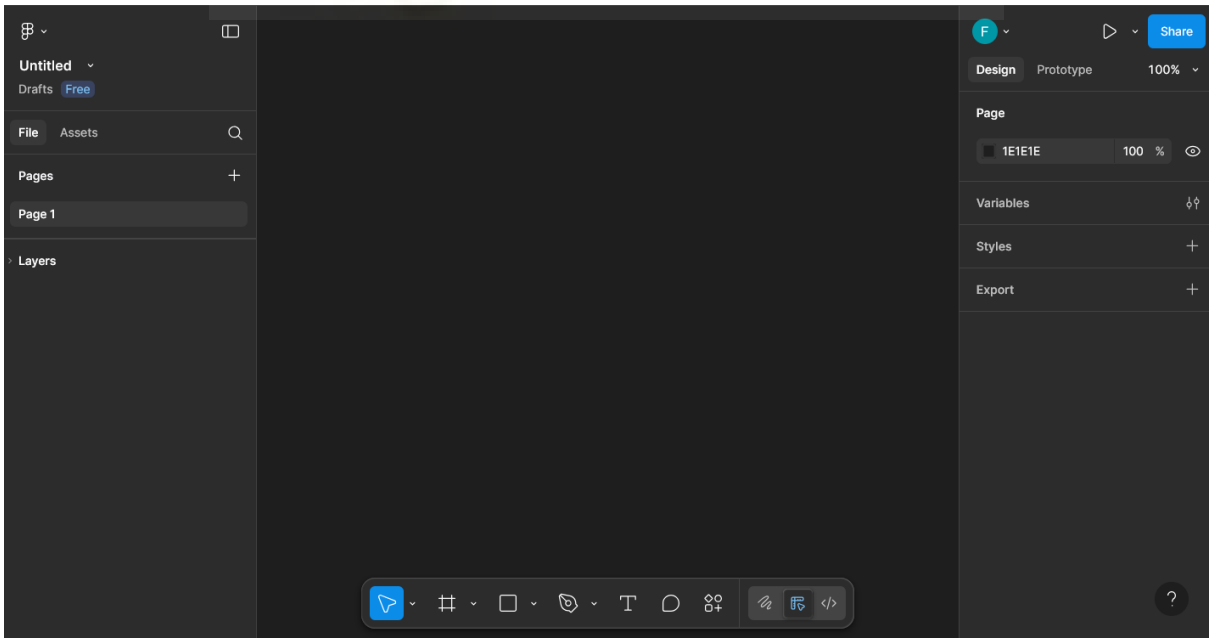


Ilustración 12: Interfaz de la página de edición de Figma

En la ilustración 12 se muestra la interfaz de Figma. Actualmente, es una de las aplicaciones más populares para prototipado. Se puede usar de forma gratuita con espacio de almacenamiento en la nube para guardar tus proyectos y poder trabajar de forma colaborativa.

2.5.3. INKSCAPE

Inkscape [24] es un editor de gráficos vectoriales, que destaca por ser una alternativa a otras herramientas de forma gratuita y de código abierto para Windows, Linux y MacOS. En la ilustración 13 podemos ver la interfaz de la aplicación

Esta sigue un desarrollo colaborativo muy activo y se actualiza continuamente con nuevos cambios y funcionalidades para adaptarse a las exigencias que van apareciendo con el paso de los años. En nuestro proyecto lo hemos utilizado para desarrollar algunos recursos gráficos que integra nuestra aplicación como el logotipo.

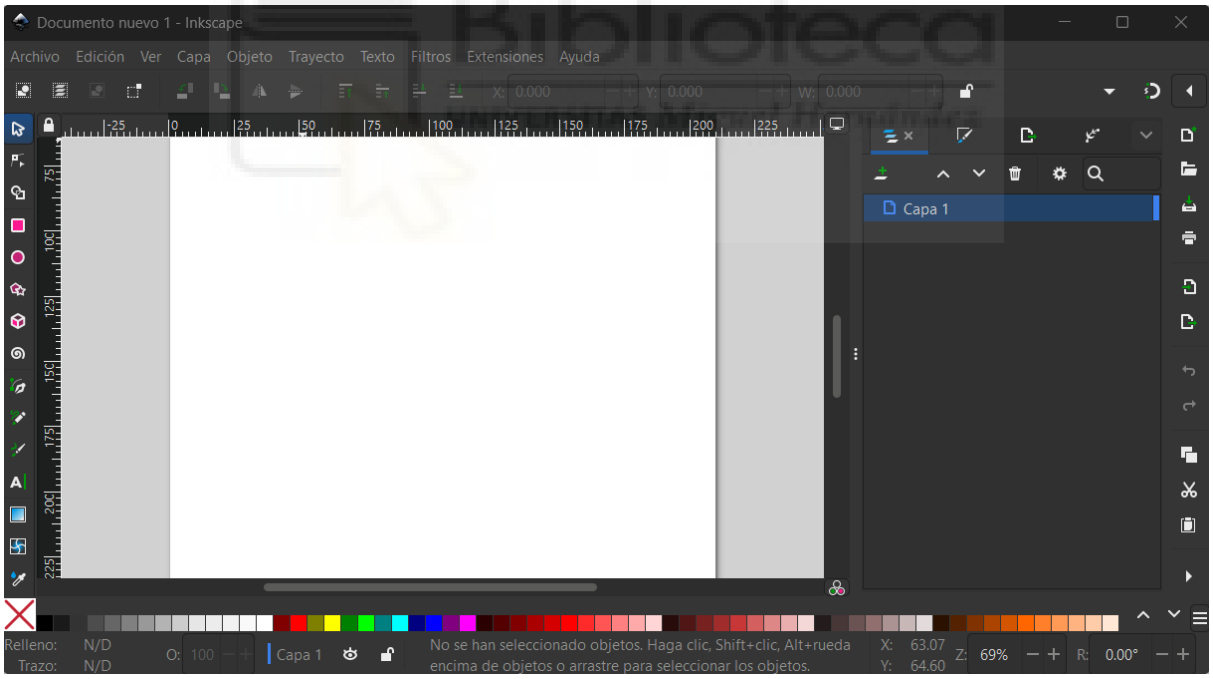


Ilustración 13: Interfaz Inkscape

2.5.4. TRELLO

Trello [25] es una herramienta gratuita y fácil de usar que se basa en la metodología Kanban. Esta aplicación integra una vista de tarjetas en un tablero, permitiendo el trabajo colaborativo entre los miembros de la empresa para la gestión de tareas y proyectos de manera eficiente. En la Ilustración 14 se muestra un ejemplo de su uso. En este proyecto se ha utilizado para la gestión del proyecto y sus tareas.

Ilustración 14: Ejemplo de uso aplicación Trello

2.6. PROPUESTA DE SOLUCIÓN

En la Ilustración 15 se resumen la arquitectura a desplegar, el tipo de recursos generados y las tecnologías utilizadas para ello.

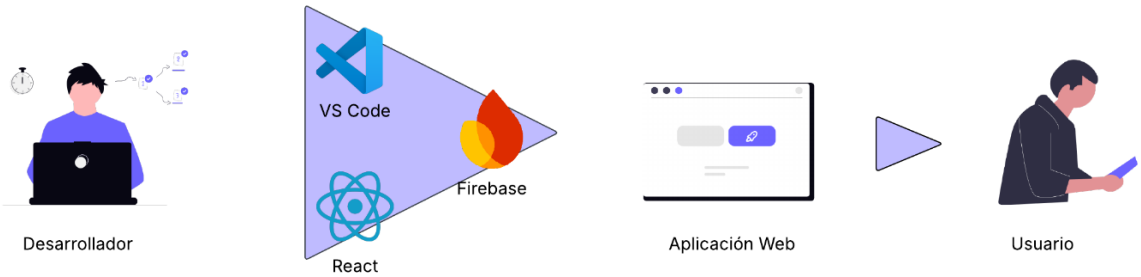


Ilustración 15: Diagrama del desarrollo de la aplicación

3. RESULTADOS

En este apartado se resumirán los principales resultados obtenidos. También se describirán aspectos relacionados con la planificación del proyecto y su ejecución.

3.1. PLANIFICACIÓN DEL PROYECTO

A la hora de planificar un proyecto se busca mejorar la productividad y llevar un control en todo momento de lo que se está haciendo. Una mala planificación aumenta la probabilidad de fracaso durante la ejecución del proyecto.

Existen múltiples métodos como los Ciclos de Vida [26] que definen una serie de etapas y cómo llevarlas a cabo. El objetivo de ciclos de vida es reducir el trabajo innecesario y ser más eficientes. Para este proyecto, no se va a usar un ciclo de vida porque suelen ser métodos muy rígidos y para un proyecto pequeño con solo una persona se queda un poco grande.

Una alternativa a los ciclos de vida son las metodologías ágiles [27]. Este tipo de metodologías permiten establecer etapas de desarrollo en ciclos cortos, de forma que, cada etapa puede ser implementada y probada al finalizarla, lo que resulta ideal para el control que se busca llevar a cabo en este proyecto. En la Ilustración 16 se muestran un resumen de las principales etapas de este tipo de metodología.

Ilustración 16: Metodologías Ágiles

3.1.1. KANBAN

Al igual que con los ciclos de vida, en las metodologías ágiles también existen varios métodos muy útiles que nos pueden ayudar en la organización de un proyecto. En concreto, en este trabajo hemos elegido Kanban [28].

La traducción de Kanban significa letrero o tarjetas de señal. Es el sistema característico que muchas veces podemos ver en empresas donde emplean *post-it* en una pizarra para organizar las tareas del proyecto que están llevando a cabo. En la Ilustración 17 se puede ver un ejemplo del uso de esta metodología.

Ilustración 17: Aplicación del modelo Kanban en la vida real

El método Kanban permite llevar un control visual del estado de las tareas, que representamos con tarjetas. Las tarjetas se organizan sobre unas columnas que representan las diferentes etapas del proyecto. Cada etapa define el estado en el que se encuentran las tareas y su finalización supone un cambio a la etapa siguiente.

Algunas de las etapas que se han definido en este proyecto son:

- **Backlog**: tareas por hacer que todavía no son principales.

- **Por hacer**: tareas a la espera de ser desarrolladas.
- **En desarrollo**: tareas que están siendo desarrolladas.
- **Pruebas**: tareas desarrolladas pero que deben ser probadas.
- **Listo**: tareas finalizadas.

Hoy en día, este método se puede aplicar para el desarrollo de software, sin necesidad de una pizarra y *post-it*s. Existen programas muy completos que implementan estas funciones para representar la pizarra y definir cualquier etapa y tarea que queramos, como es el caso del mencionado Trello.

3.1.2. DIAGRAMA DE GANTT

En la Ilustración 18 se muestra un diagrama de Gantt del desarrollo de nuestro proyecto. Como se puede ver en la primera columna se describen las principales actividades realizadas. La realización de este TFG se ha extendido desde enero a junio de 2025 con una dedicación a tiempo parcial.

