UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Votalo: desarrollo de una aplicación para gestión de votaciones en eventos

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Madrid Valiente, Sergio

Tutor/a: Letelier Torres, Patricio Orlando

CURSO ACADÉMICO: 2023/2024

Agradecimientos

A todos los amigos que hice en la universidad, por hacer de estos cuatro años los mejores de mi vida y por aportarme tantos recuerdos inolvidables. En especial, a todos aquellos que han pasado tantas horas conmigo en la universidad, no podría nombrarlos a todos, pero ellos saben quiénes son.

A Patricio Letelier, por permitirme realizar un TFG de emprendimiento con él y ofrecerme tantas facilidades y sugerencias para que el desarrollo de la aplicación fuera un éxito.

A Daniel Chinesta Nuñez, por ser el mejor profesor posible y explicarnos el temario durante tantas tardes en Discord.

A los participantes de los experimentos, por todo el feedback aportado y por prestarnos parte de su tiempo.

Y sobre todo a mi familia, por todo su apoyo incondicional durante la carrera y por aportarme los valores que me han llevado a ser quien soy hoy en día.

Resum

En els esdeveniments, la votació de projectes juga un paper crucial per a determinar els guanyadors i brindar retroalimentació constructiva als participants. No obstant això, moltes d'aquestes votacions encara es realitzen mitjançant mètodes tradicionals com l'ús de paper, fulls de càlcul o discussions grupals, que manquen d'eficiència i poden portar a resultats imprecisos i massa subjectius. A més, la falta d'aplicacions especialitzades que donen suport a aquests processos de votació ha creat una necessitat significativa en el mercat.

Votalo és una aplicació web que aborda aquesta necessitat simplificant i optimitzant el procés de votació en esdeveniments com hackatons, classes i votacions laborals. Votalo es va dissenyar per a cobrir un nínxol no explotat en el mercat: les votacions per aspectes específics a avaluar, personalitzats segons les necessitats de l'organitzador de l'esdeveniment.

Aquest TFG s'ha desenvolupat com un projecte d'emprenedoria amb el suport de start.inf, l'espai d'emprenedoria de la ETSINF.

Votalo té com a objectiu principal facilitar els processos de votació, permetent una avaluació detallada i estructurada per part dels jutges. A més, ofereix eines per a fomentar la retroalimentació saludable entre jutges i participants, millorant així la qualitat del feedback rebut i promovent el desenvolupament dels participants. Amb Votalo, es busca no sols millorar l'eficiència i precisió en les votacions, sinó també assegurar una major transparència i justícia en el procés, eliminant la dependència de mètodes rudimentaris i propensos a errors.

Paraules clau: emprenedoria, desenvolupament web, votacions

Resumen

En los eventos, la votación de proyectos juega un papel crucial para determinar los ganadores y brindar retroalimentación constructiva a los participantes. Sin embargo, muchas de estas votaciones aún se realizan mediante métodos tradicionales como el uso de papel, hojas de cálculo o discusiones grupales, que carecen de eficiencia y pueden llevar a resultados imprecisos y demasiado subjetivos. Además, la falta de aplicaciones especializadas que apoyen estos procesos de votación ha creado una necesidad significativa en el mercado.

Votalo es una aplicacion web que aborda esta necesidad simplificando y optimizando el proceso de votación en eventos como hackatones, clases y votaciones laborales. Votalo se diseñó para cubrir un nicho no explotado en el mercado: las votaciones por aspectos específicos a evaluar, personalizados según las necesidades del organizador del evento.

Este TFG se ha desarrollado como un proyecto de emprendimiento con el apoyo de start.inf, el espacio de emprendimiento de la ETSINF.

Votalo tiene como objetivo principal facilitar los procesos de votación, permitiendo una evaluación detallada y estructurada por parte de los jueces. Además, ofrece herramientas para fomentar la retroalimentación saludable entre jueces y participantes, mejorando así la calidad del feedback recibido y promoviendo el desarrollo de los participantes. Con Votalo, se busca no solo mejorar la eficiencia y precisión en las votaciones, sino también asegurar una mayor transparencia y justicia en el proceso, eliminando la dependencia de métodos rudimentarios y propensos a errores.

Palabras clave: emprendimiento, desarrollo web, votaciones

Abstract

At events, project voting plays a crucial role in determining winners and providing constructive feedback to participants. However, many of these votes are still conducted using traditional methods such as paper, spreadsheets, or group discussions, which lack efficiency and can lead to inaccurate and overly subjective results. Moreover, the lack of specialized applications supporting these voting processes has created a significant need in the market.

Votalo is a web application that addresses this need by simplifying and optimizing the voting process at events such as hackathons, classes, and workplace votes. Votalo is designed to cover an untapped niche in the market: voting based on specific aspects to be evaluated, customized according to the organizer's needs.

This final year project has been developed as an entrepreneurial project with the support of start.inf, the entrepreneurship space at ETSINF.

Votalo's primary goal is to facilitate voting processes, allowing for detailed and structured evaluation by judges. Additionally, it offers tools to encourage healthy feedback between judges and participants, thereby improving the quality of feedback received and promoting participants' development. With Votalo, the aim is not only to enhance efficiency and accuracy in voting but also to ensure greater transparency and fairness in the process, eliminating reliance on rudimentary and error-prone methods.

Key words: entrepreneurship, web development, voting

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

1.1 Motivación

Las tecnologías en la actualidad nos ayudan en una gran parte de nuestras tareas diarias. Además de otorgarnos nuevos aportes, que antes de estas eran impensables, también nos han ayudado a automatizar muchas otras tareas que, de otro modo, serían tediosas y consumirían mucho tiempo

Una de estas tareas cotidianas es la toma de decisiones colectiva o la votación entre varias opciones. Esto puede variar desde elegir el mejor destino de viaje hasta decidir qué software ha destacado más en un concurso o quién es merecedor de un premio en un grupo. Si bien existen herramientas para facilitar votaciones simples, a menudo estas herramientas tratan la elección de manera genérica y no consideran aspectos específicos críticos.

Un ejemplo notable son los hackatones[1], donde diferentes proyectos de software se valoran en un tiempo limitado. En estos eventos, es común que los jueces utilicen métodos tradicionales como el voto en papel, discusiones grupales o hojas de cálculo para la votación. Estos métodos no solo carecen de eficiencia, sino que también fallan en profundizar en las características distintivas de cada proyecto, lo que a menudo lleva a pasar por alto información valiosa.

Otra preocupación significativa es la ausencia de retroalimentación detallada para los participantes sobre sus proyectos, ya que no tienen forma de saber las áreas de mejora o los puntos fuertes de sus proyectos, perdiendo así valiosas oportunidades de aprendizaje.

Ante todas estas problemáticas, surgió la idea de desarrollar Votalo, una aplicación de votaciones. Esta permitirá a los organizadores crear diferentes eventos personalizados, eligiendo criterios específicos de valoración y determinando su ponderación en la nota final. Además, ofrecerá a los jueces la posibilidad de dejar comentarios sobre cada proyecto, proporcionando así retroalimentación constructiva y transparente a los participantes.

Votalo no solo garantizará una mayor transparencia y justicia en el proceso de votación, sino que también optimizará la labor de los jueces. Eliminará la necesidad de métodos obsoletos y propensos a errores, asegurando una evaluación más precisa y objetiva. Con ello, se espera mejorar significativamente tanto la experiencia de los jueces como la de los participantes, fomentando un ambiente de competencia sano y constructivo.

[1] Un hackatón es un evento, generalmente de uno o dos días de duración, en el que varios equipos se reúnen para colaborar intensivamente en proyectos, a menudo con un enfoque en la innovación y la solución de problemas específicos.

Tras hablar con diversos jueces y participantes en este tipo de eventos, quedó clara la necesidad de optimizar el sistema de votaciones, ya que, hoy en día, seguimos utilizando métodos poco eficientes.

Votalo tiene como meta a largo plazo el establecimiento de un nuevo estándar en sistemas de votación para eventos, donde la eficiencia y la transparencia sean las prioridades, contribuyendo significativamente al campo de la tecnología de eventos.

1.2 Objectivos

El principal objetivo de este trabajo es desarrollar Votalo, una aplicación innovadora diseñada para revolucionar el proceso de votación en diversos eventos, que permita acelerar el proceso de votación, minimizar la intervención humana para reducir errores, y proporcionar retroalimentación valiosa que fomente el desarrollo de los participantes.

Los objetivos específicos de este TFG son:

• Diseñar e implementar una aplicación que automatice los sistemas de votaciones y permita gran versatilidad.

• Implementar sistemas que permitan la actualización de los datos en tiempo real.

• Realizar un correcto tratamiento de las credenciales de los participantes y un buen uso de las cookies, asegurando la protección de los datos.

Este TFG ha seguido las pautas asociadas a un TFG de emprendimiento proporcionadas por Start.inf, asegurando que el desarrollo del proyecto no solo cumpla con los requisitos académicos sino que también esté alineado con las mejores prácticas.

1.3 Estructura de la memória

La estructura de esta memoria esta organizada en distintos capítulos:

Comenzamos con una introducción que, como ya hemos visto, abarca la motivación detrás de la elección de este proyecto, y los objetivos que pretende alcanzar este trabajo.

En el capítulo 2, se realiza una evaluación de la idea de negocio, en la cual se detallan nuestros clientes potenciales, se realiza un estudio de mercado y un análisis de competidores, y se destaca lo que nuestra aplicación aporta en comparación con soluciones existentes. También se incluye un análisis financiero preliminar, un análisis DAFO [1], un modelo de negocio Lean Canvas [2], y finalmente, las conclusiones de la evaluación.

En el capítulo 3 se exponen las tecnologías utilizadas para el desarrollo del proyecto. El siguiente capitulo de la memoria es el desarrollo de la solución, donde se muestra la metodología utilizada y las herramientas de apoyo que se han elegido, veremos los requisitos de nuestra aplicación y como se ha realizado el diseño de esta. También se encontrará información sobre las pruebas realizadas, una descripción de la aplicación y una cronología del desarrollo del proyecto.

En el capítulo 4 se realiza el desarrollo de la solución, mostrando las metodologías utilizadas, anécdotas de la programación, junto con las pruebas realizadas.

En el capítulo 5 se realiza la descripción de la aplicación y los diferentes roles que los usuarios pueden tener dentro de ella.

En el capítulo 6 se muestra la cronología de la aplicación, enseñando el primer y el segundo experimento.

En el capítulo 7, se mostrarán las conclusiones del proyecto, junto con la formación y experiencia que este me ha aportado.

Para finalizar, se expondrán los posibles trabajos futuros que se pueden realizar sobre este proyecto para mejorarlo, así como las referencias necesarias.

CAPÍTULO 2

Evaluación de la idea de negocio

En este capítulo se presenta la evaluación de la idea de negocio para comprobar la viabilidad del proyecto. Para esto, se estudiaron los clientes, se realizó un estudio de mercado para analizar la competencia, se hizo una proyección de ingresos y gastos, un análisis DAFO del proyecto, el Lean Canvas y, finalmente, se sacaron las conclusiones para determinar la viabilidad del proyecto.

2.1 Clientes

En el modelo de negocio de nuestra aplicación de votaciones, quienes realizarán los pagos para acceder a sus funcionalidades serán principalmente los organizadores de los eventos.

Estos organizadores pueden variar desde individuos que gestionan eventos pequeños hasta entidades más grandes como universidades o empresas.

Además de los organizadores, otro tipo de usuarios clave que interactuará con nuestra aplicación es el administrador del evento, que se encargará de configurar los eventos de votación en la plataforma. Sus tareas incluyen la adición de criterios de votación, la ponderación de estos criterios y la configuración general de cada votación.

La aplicación también será utilizada por los jueces, los cuales se encargarán de realizar las votaciones y añadir comentarios, y finalmente por los participantes, para consultar las votaciones y comentarios de los jueces.

Para evaluar la viabilidad comercial del proyecto Votalo, es esencial realizar un análisis de TAM, SAM y SOM [3] [4], que nos permita entender el tamaño del mercado y la porción del mercado que podemos capturar.

TAM (Total Addressable Market)

El TAM representa el mercado total disponible para nuestra aplicación de votaciones. Según varios estudios [5][6][7], el mercado global de la industria de eventos alcanzó un valor de aproximadamente 1.4 billones de dólares en 2023 y se espera que crezca a una tasa compuesta anual (CAGR) del 5.1 % entre 2024 y 2032, alcanzando alrededor de 2.2 billones de dólares para 2032.

SAM (Serviceable Available Market)

El SAM se refiere a la porción del TAM que nuestra aplicación puede atender. Dado que Votalo está diseñado específicamente para eventos que requieren una evaluación detallada por aspectos, podemos centrarnos en eventos tecnológicos, académicos y corporativos que involucren procesos de votación estructurados. Esto reduce el mercado objetivo a una fracción más manejable del TAM. Si consideramos que un 10 % del mercado global de eventos y conferencias se ajusta a estos criterios específicos, nuestro SAM estaría valorado en aproximadamente 140 mil millones de dólares.

SOM (Serviceable Obtainable Market)

El SOM es la porción del SAM que nuestra aplicación puede capturar, dado nuestro alcance y recursos actuales. Inicialmente, nos enfocaremos en eventos tecnológicos y académicos en regiones específicas, como Europa y América del Norte, donde ya tenemos ciertos contactos y podemos llevar a cabo estrategias de marketing más efectivas. Estimamos que podríamos capturar alrededor del 1 % del SAM en los primeros años de operación, lo que se traduciría en un mercado potencial de alrededor de 1.4 mil millones de dólares.