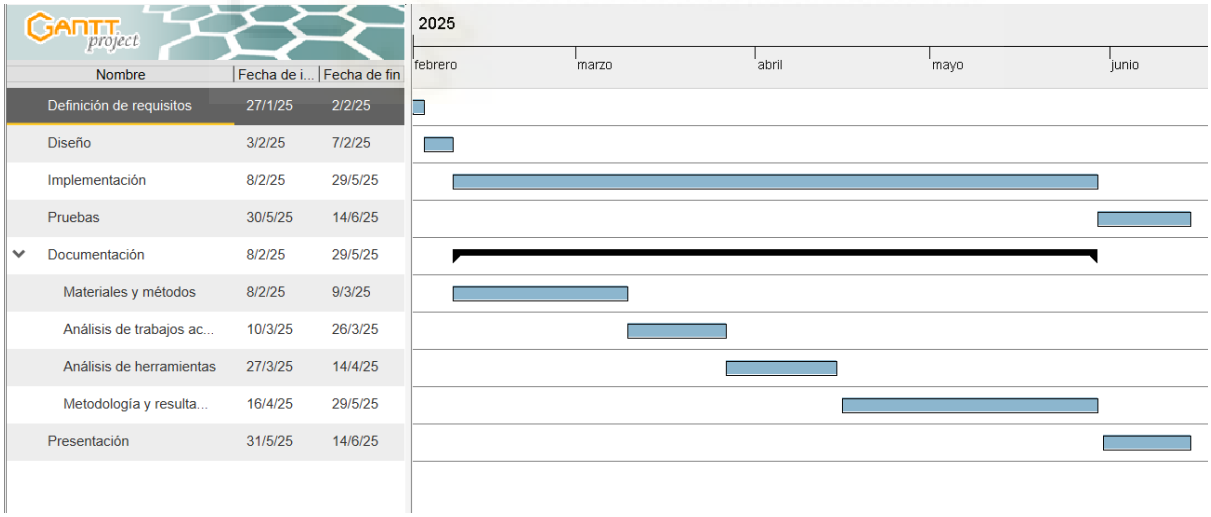


Ilustración 18 Diagrama de Gantt del proyecto

3.2. ANÁLISIS Y DISEÑO DEL SOFTWARE

Tras identificar los requisitos y describir los casos de uso, procedemos con el apartado de diseño de software. Aquí abordaremos tanto el diseño de la base de datos como el de la interfaz gráfica de la aplicación.

3.2.1. ANÁLISIS DE REQUISITOS

Como hemos comentado, para la gestión del proyecto vamos a usar el método Kanban usando la herramienta Trello.

Como se observa en la Ilustración 19, hemos registrado en Trello los requisitos como funcionales y no funcionales. Además, se han definido dos etapas principales: (1) "*Backlog*" para tareas que no son prioritarias, y (2) "*Por Hacer*" para las tareas que vamos a empezar desarrollando.

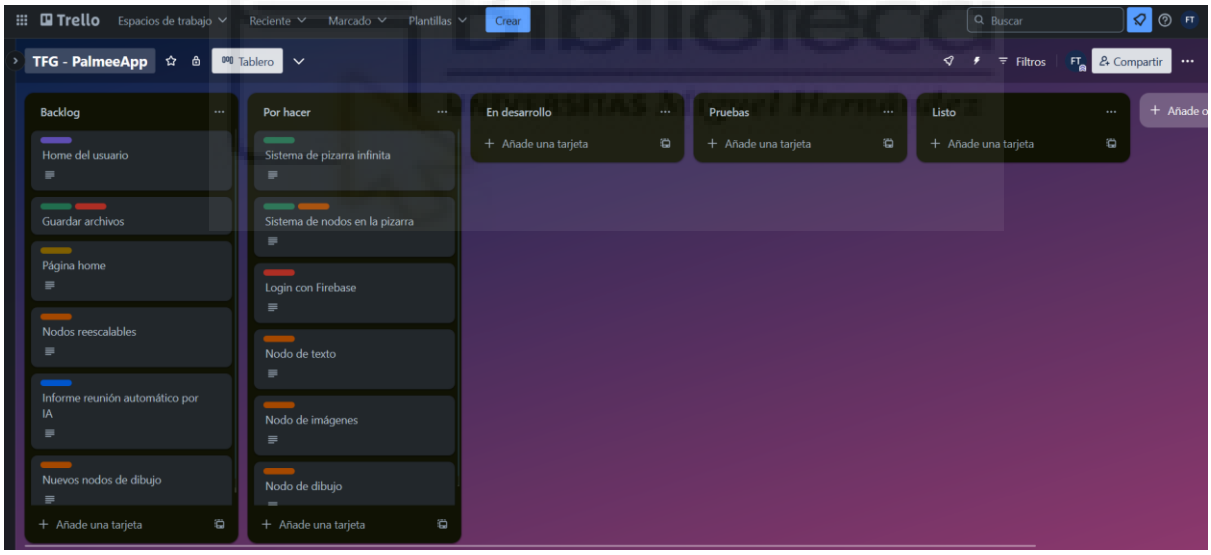


Ilustración 19: Organización del proyecto con Trello

3.2.1.1. REQUISITOS FUNCIONALES

En las Tablas 2-8 se describen con mayor detalle los diferentes requisitos funcionales identificados.

RF-1	Pizarra infinita
Descripción	Es la implementación del sistema que utilizan los editores de diagramas para tener un espacio de trabajo ilimitado.

Tabla 2 Requisito funcional "Pizarra infinita"

RF-2	Nodos
Descripción	Son los contenedores vacíos que se usan dentro de la pizarra del editor, pueden ser controlados a gusto del usuario e incluyen otros contenedores predefinidos: • **Nodo de texto**: Editor de texto con Markdown. • **Nodo de imágenes**: Visualizador de imágenes. • **Nodo de dibujo**: Pincel de dibujo.

Tabla 3 Requisito funcional "Nodos"

RF-3	Página de home
Descripción	Página principal y de presentación de la aplicación.

Tabla 4 Requisito funcional "Página de home"

RF-4	Página home del usuario
Descripción	Página para usuarios con la sesión iniciada, que permite ver los archivos guardados en la nube de Firebase.

Tabla 5 Requisito funcional "Página home del usuario"

RF-5	Login

Descripción	Sistema para iniciar sesión usando una cuenta registrada de Google.

Tabla 6 Requisito funcional "Login"

RF-6	**Guardar archivos**
Descripción	Sistema para generar un archivo JSON a partir de una pizarra con nodos que el usuario ha editado, para que posteriormente se almacene en la nube de Firebase.

Tabla 7 Requisito funcional "Guardar archivos"

RF-7	**Informe reunión por IA**
Descripción	Sistema para generar un informe dentro del nodo de texto ya definido con Markdown a partir del audio de una reunión.

Tabla 8 Requisito funcional "Informe reunión por IA"

3.2.1.2. REQUISITOS NO FUNCIONALES

De manera complementaria, en las Tablas 9-11 se describen con mayor detalle los diferentes requisitos no funcionales identificados.

RNF-1	**Interfaz simple**
Descripción	La interfaz de la aplicación debe ser simple e intuitiva para facilitar su uso a cualquier tipo de usuario.

Tabla 9 Requisito no funcional "Interfaz simple"

RNF-2	Seguridad
Descripción	El sistema debe asegurar a los usuarios que sus datos no se ven afectados ni comprometidos durante el uso de la aplicación.

Tabla 10 Requisito no funcional "Seguridad"

RNF-3	Rendimiento
Descripción	El sistema debe asegurar a los usuarios que sus datos no se ven afectados ni comprometidos durante el uso de la aplicación.

Tabla 11 Requisito no funcional "Rendimiento"

3.2.2. CASOS DE USO

Los casos de uso son el apartado donde relacionamos los diferentes requisitos y funciones que hemos estado definiendo en el proyecto. En la Ilustración 20 se muestra el diagrama general de los casos de uso (extendidos en el Anexo I).

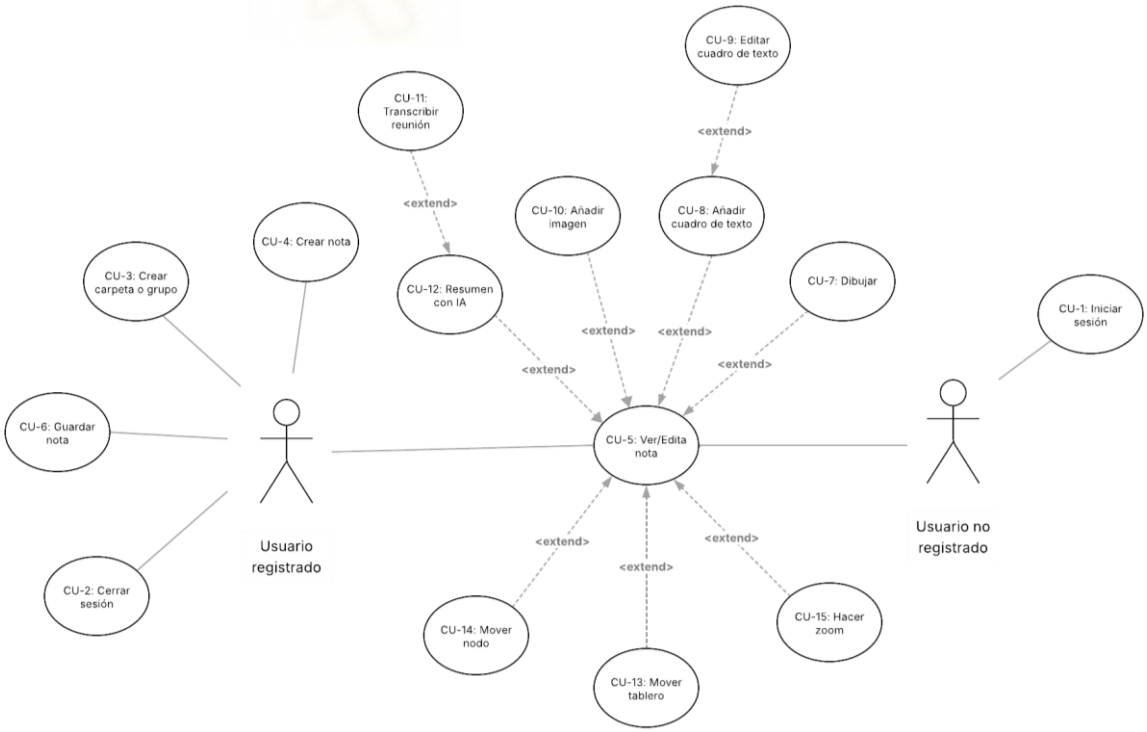


Ilustración 20: Diagrama de usuarios y casos de uso

3.2.3. ROLES DE USUARIOS

En cuanto a los roles de usuarios, como se observa en el diagrama de casos de uso tenemos definidos dos roles. Son solo dos porque la aplicación no requiere de un usuario que administre el sitio, ni un usuario con privilegios diferentes. Solo se diferenciará entre los usuarios que han iniciado sesión y los que no.

En las Tablas 12 y 13 se ofrece una descripción de cada uno de los roles definidos. Para cada rol se detallan sus capacidades y se relacionan con los casos de uso de la aplicación.

Usuario	**Usuario no registrado**
Descripción	Usuario que no se ha identificado con su cuenta de Google. Puede acceder a la página principal y al modo edición, pero no puede guardar los archivos en la nube solo en local, ni tampoco acceder a la página de usuario.
Casos de uso	CU-1, CU-3, CU-4, CU-5, CU-7, CU-8, CU-9, CU-10, CU-11, CU-12, CU-13, CU-14

Tabla 12 Descripción del rol de usuario "Usuario no registrado"

Usuario	**Usuario registrado**
Descripción	Usuario identificado con su cuenta de Google. Tiene permitido acceder a todas las páginas, así como guardar archivos en la nube de Firebase y consultar estos archivos en su página de usuario.
Casos de uso	CU-2, CU-3, CU-4, CU-5, CU-6, CU-7, CU-8, CU-9, CU-10, CU-11, CU-12, CU-13, CU-14

Tabla 13 Descripción del rol de usuario "Usuario registrado"

3.2.4. DIAGRAMA DE SECUENCIA

Con el fin de ilustrar con mayor detalle la interacción entre los diferentes elementos de nuestro sistema adjuntamos el diagrama de secuencia en la Ilustración 21.