Con este análisis, podemos ver que Votalo tiene un potencial significativo para capturar una porción valiosa del mercado global de eventos y conferencias, especialmente aquellos que requieren procesos de votación detallados y estructurados.

2.2 Estudio de mercado y competidores

En este apartado se estudiarán los productos competidores, se verá una tabla comparativa y se expondrán las innovaciones que ofrece nuestro proyecto sobre los ya existentes.

2.2.1. Aplicaciones similares

A la hora de realizar un proyecto, es muy importante ver los productos de la competencia y ver qué es lo que ofrecen, para encontrar la forma de diferenciar tu proyecto de los demás.

A continuación se verán las principales aplicaciones que serían competidores de nuestra aplicación.

Google Forms

En primer lugar, hablaremos de Google Forms[1], que es una aplicación mundialmente conocida que nos permite realizar encuestas simples de manera rápida, sin la necesidad de estar registrados.

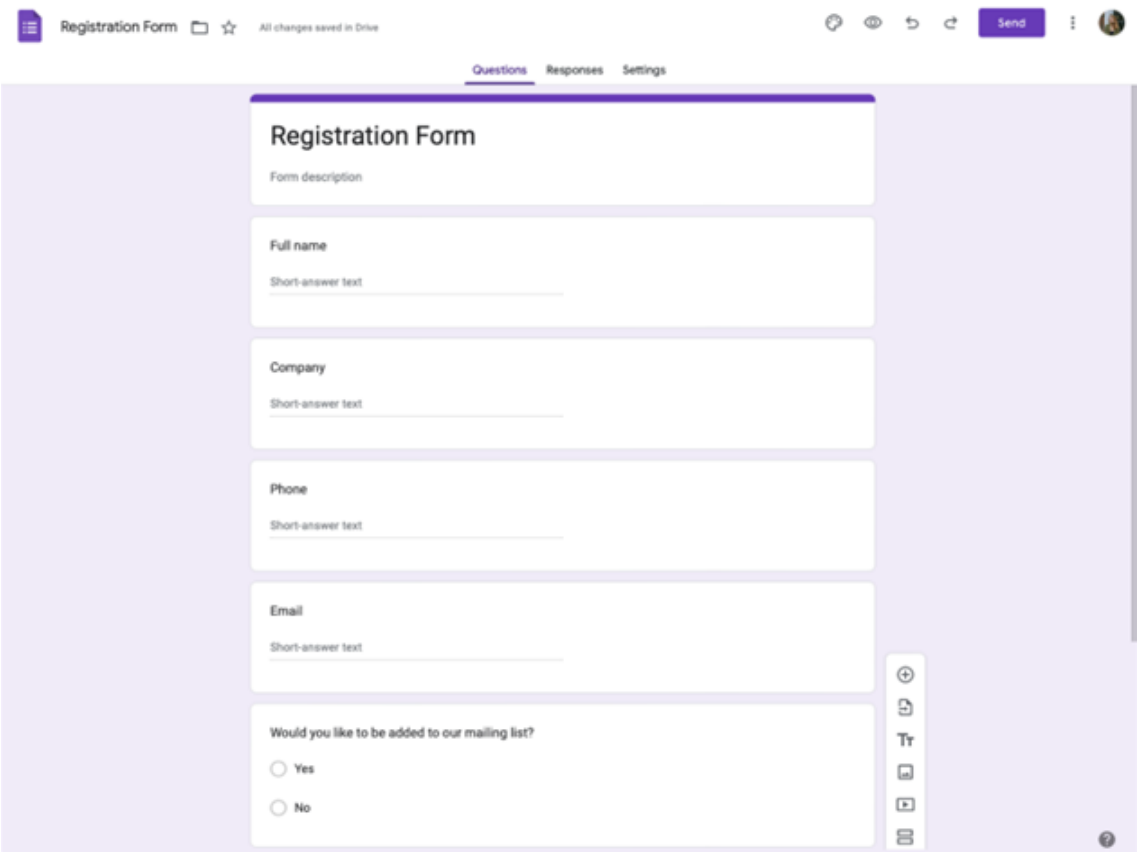

Figura 2.1: Interfaz de Google Forms

En la figura 2.1 se puede observar la interfaz de Google Forms.

Esta aplicación te ofrece la posibilidad de crear encuestas básicas, mostrando los resultados en diferentes tipos de gráficas. La principal diferencia de nuestra aplicación con Google Forms es la que encontramos con prácticamente todas, y es que Google Forms está principalmente orientada a encuestas sencillas, donde el objetivo es elegir entre varias opciones, sin una profundidad significativa en la evaluación de múltiples criterios. En cambio, nuestra aplicación está diseñada específicamente para facilitar encuestas y votaciones con un nivel de detalle mucho mayor.

A diferencia de Google Forms, Votalo permite a los usuarios votar y evaluar varios aspectos de cada ítem, cada uno con su propia ponderación. Esta funcionalidad es especialmente valiosa en contextos donde se requiere una evaluación exhaustiva, como en la valoración de proyectos o trabajos en hackatones, competiciones de diseño o eventos similares. Nuestra plataforma no solo busca determinar la opción más popular, sino identificar al ganador en un contexto específico basado en una evaluación multifacética.

Otra ventaja significativa de nuestra aplicación es su adaptabilidad a diferentes escenarios y requisitos de votación. Mientras que Google Forms ofrece una plataforma más genérica y adecuada para gran variedad de contextos, nuestra aplicación se destaca en su

[1] https://www.google.com/forms/about/

capacidad para ser personalizada específicamente para eventos que requieren una evaluación más detallada y estructurada.

Pollie

Pollie[2], al igual que Google Forms, es una solución eficaz para la realización de encuestas generales de manera rápida y sencilla. Ofrece características interesantes como la generación de gráficos y estadísticas sobre los resultados de las votaciones, y la posibilidad de exportar estos datos a formato .csv.