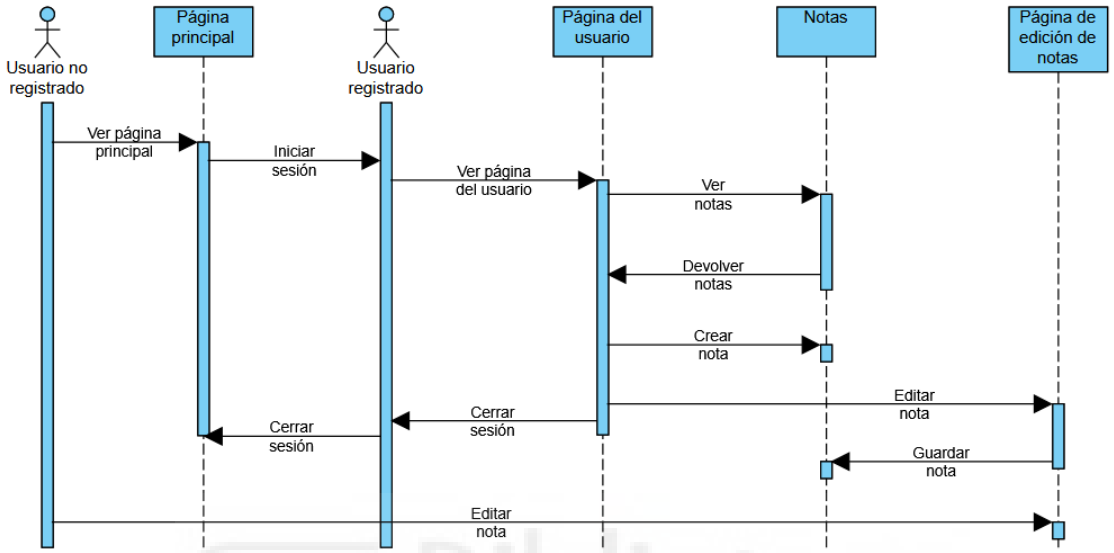


Ilustración 21: Diagrama de secuencia

3.2.5. DISEÑO DE LA BASE DE DATOS

Uno de los motivos por los que se ha optado a usar Firebase en esta aplicación es que, al usar un estándar como Markdown basado en texto para almacenar la información de las notas, las necesidades de esquemas de datos complejos se reducen ya que los tipos de datos a almacenar son pocos. Además de Markdown, el uso de Firebase permite simplificar el proceso de registro de usuarios; por ejemplo, usando Firebase simplificamos la gestión de usuarios y contraseñas al tener la identificación automática de Google. Usando estas opciones, el peso de la gestión de datos residirá principalmente en almacenar los datos de las notas que los usuarios han creado y los nodos que contiene cada nota. Esto reduce la necesidad de definir relaciones o tablas complejas.

Merece la pena destacar que la simplicidad de Firebase no implica que tengamos la puerta cerrada a futuras ampliaciones. Aunque esta plataforma usa bases de datos no relacionales (NoSQL) sin relaciones entre las diferentes tablas, siempre podríamos

relacionarlas manualmente si fuera el caso usando el formato de objetos JSON de JavaScript para estructurar los datos como listas.

A continuación, nos centramos en la descripción de nuestra tabla principal para persistir las notas. Se utilizará para ello un diseño NoSQL en formato clave-valor con JSON. En la Ilustración 22 se muestra un diagrama resumen del diseño propuesto.

Firebase estructura los datos con colecciones y documentos, de forma que cada colección guarda una lista de documentos y cada documento guarda la información de un tipo de datos. A su vez estos elementos pueden contener nuevas sub-colecciones. Esto nos lleva a crear tres colecciones anidadas para poder guardar los nodos de cada nota y las notas de cada usuario.

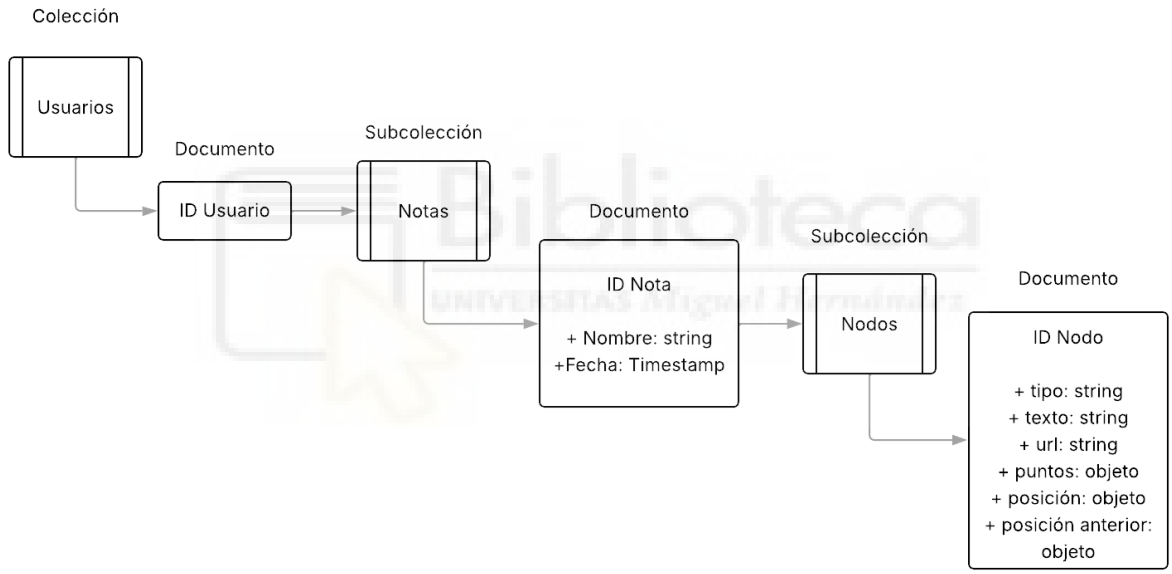


Ilustración 22 Diagrama con el diseño NoSQL de las notas

En cuanto a los campos almacenados para cada *Nota*, podemos encontrar los siguientes:

- **ID nota:** identificador único para cada nota.
- **Nombre:** título que aparece en cada nota.
- **Fecha:** fecha y hora de la última actualización.
- **Nodos:** información referente a cada nodo.

Asimismo, para cada nodo se almacenará la siguiente información:

- **ID nodo:** identificador único para cada nodo de cada nota.
- **Tipo de nodo:** describe si el nodo es de texto, imagen o dibujo.
- **Texto:** texto en formato Markdown si el nodo es de tipo texto.
- **URL imagen:** dirección URL que aparece en el nodo del tipo imagen.
- **Puntos:** lista de puntos que forman el nodo de tipo dibujo.
- **Posición previa:** guarda la posición del nodo sin transformación.
- **Posición actual:** posición final del nodo relativa a la previa.
- **Índice Z:** nivel de solapamiento entre los demás nodos.

3.2.6. PROTOTIPADO DE LA INTERFAZ DE USUARIO

Como se ha estado comentado previamente, se busca diseñar una interfaz que sea fácil de usar por cualquier usuario, aplicando un diseño minimalista con estructura y colores simples. Este tipo de diseño que se ha vuelto muy popular entre las aplicaciones de hoy en día y además nos facilitará su implementación durante el desarrollo.

La interfaz de usuario se ha desarrollado en tres fases: (1) el boceto, (2) el wireframe y (3) el prototipo. También se ha realizado un logotipo, y definido una guía de colores y la tipografía con el fin de establecer una imagen corporativa reconocible.

3.2.6.1. BOCETO

El boceto [29] es la primera representación de la idea que tenemos en mente. Con él realizaremos un dibujo simple mostrando algunas de las páginas que tendrá la aplicación, como la página de inicio (Ilustración 23), la página de usuario (Ilustración 24) y la página de edición de notas (Ilustración 25).

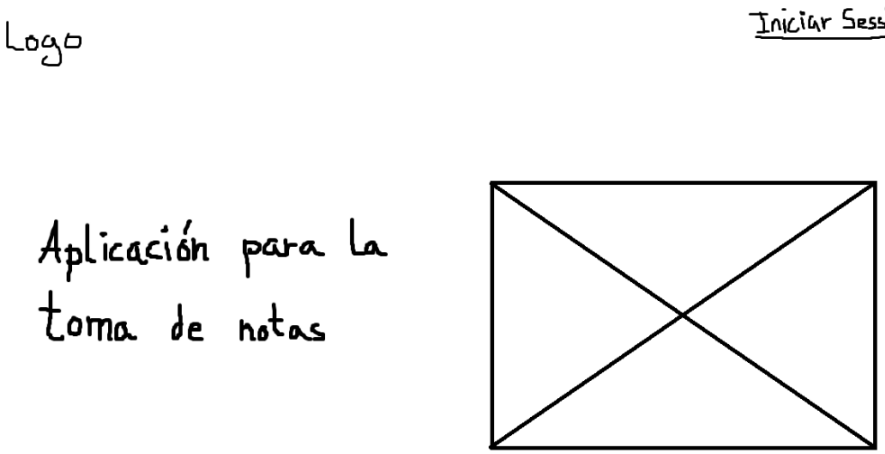


Ilustración 23: Boceto de la página de inicio

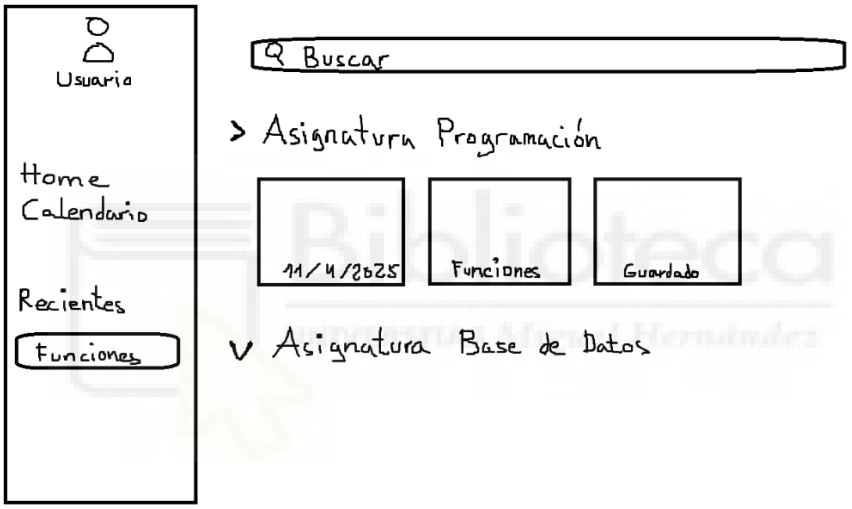


Ilustración 24: Boceto de la página del usuario

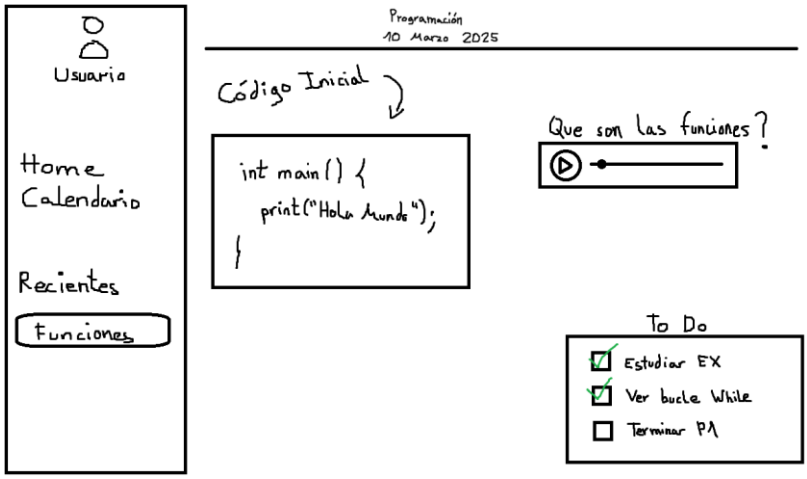


Ilustración 25: Boceto de la página de edición de notas

3.2.6.2. WIREFRAME

El Wireframe [30] es la representación a partir del boceto que realizamos usando una herramienta de diagramas que muestra con mayor detalle la idea que buscamos representar durante el diseño de la interfaz, pero sin entrar en detalles. En este apartado adjuntamos los wireframes de: la página de inicio (Ilustración 26), la página de usuario (Ilustración 27) y la página de edición de notas (Ilustración 28).