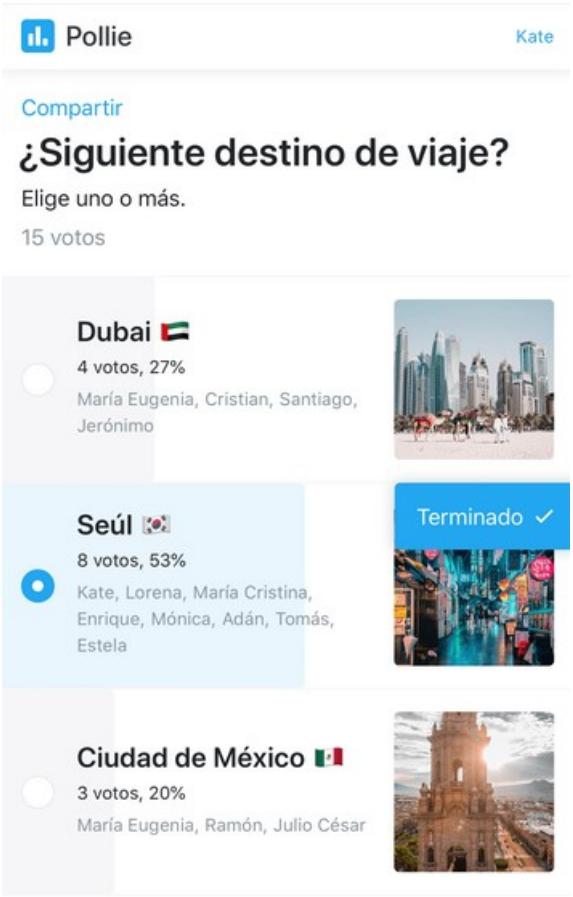

Figura 2.2: Interfaz de Pollie

En la figura 2.2 se puede observar la interfaz de Pollie.

Aunque Pollie ofrece un nivel de personalización mayor en comparación con algunas otras alternativas, su objetivo es más las encuestas rápidas y decisiones generales. Esto lo hace atractivo para usuarios que buscan una herramienta eficiente y fácil de usar para recopilar feedback o tomar decisiones rápidas, pero no necesariamente para aquellos que requieren un análisis más profundo.

Al igual que ocurría con Google Forms, este proyecto no nos da la posibilidad de generar votaciones detalladas y multifacéticas, sino que se centra en hacer encuestas más generales, por lo que, aun siendo una aplicación de encuestas, abarca un nicho diferente al de nuestra aplicación.

[2] https://www.pollie.app/

Evalato

Finalmente analizaremos Evalato[3], este se posiciona como un competidor más cercano a nuestro proyecto en términos de funcionalidad, pero con diferencias notables en el enfoque y la audiencia objetivo.

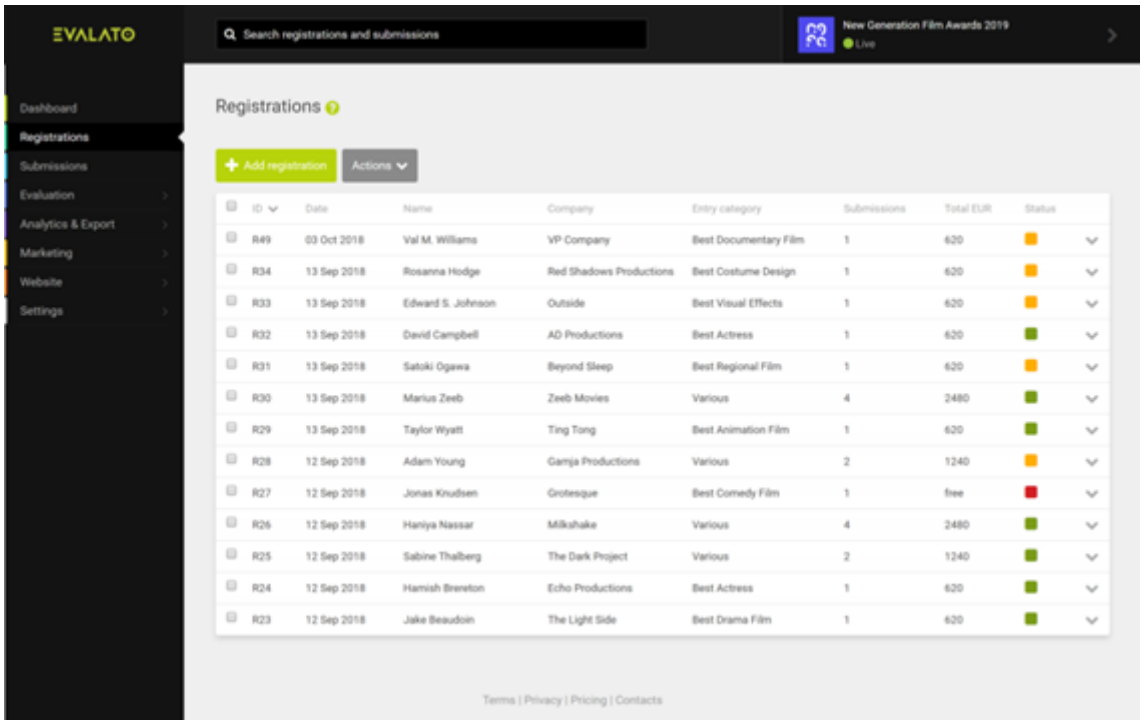

Figura 2.3: Dashboard de Evalato

En la figura 2.3 se puede observar la interfaz de Evalato.

Evalato nos permite generar votaciones por rondas con gran nivel de detalle, pero está más enfocado al uso empresarial por la gran variedad de posibilidades que ofrece, las cuales hacen que no sea tan sencilla de utilizar. Además, para poder utilizarla debes pagar anualmente, siendo la opción más básica la de 900€. Debido a esto, la aplicación no ha podido ser probada a fondo para realizar un análisis más exhaustivo.

Si bien es cierto que lo que nuestra aplicación hace también parece ser posible en Evalato, los propósitos de ambas son muy distintos, ya que como hemos mencionado antes, Evalato tiene propósitos más empresariales, con una mayor complejidad de uso, y enfocada en repartir premios o becas. En cambio, nuestro proyecto tiene como objetivo ser una herramienta accesible y fácil de utilizar que permita facilitar el trabajo de los jueces en concursos, y sobre todo, facilitando la aportación de feedback a los participantes.

[3] https://www.google.com/forms/about/

2.2.2. Tabla comparativa

Como en todo estudio de mercado, se ha realizado una tabla comparativa con todas las anteriormente mencionadas.

Nombre	**Google Forms**	**Pollie**	**Evalato**	**Votalo**
Orientación de uso	Votaciones simples generales	Votaciones simples generales	Votaciones muy detalladas para entregar premios	Votaciones detalladas para concursos
Versión web	SI	NO	SI	SI
Versión móvil	NO	SI	NO	SI
Multilenguaje	SI	NO	NO	SI
Historial de encuestas anteriores	SI	NO	SI	SI
Facilidad para añadir feedback	SI	NO	NO	SI
Diferentes Criterios de votación	NO	NO	SI	SI
Grado de Personalización	BAJO	MEDIO	ALTO	Medio
Precio Anual	Gratuita	Gratuita	Desde 900€	80€

Tabla 2.1: Comparación de diferentes plataformas de votación

Como se observa en la Tabla 2.1, hay muchas aplicaciones enfocadas de manera general a las votaciones, pero no hemos podido encontrar una aplicación accesible enfocada votaciones en concursos. Es aquí donde nuestra aplicación introduce un valor distintivo y significativo.

Nuestra plataforma se destaca por permitir a los organizadores configurar votaciones con un gran nivel de detalle. Facilita la definición de criterios específicos para evaluar proyectos, permitiendo a los organizadores establecer los atributos específicos que serán valorados y la importancia relativa de cada uno de estos en la puntuación final. Esta capacidad para personalizar detalladamente los criterios de evaluación y ponderación es una funcionalidad que no hemos encontrado en otras aplicaciones del mercado.

Otro aspecto clave es la posibilidad que ofrece a los jueces de aportar comentarios constructivos sobre cada proyecto. Esto promueve la transparencia en el proceso de votación y contribuye significativamente al desarrollo y mejora de los participantes.

Con nuestro modelo de aplicación, se ahorraría mucho tiempo en este tipo de procesos, que actualmente se suelen realizar con métodos poco eficientes como escribiendo a mano o con hojas de Excel. También evitaríamos en gran medida fallos en los datos de las votaciones que puedan ocurrir debido a fallos humanos, y finalmente, garantiza resultados en las votaciones transparentes y confiables.

2.3 Modelo de negocio y proyección económica

El desarrollo de un proyecto incluye un análisis detallado del modelo de negocio y la proyección económica. Este análisis permite estimar los costos y los ingresos a lo largo de los próximos tres años, evaluando la viabilidad del proyecto y previendo cuándo comenzará a generar ingresos reales.

Para Votalo, los ingresos provienen de suscripciones trimestrales que los organizadores de eventos deben pagar tras finalizar el período de prueba gratuita. Cada suscripción trimestral tiene un costo de 20€, lo que es accesible y competitivo, asegurando que nuestra aplicación sea una opción atractiva para una amplia gama de usuarios.

2.3.1. Proyección de Ingresos

En la Tabla 2.2, se presenta una estimación de ingresos donde, en el primer año, se anticipa que Votalo tendrá suscripciones trimestrales crecientes según la siguiente proyección:

Trimestre	1	2	3	4	5	6	7	8	9	10	11	12
Suscripciones	5	40	70	100	150	250	400	600	900	1300	1800	2500
Ingresos (€)	100	800	1400	2000	3000	5000	8000	12000	18000	26000	36000	50000

Tabla 2.2: Proyección de suscripciones e ingresos trimestrales

El valor de suscripción de Votalo será 80€ anuales, es considerablemente más accesible en comparación con Evalato, que puede llegar a 900€ o más anualmente. Google Forms ofrece funcionalidades básicas de forma gratuita, pero para características avanzadas el costo puede llegar a 200€ anualmente por usuario en Google Workspace Business. Pollie, con su modelo freemium, puede atraer a usuarios con su versión gratuita pero potencialmente cobrar por funciones avanzadas.

Comparando estos precios, Votalo se posiciona competitivamente en el mercado, ofreciendo una suscripción asequible con funcionalidades específicas para votaciones detalladas y estructuradas que no se encuentran en Google Forms o Pollie y a un precio significativamente menor que Evalato.

La proyección de Votalo anticipa alcanzar 215 suscripciones en el primer año. Considerando que Google Forms y Pollie atraen a un amplio público debido a su gratuidad y facilidad de uso, y Evalato a un nicho específico de organizadores de eventos dispuestos a pagar más por funcionalidades avanzadas, la proyección de Votalo parece razonable si se considera una estrategia de marketing efectiva y la capacidad de demostrar su valor único en eventos que requieren votaciones detalladas.

Con estas cifras, los ingresos anuales proyectados son: - Primer año: 4.300€ - Segundo año: 35.000€ - Tercer año: 137.000€

Este análisis muestra que Votalo tiene un modelo de suscripción competitivo y una proyección de suscriptores razonable en comparación con sus competidores. La clave del éxito radica en la capacidad de Votalo para atraer y retener a los usuarios que buscan una solución específica para votaciones detalladas y estructuradas.

2.3.2. Proyección de Gastos

Los gastos estimados incluyen la compra de un ordenador, campañas de marketing recurrentes, servicios de hosting mensual y el salario de un desarrollador. También se proyecta un crecimiento en los gastos a lo largo de los tres años para reflejar el aumento de actividades y la expansión del equipo.

Gastos	Cantidad	Precio trimestral (€)	1º Trimestre (€)	2º Trimestre (€)	3º Trimestre (€)	4º Trimestre (€)
Ordenador	1	1000	1000	0	0	0
Marketing	-	1500	1500	2000	2500	3000
Desarrollador	1	5100	5100	5100	5100	5100
Hosting	-	60	60	70	80	90
Administración	-	1000	1000	1000	1000	1000
Soporte	-	1500	1500	1600	1700	1800
TOTAL	-	-	11660	10160	10480	10990

Tabla 2.3: Proyección de gastos trimestrales del primer año

Gastos	Cantidad	Precio trimestral (€)	5º Trimestre (€)	6º Trimestre (€)	7º Trimestre (€)	8º Trimestre (€)
Ordenador	1	1000	0	0	0	0
Marketing	-	3500	3500	4000	4500	5000
Desarrollador	1	5250	5250	5250	5250	5250
Hosting	-	100	100	110	120	130
Administración	-	1200	1200	1300	1400	1500
Soporte	-	2000	2000	2100	2200	2300
TOTAL	-	-	12050	12910	14120	15480

Tabla 2.4: Proyección de gastos trimestrales del segundo año

Gastos	Cantidad	Precio trimestral (€)	9º Trimestre (€)	10º Trimestre (€)	11º Trimestre (€)	12º Trimestre (€)
Ordenador	1	1000	0	0	0	0
Marketing	-	6000	6000	6500	7000	7500
Desarrollador	1	5400	5400	5400	5400	5400
Hosting	-	150	150	160	170	180
Administración	-	1600	1600	1700	1800	1900
Soporte	-	2500	2500	2600	2700	2800
TOTAL	-	-	15650	16860	18570	20400

Tabla 2.5: Proyección de gastos trimestrales del tercer año

Para el primer año, los gastos totales se estiman en 43.290€. Los gastos aumentan significativamente en los años siguientes debido a las actividades de marketing y los costos de hosting. Se espera que estos gastos aumenten para reflejar la expansión del negocio y las mayores inversiones en marketing y soporte.

2.3.3. Resumen Trimestral de Ingresos y Gastos

La Tabla 2.6 presenta un resumen trimestral de los ingresos y gastos de Votalo durante los primeros 12 trimestres. Esta tabla muestra cómo los ingresos aumentan gradualmente a medida que más usuarios se suscriben a la aplicación, mientras que los gastos se incrementan progresivamente para apoyar el crecimiento del negocio. Esta proyección es esencial para evaluar el progreso hacia un equilibrio financiero.