Ilustración 26: Wireframe de la página de inicio

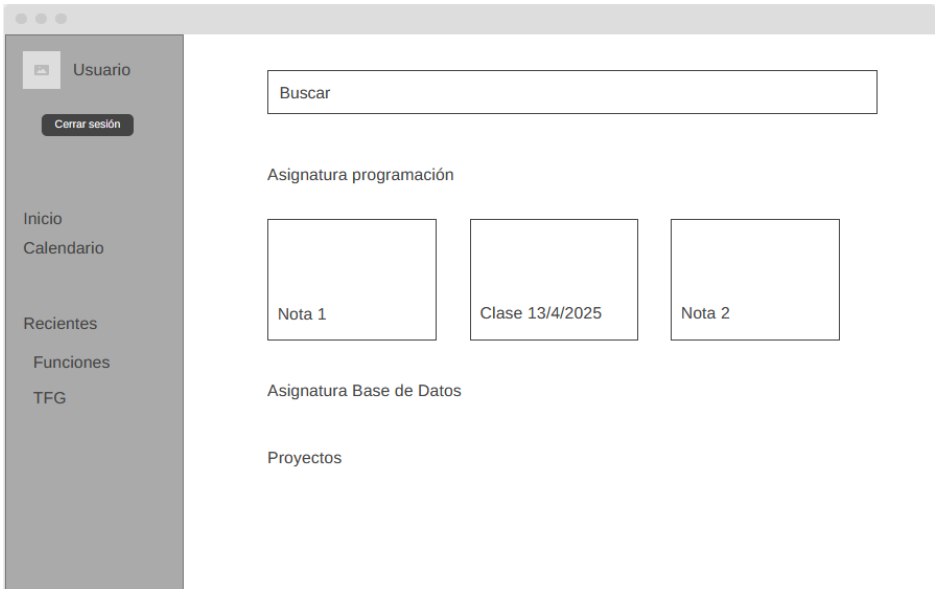


Ilustración 27: Wireframe de la página del usuario

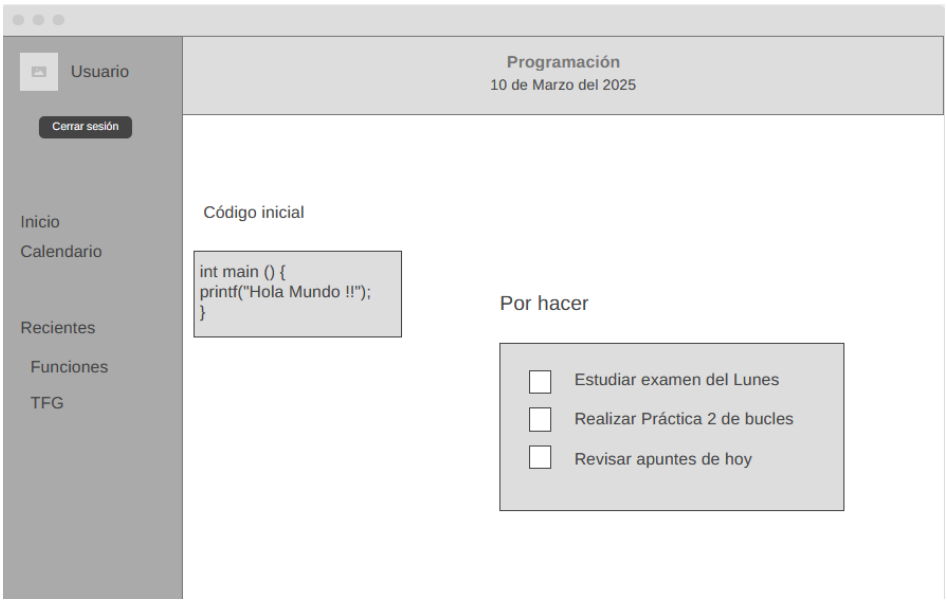


Ilustración 28: Wireframe de la página de edición de notas

3.2.6.3. PROTOTIPO

Los prototipos [31] nos permiten representar con mayor detalle los diseños trabajados anteriormente con los bocetos y los wireframes. Nuestro prototipo se asemeja mucho a cómo debería ser nuestra aplicación, incluyendo todo tipo de detalles visuales.

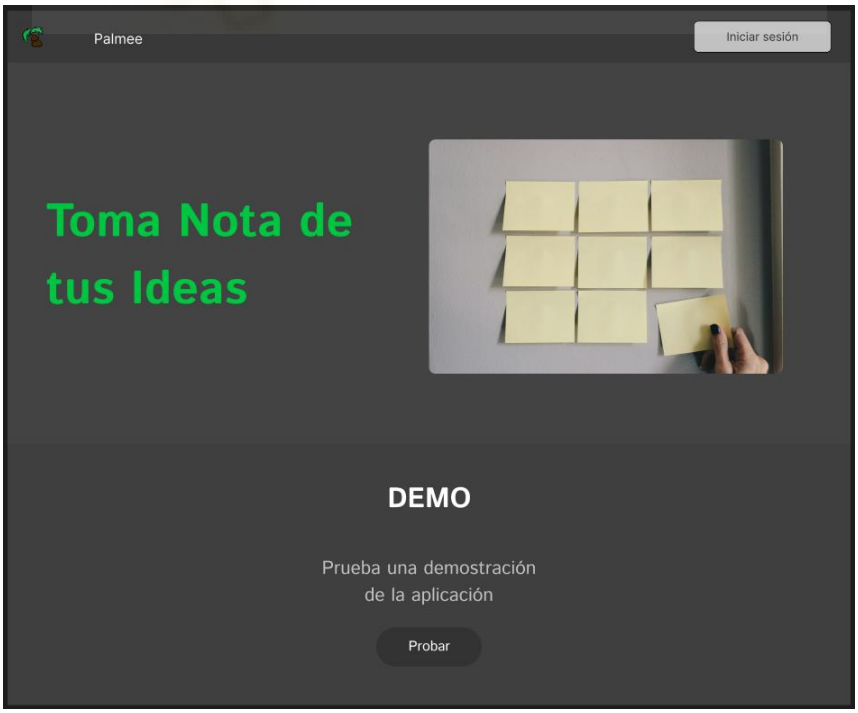


Ilustración 29: Prototipo de la página de inicio con Figma

Para realizar los prototipos podemos encontrar múltiples aplicaciones con herramientas suficientes como para incluir funcionalidad a través de un prototipo funcional. En este caso no es necesario y realizamos solo el apartado visual utilizando Figma. Siguiendo el esquema anterior, se muestran algunos prototipos generados para nuestra aplicación: la página de inicio (Ilustración 29), la página de usuario (Ilustración 30) y la página de edición de notas (Ilustración 31).

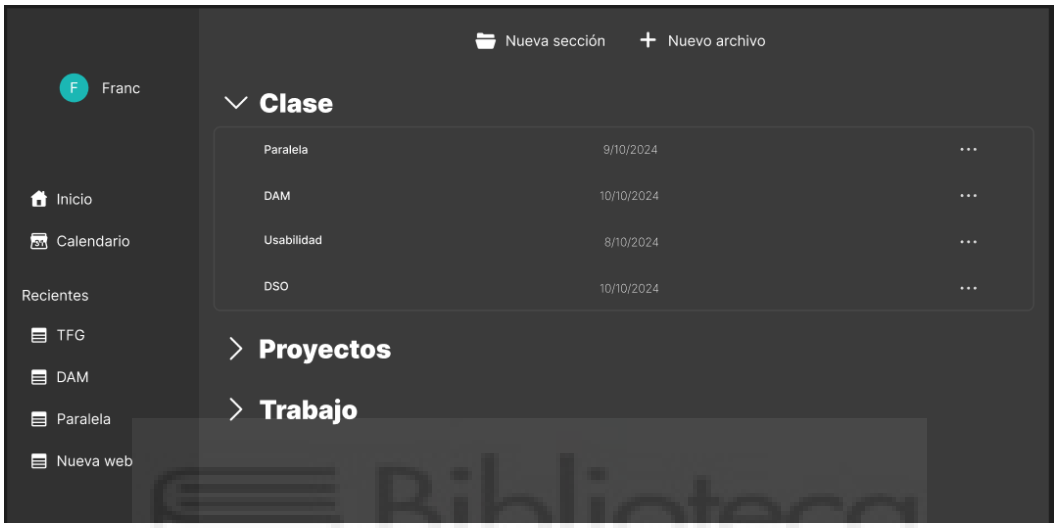


Ilustración 30: Prototipo de la página del usuario

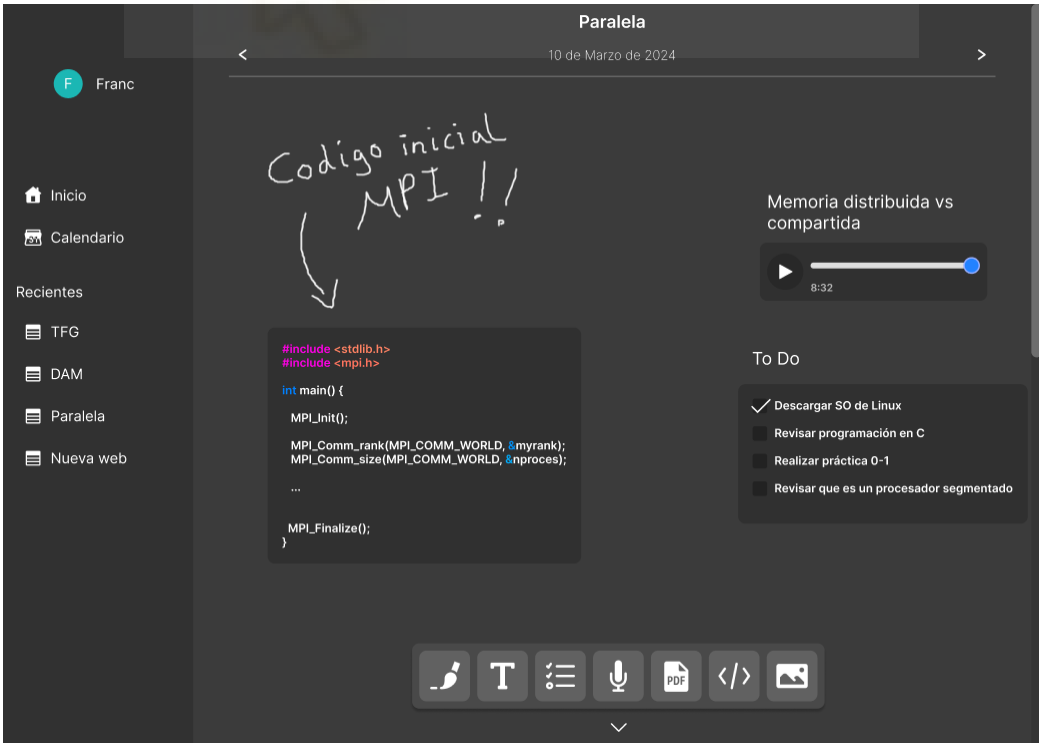


Ilustración 31: Prototipo de la página de edición de notas

3.2.7. LOGOTIPO, COLORES Y TIPOGRAFÍA

Además del diseño de las interfaces, con el fin de dotar a nuestra aplicación de identidad visual se ha realizado el diseño de un logotipo mediante el editor de gráficos vectoriales Inkscape. El logotipo se puede ver en la Ilustración 32.