Trimestre	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Ingresos (€)	100	800	1400	2000	3000	5000	8000	12000	18000	26000	36000	50000
Gastos (€)	11660	10160	10480	10990	12050	12910	14120	15480	15650	16860	18570	20400

Tabla 2.6: Proyección trimestral de ingresos y gastos de los primeros tres años

En la Figura 2.4, se ilustra la proyección trimestral de ingresos y gastos, destacando la evolución de ambos a lo largo del tiempo. Se observa que los ingresos comienzan a incrementarse significativamente a partir del noveno trimestre, mientras que los gastos aumentan progresivamente, lo que es un indicativo positivo hacia la sostenibilidad financiera del proyecto.

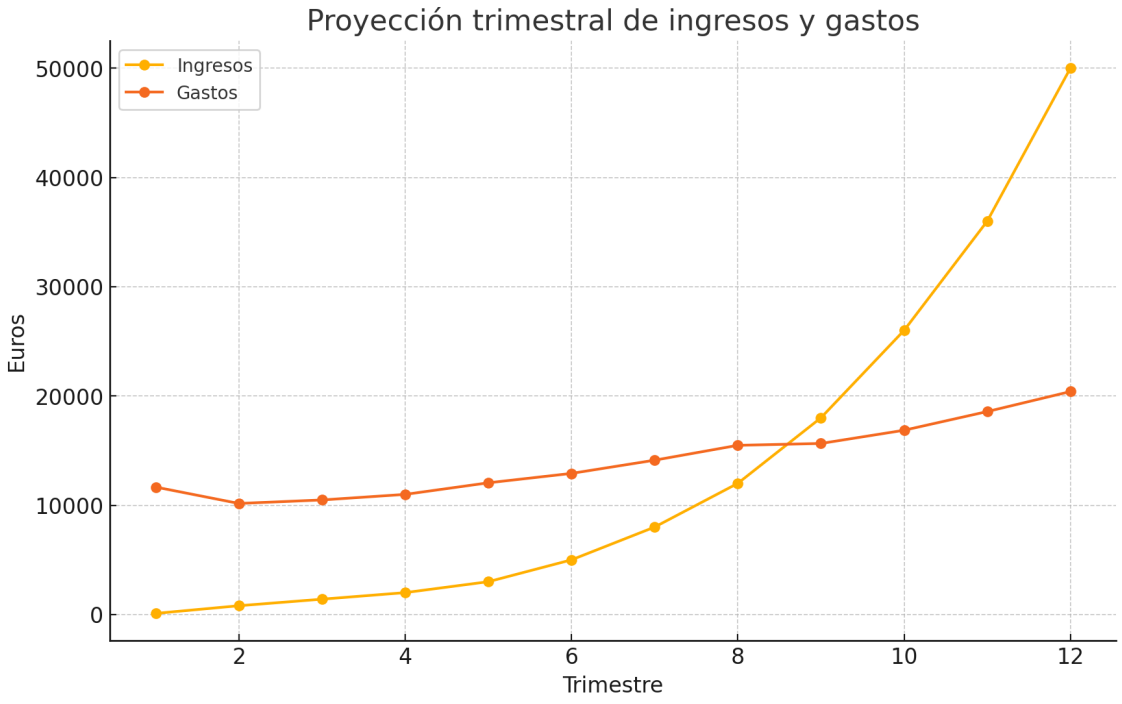

Figura 2.4: Proyección trimestral de ingresos y gastos

2.3.4. Gráfico de Beneficio Trimestral

La Figura 2.5 muestra el gráfico de beneficios trimestrales, destacando que, aunque los primeros trimestres presentan pérdidas, se espera que a partir del noveno trimestre la tendencia se revierta, generando beneficios sostenidos. Este gráfico es crucial para visualizar el punto de inflexión donde los ingresos comienzan a superar a los gastos, indicando el inicio de la rentabilidad del proyecto.

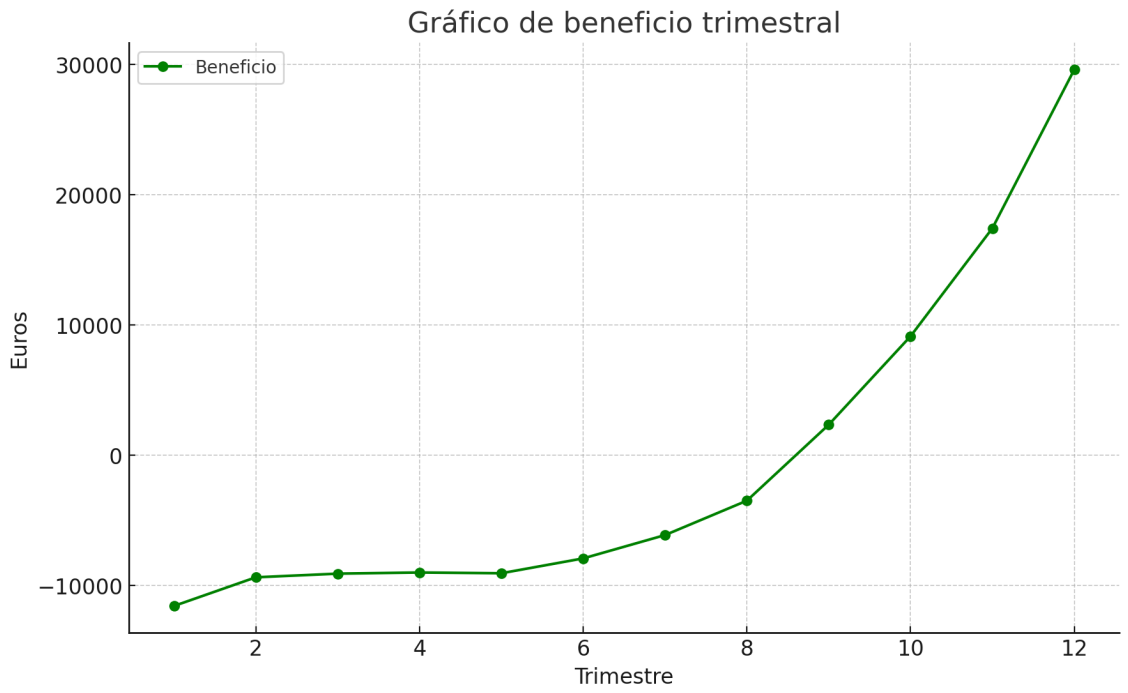

Figura 2.5: Gráfico de beneficio trimestral

2.3.5. Gráfico de Beneficio Acumulado Trimestral

En la Figura 2.6, se ilustra el beneficio acumulado de Votalo durante los primeros 12 trimestres del proyecto. Este gráfico es esencial para entender la evolución del flujo de efectivo a lo largo del tiempo y en qué momento el proyecto comienza a ser financieramente viable.