Ilustración 32: Logotipo de la aplicación

Para los colores (Ilustración 33), principalmente se han usado tonos oscuros por toda la aplicación porque de esta forma evitamos posibles problemas de fatiga visual causados por la cantidad de luz que transmiten las pantallas de los dispositivos a nuestros ojos al usar tonos claros. Para el texto se ha utilizado el color blanco por defecto, pero con el color verde del logotipo y naranja para destacar títulos y palabras.

Ilustración 33: Guía de colores

Por último, en cuanto al tipo de letra de la aplicación, se ha elegido la tipografía de Quicksand para usarla en los títulos y en los textos (Ilustración 34). Esta tipografía la

podemos encontrar de forma gratuita en la tienda de fuentes de Google Fonts para importarla a nuestro proyecto con distintos grosores.

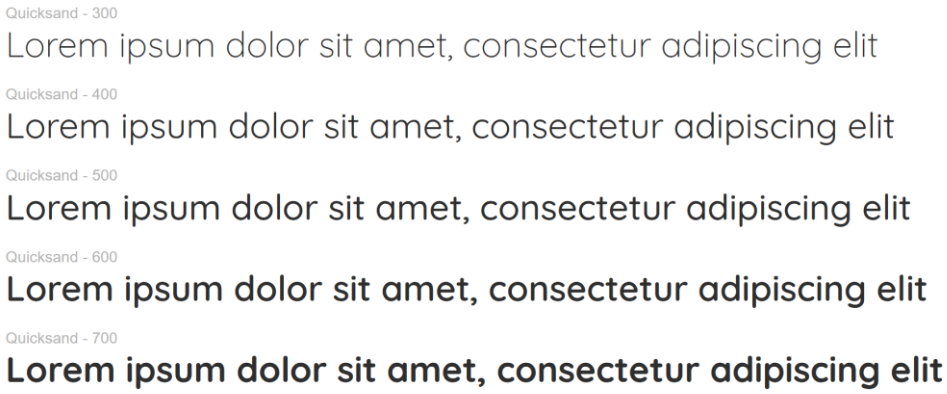


Ilustración 34: Tipografía

3.3. DETALLES DE LA IMPLEMENTACIÓN

En este apartado vamos a describir algunos aspectos relevantes de la aplicación desarrollada. Muchos de ellos están relacionados con nuestra apuesta por implementar una aplicación web minimalista e integrada con terceros. Esto ha requerido combinar la implementación con React con su integración con diferentes APIs de terceros.

3.3.1. CONEXIÓN BASE DE DATOS FIREBASE

Como hemos comentado anteriormente, Firebase es una plataforma de Google que incluye múltiples herramientas para dar servicio a una aplicación web o móvil. Para ello, Google proporciona una API y una documentación con la que poder realizar la conexión de nuestra aplicación con el servicio.

Si nos fijamos en la ilustración 35, desde la API de Firebase se proporcionan una serie de funciones para hacer el proceso más fácil. Entre ellas encontramos la primera, que es para inicializar la API a partir de una configuración proporcionada por Firebase, donde se incluye una llave única para nuestra aplicación. Después, con la API inicializada, llamamos al servicio de la base de datos, llamado Firestore, con la función "*getFirestore*", que usaremos de referencia para realizar la búsqueda de colecciones

o documentos. En el ejemplo de la ilustración 35 se obtienen todas las notas de un usuario a partir de su identificador único. Esto podemos verlo reflejado en la función llamada "*collection*", que devuelve una referencia de la sub-colección notas ubicada en el documento con el nombre del identificador único del usuario de la colección usuarios (en la ilustración aparece "*auth.currentUser.uid*" porque es el id del usuario con la sesión iniciada). Finalmente, esta referencia se usa para llamar a la función "*getDocs*" que nos devuelve todos los documentos.

```
// Initialize Firebase
const app = initializeApp(firebaseConfig);

const db = getFirestore(app);

export async function getAllNotes() {
  const notaRef = await getDocs(collection(db, "users", auth.currentUser.uid, "notes"));
  return notaRef
}
```

Ilustración 35: Código para obtener todas las notas con Firebase

3.3.2. TRANSCRIPCIÓN DE AUDIO A TEXTO

Aquí describiremos el proceso de la grabación de una sesión para poder realizar un resumen con la herramienta incorporada de resúmenes por IA. Para ello, unos de los procesos intermediarios que se realizan es la transcripción del audio a texto, de forma que sea más sencilla la interpretación de la información por parte de la IA. Para hacer esto posible, se utiliza una librería llamada React Speech Recognition que proporciona un Hook, o clase con estados, con la que poder controlar el componente.

Como podemos ver en la ilustración 36, al inicio de la función principal tenemos el Hook "*useSpeechRecognition*"; este hook nos proporciona el estado llamado "*transcript"* encargado de actualizar el texto transcrito conforme va cambiando. Más abajo en el código, tenemos una condición que inicia o para la grabación cuando pulsamos un botón en pantalla.

```
import SpeechRecognition, { useSpeechRecognition } from 'react-speech-recognition';

function IAResumeTool(props) {

    const {
        transcript,
        listening,
        resetTranscript,
        browserSupportsSpeechRecognition
    } = useSpeechRecognition();

    useEffect(() => {
        if (recording) {
            document.getElementById('recording-button').innerHTML = "<ion-icon name='pause'></ion-icon>"
            SpeechRecognition.startListening({ continuous: true })
        } else {
            document.getElementById('recording-button').innerHTML = "<ion-icon name='play'></ion-icon>"
            SpeechRecognition.stopListening()
        }
    }, [recording])

    useEffect(() => {
        document.getElementById('resume-transcript').innerHTML = transcript
    }, [transcript])
```

Ilustración 36: Código transcripción de audio a texto

3.3.3. GENERACIÓN DE PROMPTS CON PUTER.JS

Como ya se ha comentado, con Puter.js podemos generar la respuesta de un *chatbot*, como es GPT-4 a partir del *prompt* que le queramos proporcionar. Para ello, usando la transcripción de audio que hemos visto en el punto anterior podemos solicitar al *chatbot* que nos genere un resumen de la sesión. Además, se puede solicitar que lo convierta en formato Markdown para tenerlo bien integrado y documentado en la aplicación.

Como podemos ver en la ilustración 37, se muestra una función que nos devuelve la respuesta del GPT para poder usarla cuando queramos.

```
async function chatWithGPT(message) {
    const prompt = "Escribe un resumen en formato markdown del siguiente texto, sin añadir ```markdown al principio y al final. Texto: "
    const response = await puter.ai.chat( prompt + "" + transcript)
    return response.message.content
}
```

Ilustración 37: Código generación resumen con IA

3.4. INSTALACIÓN Y DESPLIEGUE

El uso de React requiere de Node.js y de una herramienta de compilación para ejecutar el código fuente, que además nos proporcione un servidor local como Vite. El uso de Vite permite realizar todo este proceso de instalación y despliegue de manera más sencilla.

Para desplegar el proyecto, deberemos hacer uso del terminal. Desde la carpeta del proyecto podremos iniciar la aplicación en el puerto local 5173 con el comando de Node.js para el modo desarrollo:

```
$$ npm run, o npm run dev
```

En la Ilustración 38 se muestra como ejemplo el uso del comando anterior y como se puede acceder en local a la aplicación desarrollada.

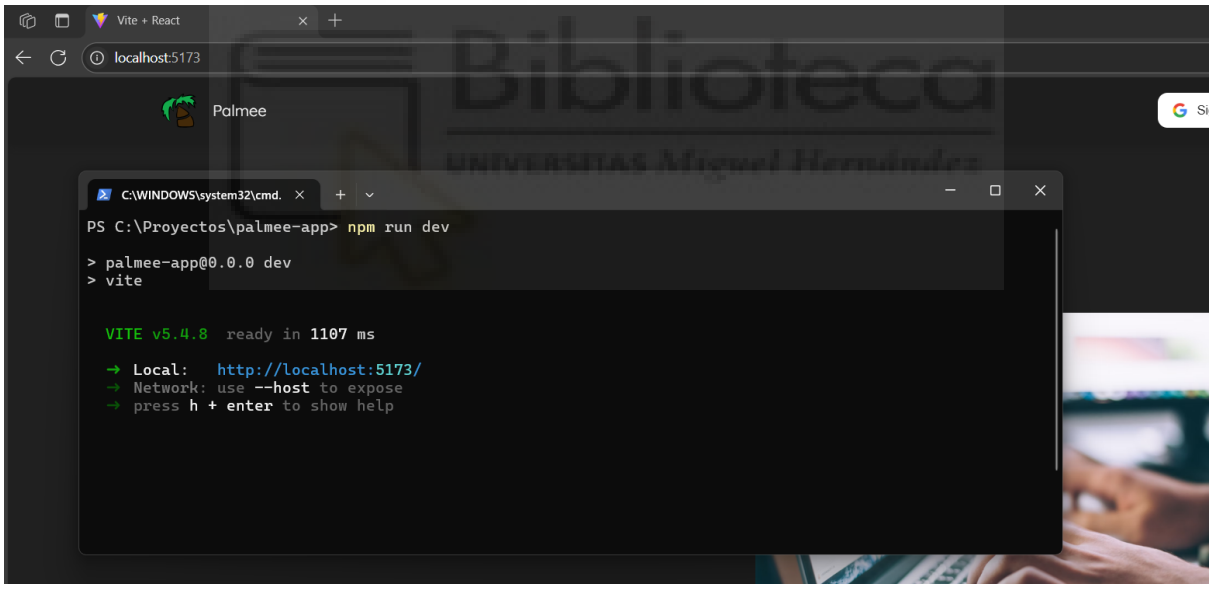


Ilustración 38: Inicio de la aplicación

3.5. EJEMPLO DE USO DE LA APLICACIÓN

En este apartado queremos describir a través de capturas de la aplicación cómo sería su funcionamiento por parte de un usuario.

3.5.1. PÁGINA DE INICIO

La primera de las tres páginas que encontraremos es la página de inicio (Ilustración 39). Desde aquí podremos tener un primer vistazo y obtener información relevante sobre la aplicación.

Como se observa desde el botón de la parte superior derecha se puede iniciar sesión usando una cuenta de Google. Además, en el caso de que el usuario ya tenga la sesión iniciada, esta página no se mostraría; en su lugar aparecerían directamente la vista con las notas guardadas por dicho usuario.

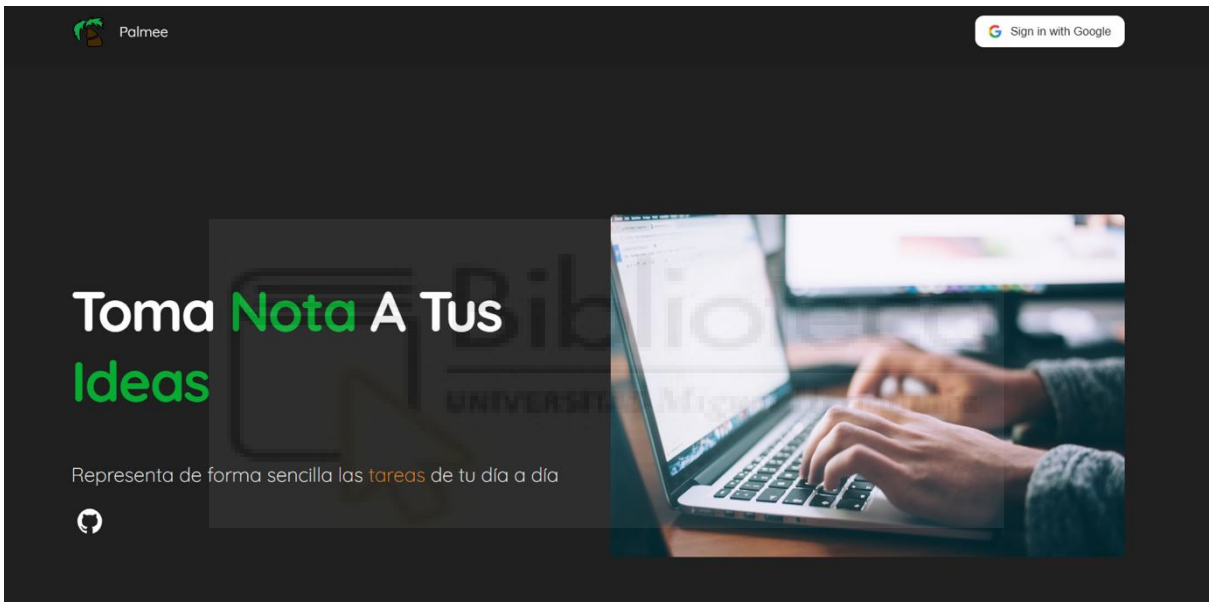


Ilustración 39: Página principal de la aplicación

3.5.2. PÁGINA DE USUARIO

Una vez el usuario se ha autenticado en el sistema, podrá acceder a la página de usuario. Como se puede observar en la Ilustración 40, en esta página se van a mostrar las notas creadas por cada usuario.

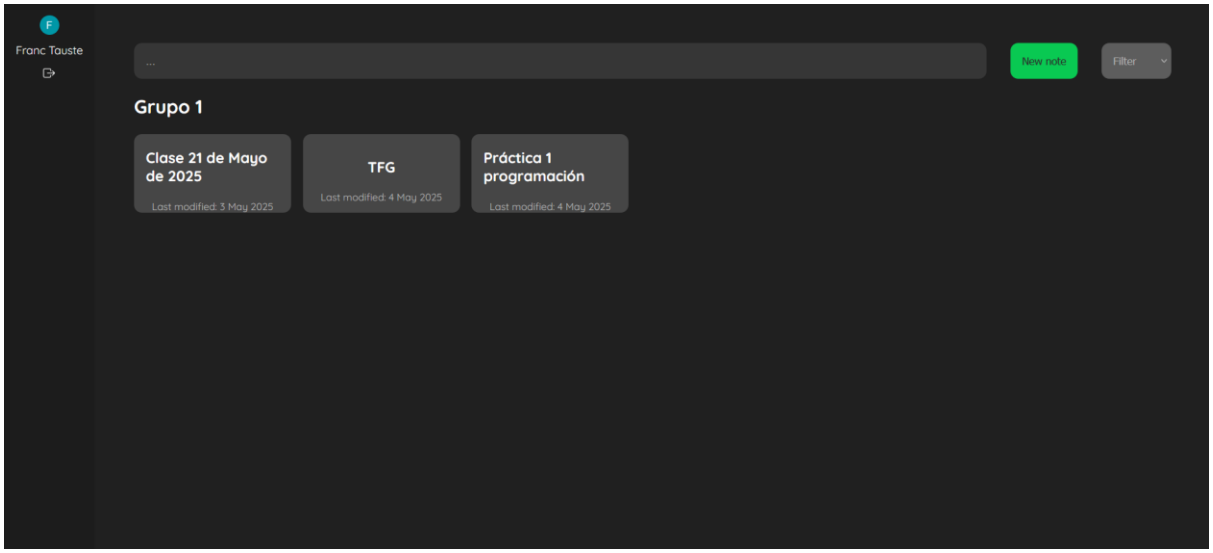


Ilustración 40: Ejemplo de las notas del usuario

3.5.2.1. AGRUPAMIENTOS DE LAS NOTAS

Las notas se pueden ordenar por varios tipos de filtros: notas recientes (según la fecha de la última actualización), por nombre (según el orden alfabético del nombre de la nota) o por grupos a los que pueden pertenecer (descrito como trabajo futuro). Para ello, en la Ilustración 41 se muestra un ejemplo de cómo podríamos realizar este tipo de filtros.

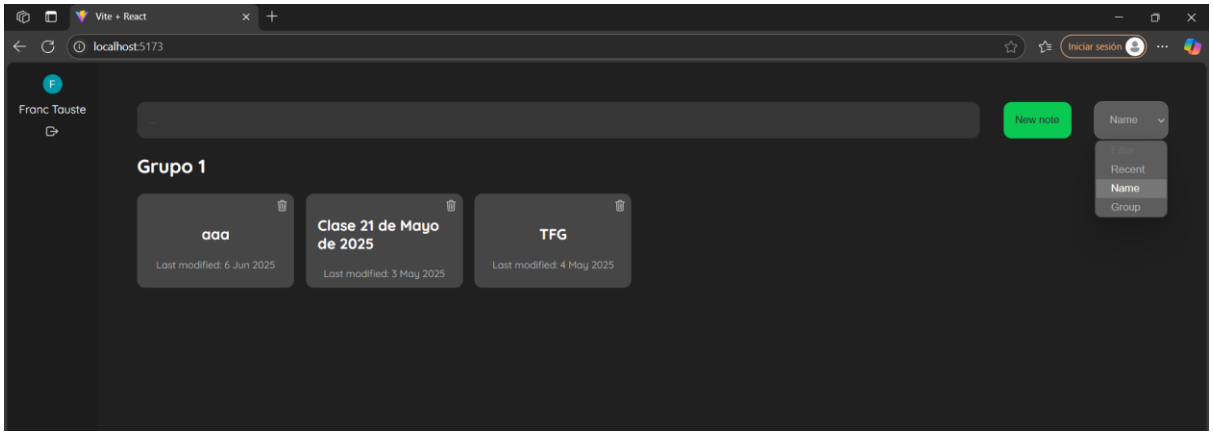


Ilustración 41: Ejemplo filtrado de notas por nombre

3.5.2.2. FILTRADO DE LAS NOTAS

Además de ofrecer el agrupamiento de notas, también tenemos una barra de buscar en la parte superior de la página. A través de este formulario se permite filtrar las notas que aparecen según el nombre introducido (ver ilustración 42).

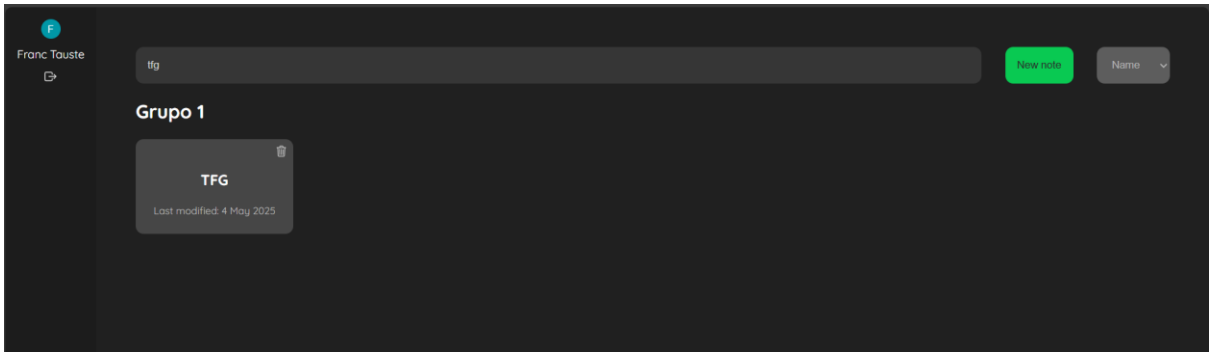


Ilustración 42: Ejemplo barra de buscar notas

3.5.2.3. ELIMINACIÓN DE LAS NOTAS

Por último, en esta página, podemos encontrar el icono de una papelera en cada nota que nos permite eliminarla. Este proceso solo ocurre a través de una confirmación para evitar posibles errores del usuario como podemos ver en la ilustración 43.

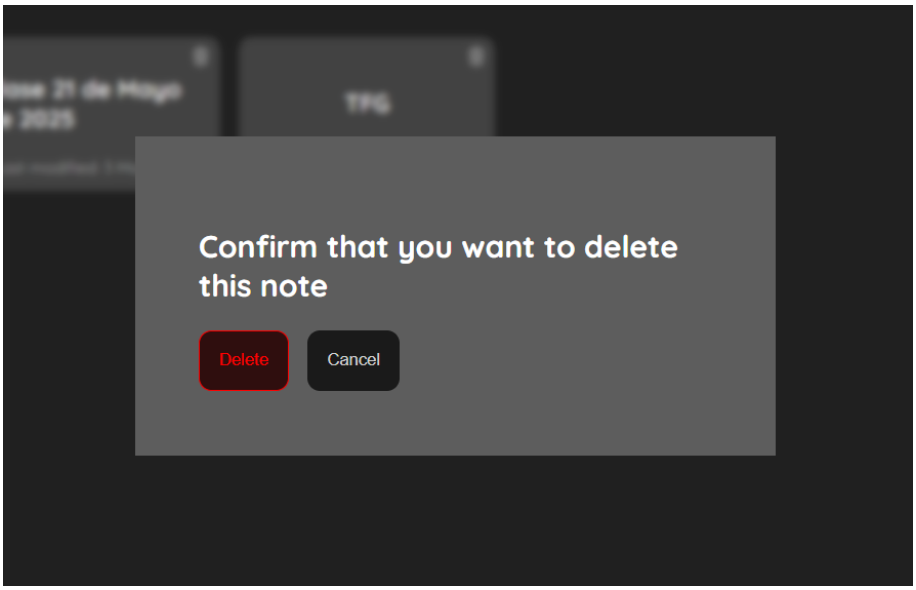


Ilustración 43: Ventana de confirmación para eliminar una nota

3.5.3. PÁGINA DE EDICIÓN DE NOTAS

Para crear nuevas notas se podrá pulsar el botón verde de la parte superior derecha de la página de usuario. La creación de notas por parte del usuario es la parte más completa. Para ello se ha implementado el editor de notas que se puede observar en la Ilustración 44.