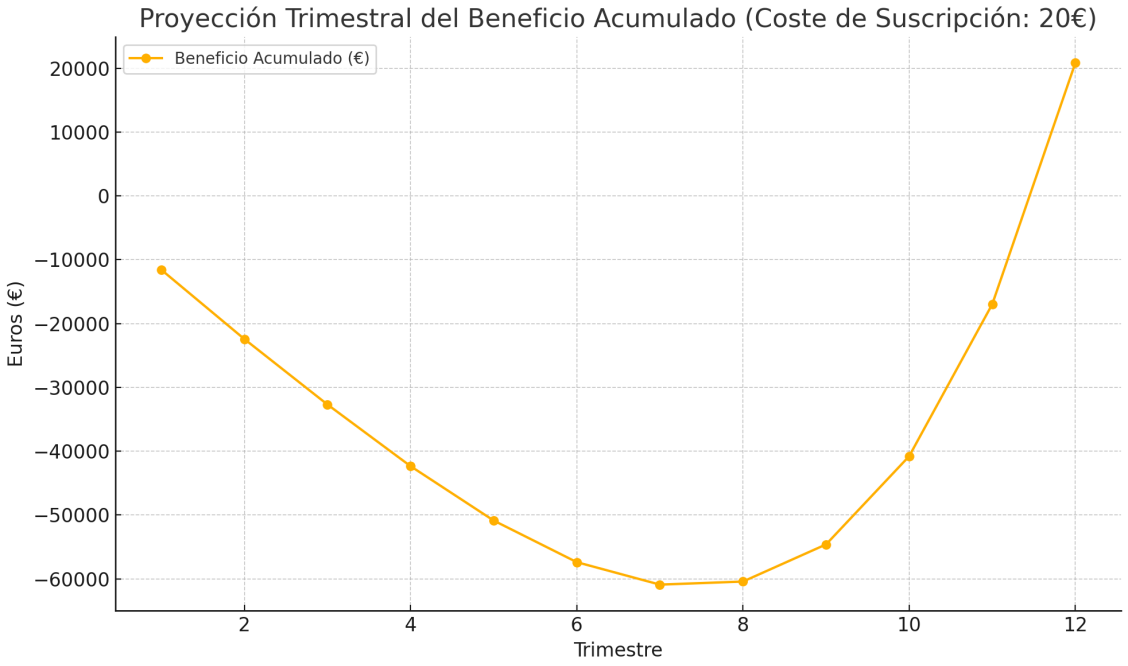

Figura 2.6: Gráfico de beneficio acumulado trimestral

Al principio del gráfico, se observa una tendencia negativa en los primeros trimestres, lo cual es esperado debido a los altos gastos iniciales, incluyendo la compra de equipos y el desarrollo inicial, que superan a los ingresos generados por las suscripciones. En el primer trimestre, los gastos incluyen el costo del ordenador y otros gastos operativos, resultando en una salida significativa de efectivo.

A partir del noveno trimestre, comenzamos a ver un cambio positivo en el beneficio acumulado. Este cambio se debe al incremento sostenido en el número de suscripciones, lo que aumenta los ingresos trimestrales significativamente.

El beneficio acumulado sigue una tendencia ascendente más pronunciada después del noveno trimestre, indicando que los ingresos comienzan a superar consistentemente a los gastos. Esto demuestra que, a partir del noveno trimestre, el proyecto Votalo empieza a generar ingresos reales y se vuelve financieramente viable.

2.4 Análisis DAFO

El análisis DAFO es una herramienta que nos permite analizar las características internas y la situación externa de una empresa en una matriz. Proviene de las siglas en inglés SWOT (Strengths, Weaknesses, Opportunities y Threats).

Esta muestra las características del proyecto como son las fortalezas y las debilidades y también su situación externa como las oportunidades o las amenazas a las que se enfrenta el proyecto.

Figura 2.7: Análisis DAFO

En la figura 2.7 se puede ver el análisis DAFO de nuestra aplicación.

La buena relación con organizadores de eventos es nuestra mayor fortaleza, ya que podríamos proponer el uso de nuestra aplicación en algunos de sus eventos. Por otro lado, la falta de experiencia en el ámbito del emprendimiento es nuestra mayor debilidad.

En cuanto al entorno externo, está claro que la cantidad de eventos en los que se realizan votaciones es muy grande, y con tendencia creciente. Esto en conjunto con la falta de competencia real es nuestra mayor oportunidad de negocio.

En cuanto a las mayores amenazas a las que nos enfrentamos, estaría la aparición de un producto que compita directamente con el nuestro en el cual se puedan invertir más recursos, y el no ser capaces de convencer a los organizadores de que usen nuestro producto.

2.5 Lean Canvas

El Lean Canvas es una herramienta estratégica de negocios que facilita el análisis visual de nuestro modelo de negocio, con el propósito de incrementar nuestras probabilidades de éxito. Nos permite resumir y concretar en qué consiste nuestro negocio y evaluar su viabilidad.

En la Figura 2.8 , se presenta el Lean Canvas de Votalo.