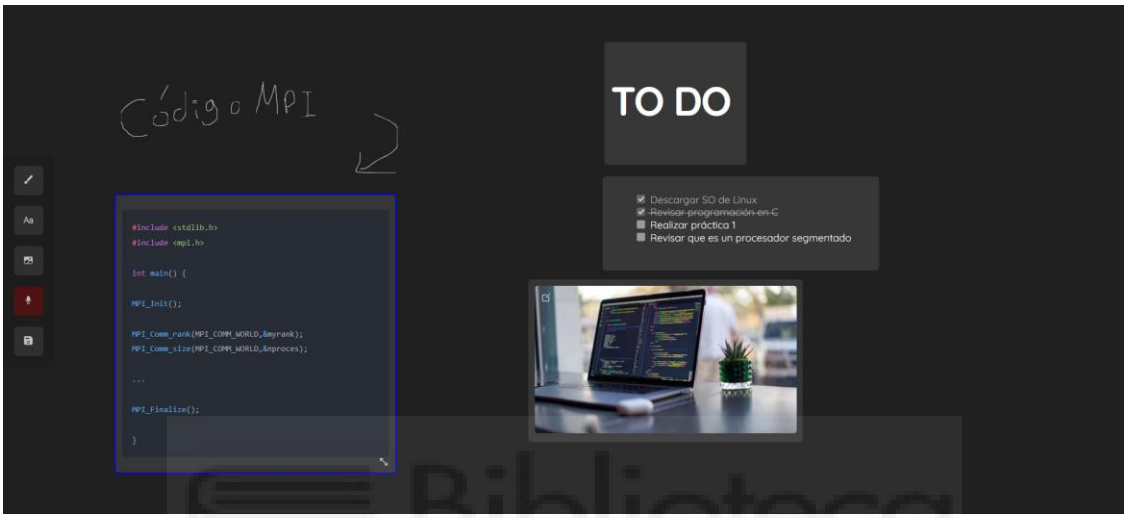


Ilustración 44: Ejemplo de una nota sobre una asignatura

El editor contiene un tablero infinito como el de los editores de diagramas, permitiendo arrastrar una serie de nodos en un espacio sin límites. Estos nodos se forman a partir de las herramientas disponibles que podemos encontrar en un panel lateral de la ventana. Entre las herramientas disponibles podemos encontrar:

- **Pincel:** permite dibujar en cualquier parte de la pantalla.
- **Cuadro de texto:** añade un cuadro de texto para escribir con teclado, pero este texto es compatible con Markdown.
- **Imágenes:** permite cargar una imagen en pantalla a partir de la dirección URL de otra imagen alojada en un servidor de Internet.
- **Resumen de reunión por IA:** incluye la posibilidad de grabar el audio de una conversación por el micrófono para transcribirlo a texto, y posteriormente, hacer un resumen por IA.
- **Guardar nota:** sube al servidor de Firebase los cambios realizados.

3.5.3.1. EJEMPLO DE PINCEL

En la Ilustración 44 se puede ver el uso de la toma de notas libre haciendo uso del pincel. En la parte superior izquierda se puede observar la anotación libre *"Código MPI"*. Se podrían tomar notas de este tipo a lo largo del lienzo infinito que crear para cada una de las notas.

3.5.3.1. EJEMPLO DE NOTA DE TEXTO

En la Ilustración 45 se puede ver de forma detallada la definición de una nota de texto usando el lenguaje Markdown. En la parte izquierda se puede ver el código fuente, mientras que en la parte derecha se pude ver como su funcionalidad quedaría representado en el modo de visualización.

Ilustración 45: Ejemplo cuadro de texto con Markdown

Como vemos, las notas de texto permiten codificar los siguientes tipos de elementos:

- **Cabeceras:** formateo de texto para títulos y cabeceras.
- **Resaltado de texto:** ofrece la posibilidad de codificar negrita y cursiva.
- **Formateo de código:** ofrece el coloreado de código para múltiples lenguajes de programación.

- **Listas:** añade lista de elementos con varios niveles. Acepta tanto listas numeradas como no numeradas.
- **Listas de tareas por hacer:** listas donde se introducen descripciones de tareas que podemos marcar como completadas.
- **Tablas:** organiza la información en filas y columnas.
- **Links**: texto que contiene un enlace a una página o recurso externo.
- **Referencias:** citas simples en el texto que permiten ser localizadas en otra sección del documento con información adicional.
- **Lista de precios:** listas con precios que se totalizan automáticamente, como por ejemplo una lista de la compra.

3.5.3.2. EJEMPLO DE IMAGEN

Otra de las herramientas que incluye la aplicación son las imágenes. Estas podemos añadirlas desde la barra de herramientas lateral. Tan solo requerirá de pulsar en el botón y posteriormente en algún lado de la pantalla. Tras ello aparecerá un nuevo cuadro de diálogo donde se deberá introducir la dirección URL de la imagen deseada como podemos ver en la Ilustración 46.

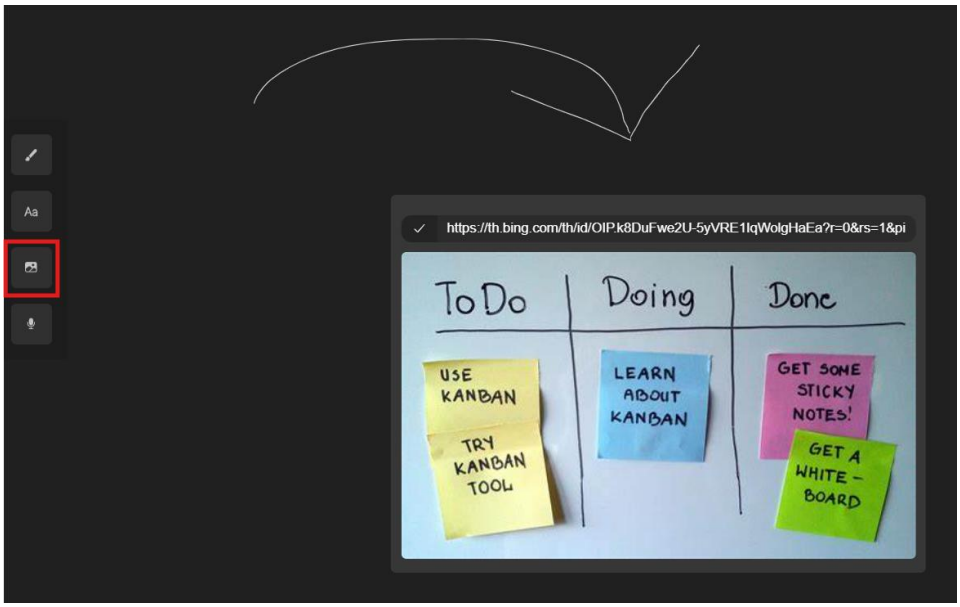


Ilustración 46: Ejemplo de la herramienta de imagen

3.5.3.3. EJEMPLO DE NOTA DE VOZ Y RESUMEN CON IA

Por último, tenemos la herramienta para generar resúmenes de una sesión con Inteligencia Artificial. Esta herramienta la encontramos en el menú lateral con el icono de un micrófono, que nos abre una nueva ventana lateral con dos botones. Uno con el icono de "*play*", que inicia la grabación del micrófono con la transcripción a texto y el otro botón que genera un resumen del texto transcrito con la ayuda de un *chatbot* con IA.

Si observamos en la ilustración 47, se ha transcrito el texto de la derecha desde el micrófono, y posteriormente se ha generado el resumen en un nuevo cuadro de texto en formato Markdown, aunque en este ejemplo el resumen generado es igual al transcrito debido a la longitud, pero podemos ver como incorpora un título y le añade la etiqueta de cabecera en formato Markdown.

Ilustración 47: Ejemplo de transcripción y resumen con IA

4. CONCLUSIONES Y TRABAJO FUTURO

A continuación, se describirán las conclusiones del presente trabajo y algunos de los posibles trabajos futuros que pueden continuar desarrollándose tras la finalización de este TFG.

Durante el desarrollo del proyecto se han cumplido los objetivos establecidos y se ha creado una aplicación para la documentación y toma notas con una interfaz y uso amigable. Algunos de los aspectos a destacar de los objetivos alcanzados son:

1. El análisis de otros trabajos académicos donde se investiga el uso de herramientas para la toma de notas ha influido positivamente en nuestro proyecto. Esto ha permitido detectar algunas funcionalidades que los usuarios destacan para dispositivos móviles, y que en nuestro caso hemos implementado en el entorno web.

2. La implementación de un sistema más elaborado para la documentación de notas usando Markdown ha permitido la incorporación de distintos formatos de texto como: cabeceras, texto en negrita, texto en cursiva, formateo de código, tablas, listas, etc. Utilizar un estándar para ello facilita el uso por parte de los usuarios acostumbrados a utilizarlo. Además, usar este lenguaje permitrá a nuestra aplicación escalar en un futuro con nuevos tipos de datos que puedan aparecer, sin que esto suponga tener que re-implementar nuestro esquema de datos basado en JSON.

3. Otro aspecto innovador de nuestra propuesta ha sido la inclusión en una aplicación web de toma de notas por voz con transcripción y resumen por IA. Estos tipos de notas facilitan por ejemplo la creación de apuntes durante una sesión sin la necesidad de dejar de prestar atención.

En definitiva, en este trabajo se ha implementado un prototipo funcional de una aplicación web de toma de notas. El hecho de ser accesible desde un navegador pretende simplificar su acceso y uso desde diferentes dispositivos, ya que no se requiere una instalación previa de aplicaciones.

El prototipo funcional desarrollado incluye funcionalidades frecuentemente demandadas por los usuarios, que han sido identificadas a través del estudio de herramientas muy populares para dispositivos móviles. El desarrollo de este trabajo ha permitido también identificar aspectos que serían interesante incluir en un futuro. A continuación, describimos algunas de estas líneas de trabajo futuro identificadas.

4.1. TRABAJO FUTURO

Principalmente la aplicación contiene las herramientas esenciales que se buscaba implementar, pero existen muchas otras que mejorarían el uso y amplificarían el alcance de la aplicación. Por ejemplo, en la toma de notas libre sería deseable incluir nuevos tipos de figuras para el dibujo (flechas, cuadrados, círculos, etc.); estos tipos fomentarían la personalización de las notas. También sería útil incluir herramientas para la gestión de proyectos como serían las tablas de tarjetas Kanban.

Otra limitación de la aplicación es la personalización de las herramientas. Por ejemplo, el pincel de dibujo no cuenta con opciones para modificar el color o la anchura. Otro ejemplo sería que el texto no se puede modificar el tamaño de letra o la tipografía. Sería interesante crear un nuevo menú donde aparecieran todas las opciones de personalización disponible.

En cuanto al trabajo colaborativo, incluir en la aplicación web esta funcionalidad sería una gran contribución para amplificar el alcance a la gestión de proyectos. Aunque no se ha incluido, el uso Firebase fue también motivado por ello ya que su base de datos permite integrar procesos que requieren modificaciones en tiempo real.

Por último, me gustaría llevar esta aplicación a producción y poder desplegarla usando el alojamiento de sitios web que Firebase proporciona. Por el momento no ha sido posible debido a posibles problemas de seguridad con el manejo de llaves de API que se encuentran registradas en el apartado del front-end.

5. BIBLIOGRAFÍA

[1] Silvie Turner, «Breve Historia del Papel», en *A Short History of Papermaking: https://iconio.com/ABCD/B/pdf/papel.pdf*, 1991, pp. 114-116.

[2] Walter Pauk, *How to Study in College*, 7.ª ed. Hougthon Mifflin Company, 2001.

[3] «Procesadores de texto», *https://es.wikipedia.org/wiki/Procesador_de_texto*.

[4] Mari van Wyk y Linda van Ryneveld, «Posibilidades de dispositivos móviles y aplicaciones para tomar notas que respalden la toma de notas cognitivamente exigente», https://link.springer.com/article/10.1007/s10639-017-9684-0.

[5] Faria Huq, Abdus Samee, David Chuan-En Lin, Alice Xiaodi Tang, y Jeffrey P Bigham, «NoTeeline: Supporting Real-Time, Personalized Notetaking with LLM-Enhanced Micronotes», *https://dl.acm.org/doi/10.1145/3708359.3712086*, mar. 2025.

[6] Anam Ahmad Khan, Sadia Nawaz, Joshua Newn, Jason M Lodge, James Bailey, y Eduardo Velloso, «Uso de la toma de notas de voz para promover la comprensión conceptual de los estudiantes», *https://arxiv.org/pdf/2012.02927*, oct. 2020.

[7] «Excalidraw».

[8] Apple Inc., «Freeform», https://apps.apple.com/es/app/freeform/id6443742539.

[9] Notion Labs Inc., «Notion», https://www.notion.so/.

[10] Microsoft, «OneNote», https://www.onenote.com/.

[11] PHP Group, «PHP», https://www.php.net/manual/es/intro-whatis.php.

[12] Microsoft, «Active Server Pages (ASP)», *https://es.wikipedia.org/wiki/Active_Server_Pages*. 2023.

[13] Mozilla Foundation, «JavaScript», https://developer.mozilla.org/es/docs/Web/JavaScript.

[14] Google, «Angular», https://angular.dev/.

[15] Evan You, «Vue», https://vuejs.org/.

[16] Facebook, «React», https://es.react.dev/.

[17] Google, «Firebase», https://firebase.google.com/.

[18] Espen Hovlandsdal, «React Markdown», https://github.com/remarkjs/react-markdown.

[19] James Brill, «React Speech Recognition», https://github.com/JamesBrill/react-speech-recognition/tree/master.

[20] Ionic Team, «Ion Icons», https://github.com/ionic-team/ionicons.

[21] Jin Choi, Mark Marcelo, y Bita Djaghouri, «Reactide», https://github.com/reactide/reactide.

[22] Microsoft, «Visual Studio Code», https://code.visualstudio.com/.

[23] Figma Inc., «Figma», https://www.figma.com/.

[24] Inkscape team, «Inkscape», https://inkscape.app/es/.

[25] Atlassian, «Trello», https://trello.com/.

[26] MP Ediciones, «Ciclo de Vida del Software», en *https://ingsw.pbworks.com/f/Ciclo+de+Vida+del+Software.pdf*.

[27] Thinkonmarketing.es, «Todo sobre las metodologías ágiles y cómo aplicarlas», *https://thinkonmarketing.es/metodologias-agiles/*.

[28] Atlassian, «Kanban», *https://www.atlassian.com/es/agile/kanban*.

[29] Anibal Álvarez, «La importancia del bocetaje», *https://foroalfa.org/articulos/la-importancia-del-bocetaje*.

[30] Itamar Haim, «¿Qué es un wireframe en el diseño web?», *https://elementor.com/blog/es/que-es-un-wireframe-en-el-diseno-webcomo-crear-procesos-herramientas/*, sep. 2024.

[31] Luís Filipe, «La importancia de la creación de prototipos de software», *https://you-x.eu/es/la-importancia-de-la-creacion-de-prototipos-de-software/*.

ANEXO I: CASOS DE USO EXTENDIDOS

En este apartado se describen con todo detalle los diferentes casos de uso que aparecen en el diagrama de casos de uso. Estos son:

1. CU-1. Iniciar sesión.
2. CU-2. Cerra sesión.
3. CU-3. Crear carpeta o grupo.
4. CU-4. Crear nota.
5. CU-5. Ver/Editar nota.
6. CU-6. Guardar nota.
7. CU-7. Dibujar.
8. CU-8. Añadir cuadro de texto.
9. CU-9. Editar cuadro de texto.
10. CU-10. Añadir imagen.
11. CU-11. Transcribir reunión.
12. CU-12. Resumen con IA.
13. CU-13. Mover tablero.
14. CU-14. Mover nodo.
15. CU-15. Hacer zoom en el tablero.

CU-1	**Iniciar sesión**
Actores	Usuario no registrado
Descripción	Accede al menú de iniciar sesión usando una cuenta de Google.
Dependencias	
Precondición	Conocer el usuario y contraseña de la cuenta de Google
Secuencia Normal	1 – El usuario presiona en el botón de iniciar sesión en la página principal 2 – Selecciona un usuario existente

Postcondición	Carga la página home del usuario con sus ficheros
Excepciones	1 - Si el usuario es incorrecto o cancela el inicio de sesión vuelve a la página de inicio

CU-2	**Cerrar sesión**
Actores	Usuario registrado
Descripción	Cierra la sesión con el usuario de Google
Dependencias	CU-1
Precondición	
Secuencia Normal	1 – El usuario presiona en el botón de cerrar sesión en la página home del usuario.
Postcondición	El usuario es redirigido a la página principal donde podrá volver a iniciar sesión
Excepciones	

CU-3	**Crear carpeta o grupo**
Actores	Usuario registrado
Descripción	Crea una nueva carpeta para que el usuario pueda organizar sus notas
Dependencias	CU-1
Precondición	Conocer el nombre de la carpeta o grupo

Secuencia Normal	1 – El usuario presiona en el botón de crear una nueva carpeta.
Postcondición	Aparece una nueva carpeta o grupo
Excepciones	

CU-4	**Crear nota**
Actores	Usuario registrado
Descripción	Crea una nueva nota dentro de una carpeta o grupo
Dependencias	CU-1
Precondición	Conocer el nombre de la nota
Secuencia Normal	1 – El usuario presiona en el botón de crear una nueva nota.
Postcondición	El usuario es redirigido a la página de edición de notas
Excepciones	

CU-5	**Ver/Editar nota**
Actores	Usuario registrado, usuario no registrado
Descripción	Carga el contenido de una nota en la página de edición de notas
Dependencias	CU-1
Precondición	Conocer la nota que se quiere editar
Secuencia Normal	1 – El usuario pulsa encima de una nota

Postcondición	El usuario es redirigido a la página de edición de notas
Excepciones	1 – Si el usuario no está registrado puede acceder a una nota que no se guarda desde la página de inicio

CU-6	**Guardar nota**
Actores	Usuario registrado
Descripción	Guarda la nota editada en la base de datos de Firebase
Dependencias	CU-1, CU-5
Precondición	
Secuencia Normal	1 – El usuario pulsa en el botón de guarda la nota
Postcondición	El sistema confirma que la nota se ha guardado con éxito
Excepciones	

CU-7	**Dibujar**
Actores	Usuario registrado, usuario no registrado
Descripción	Habilita el modo de dibujo dentro de la página de edición de notas
Dependencias	CU-1, CU-4, CU-5
Precondición	
Secuencia Normal	1 – El usuario presiona en el botón con el icono de un pincel. 2 – El usuario arrastra el cursor por la pantalla

Postcondición	Aparece una línea en pantalla según la dirección del cursor
Excepciones	

CU-8	**Añadir cuadro de texto**
Actores	Usuario registrado, usuario no registrado
Descripción	Permite añadir un cuadro de texto editable en pantalla
Dependencias	CU-1, CU-4, CU-5
Precondición	
Secuencia Normal	1 – El usuario presiona en el botón con el icono del cuadro de texto 2 – El usuario pulsa en cualquier posición de la pantalla
Postcondición	Aparece el cuadro de texto en la posición seleccionada
Excepciones	

CU-9	**Editar cuadro de texto**
Actores	Usuario registrado, usuario no registrado
Descripción	Cambia al modo de edición en el cuadro de texto
Dependencias	CU-7
Precondición	Conocer el texto a introducir dentro del cuadro de texto
Secuencia Normal	1 – El usuario pulsa dos veces encima del cuadro de texto 2 – El sistema cambia el cuadro de texto al modo de edición

	3 – El usuario introduce con el teclado el texto 4 – El usuario confirma la operación en el botón verde con un tic
Postcondición	El cuadro de texto se actualiza con los datos introducidos
Excepciones	

CU-10	**Añadir imagen**
Actores	Usuario registrado, usuario no registrado
Descripción	Permite añadir una imagen en el tablero a partir de una URL
Dependencias	CU-1, CU-4, CU-5
Precondición	Conocer la dirección URL de la imagen
Secuencia Normal	1 – El usuario presiona en el botón con el icono de una imagen. 2 – El usuario selección una posición en pantalla 3 – El sistema introduce un nuevo cuadro en la posición seleccionada y solicita al usuario la URL de la imagen 4 – El usuario introduce la URL de la imagen
Postcondición	El sistema actualiza el cuadro para introducir la imagen por la imagen elegida
Excepciones	4 – Si la URL de la imagen no es correcta el sistema no actualiza el cuadro con la imagen deseada

CU-11	**Transcribir reunión**
Actores	Usuario registrado, usuario no registrado

Descripción	Graba el audio del micrófono del usuario y realiza una transcripción en tiempo real
Dependencias	CU-1, CU-4, CU-5
Precondición	El navegador soporta el reconocimiento de audio El usuario permite la grabación del micrófono
Secuencia Normal	1 – El usuario presiona en el botón con el icono de un micrófono. 2 – El sistema muestra un nuevo cuadro con un botón para iniciar la grabación y pararla 3 – El usuario pulsa en botón de iniciar la grabación y le habla al micrófono 4 – El sistema muestra el audio transcrito en un cuadro de texto
Postcondición	
Excepciones	4 – Si el usuario pausa la grabación el sistema deja de grabar y transcribir el audio

CU-12	**Resumen con IA**
Actores	Usuario registrado, usuario no registrado
Descripción	Realiza un resumen con inteligencia artificial de la transcripción realizada
Dependencias	CU-10
Precondición	Existe una transcripción previamente
Secuencia Normal	1 – El usuario presiona sobre el botón de realizar un resumen con IA

	2 – El sistema carga en un cuadro de texto el resumen hecho por la IA
Postcondición	El sistema muestra el resumen hecho por la IA
Excepciones	

CU-13	**Mover tablero**
Actores	Usuario registrado, usuario no registrado
Descripción	Permite al usuario deslizar el tablero en cualquier dirección
Dependencias	CU-1, CU-4, CU-5
Precondición	
Secuencia Normal	1 – El usuario presiona el botón derecho del ratón sobre una zona vacía del tablero. 2 – El usuario arrastra el cursor para mover el tablero
Postcondición	El tablero actualiza su posición a la actualizada por el usuario
Excepciones	1 - Si el cursor está encima de otro elemento el tablero no se mueve

CU-14	**Mover nodo**
Actores	Usuario registrado, usuario no registrado
Descripción	Permite mover los elementos de dentro del tablero
Dependencias	CU-1, CU-4, CU-5
Precondición	Conocer el nodo que se va a mover

Secuencia Normal	1 – El usuario presiona el botón izquierdo del ratón sobre el nodo elegido 2 – El usuario arrastra el cursor por la pantalla
Postcondición	El nodo actualiza su posición a la elegida por el usuario
Excepciones	

CU-15	**Hacer zoom en el tablero**
Actores	Usuario registrado, usuario no registrado
Descripción	Aumenta o reduce el zoom sobre la vista del tablero
Dependencias	CU-1, CU-4, CU-5
Precondición	
Secuencia Normal	1 – El usuario arrastra la rueda del ratón. 2 – El sistema acerca o aleja el tablero según la dirección de la rueda del ratón
Postcondición	El sistema actualiza el zoom elegido por el usuario en el tablero
Excepciones	