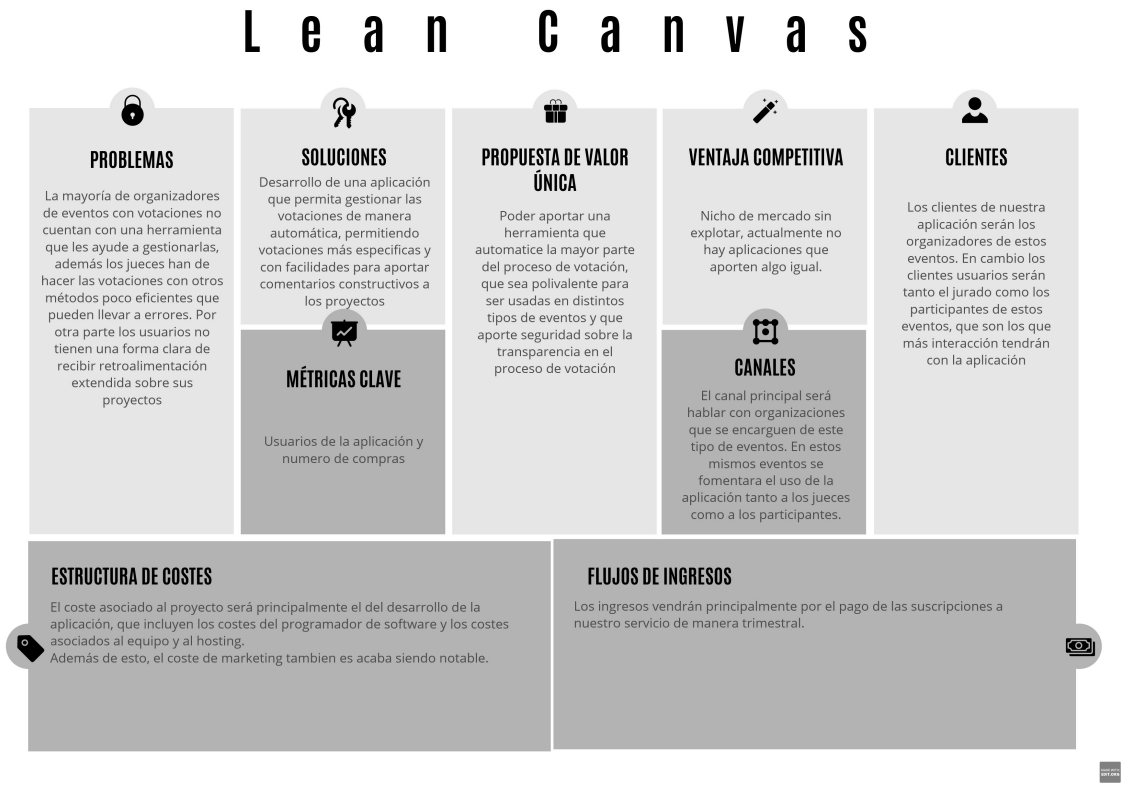

Figura 2.8: Lean Canvas

2.6 Conclusiones de la evaluación

Con toda la información expuesta anteriormente, podemos concluir que el desarrollo de la aplicación Votalo tiene una viabilidad muy alta.

Nuestra mayor oportunidad, sin duda, es la falta de competidores directos. Como hemos visto, esta aplicación ofrece funcionalidades únicas que ninguna otra aplicación ofrece, lo que sugiere que la cantidad de potenciales consumidores es muy alta.

Por otra parte, en el análisis DAFO hemos comprobado que las amenazas y debilidades son poco significativas en comparación con la gran cantidad de fortalezas y oportunidades. Aunque es crucial tener en cuenta estos aspectos negativos, una planificación adecuada puede mitigar muchos de estos problemas.

La proyección económica muestra que la inversión inicial del proyecto sería de unos 11660 euros, una cifra razonable. Además, los datos económicos indican que los beneficios comenzarán a crecer exponencialmente a partir del noveno trimestre, lo que sugiere que el riesgo económico no es muy grande.

En conclusión, Votalo es un proyecto con un gran potencial de crecimiento y liderazgo en su sector, siempre que logre darse a conocer de manera efectiva. Además, no requiere un gran capital inicial. Todos estos datos nos indican que Votalo es un proyecto viable y prometedor

CAPÍTULO 3
Tecnologías y herramientas utilizadas

En este capítulo, se mostrarán y detallarán las tecnologías y herramientas utilizadas durante el desarrollo del proyecto.

Desde el primer momento, se decidió separar el Frontend del Backend, desarrollando dos aplicaciones paralelas. De esta forma, podríamos reducir enormemente la carga en la máquina del usuario, mejorando así la experiencia de uso.

Aunque la idea principal era desarrollar una aplicación para móvil, finalmente se decidió desarrollarla como una aplicación web. Dado el uso esperado de la aplicación, se determinó que sería mucho más fácil de utilizar para los usuarios, ya que al evitar que estos tengan que descargarse una aplicación, se facilita considerablemente la entrada de nuevos usuarios.

3.1 Visual Studio Code

Visual Studio Code[1] es un editor de código fuente desarrollado por Microsoft que se ha convertido en una herramienta esencial para desarrolladores de todo el mundo. Es un editor ligero pero poderoso, con soporte para la depuración, control de versiones integrado, y una vasta colección de extensiones que permiten personalizar y ampliar sus funcionalidades.

Ofrece características avanzadas como la finalización de código, resaltado de sintaxis, refactorización, y navegación de código. Además, su integración con Git permite gestionar repositorios de código de manera eficiente, facilitando el control de versiones.

En el desarrollo de Votalo, se decidió utilizarlo para la programación del Frontend en React por todas las facilidades que ofrece, además de la gran cantidad de extensiones que tiene para mejorar la experiencia de desarrollo con este lenguaje de programación.

En la Figura 3.1 podemos observar cómo es el entorno de desarrollo de Visual Studio Code

[1] https://code.visualstudio.com/

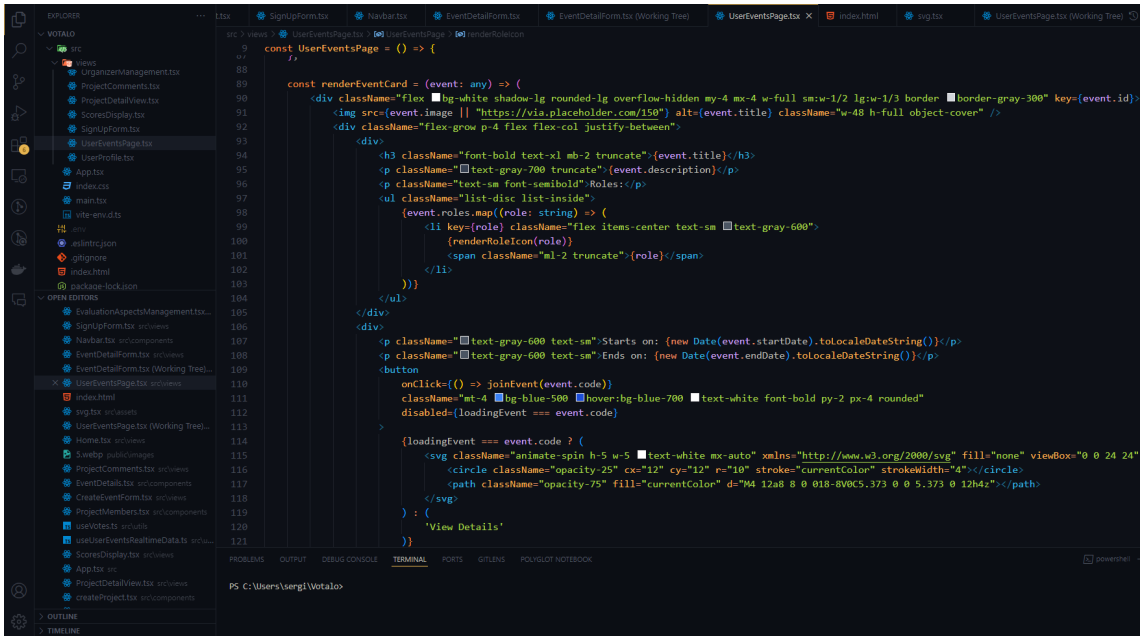

Figura 3.1: Interfaz de Visual Studio Code

3.2 Visual Studio 2022

Visual Studio 2022[2] es un entorno de desarrollo integrado (IDE) desarrollado por Microsoft, ampliamente reconocido por su robustez y su capacidad para manejar proyectos de gran escala y complejidad. Es una herramienta integral que proporciona todo lo necesario para el desarrollo de aplicaciones, desde la codificación hasta la depuración y el despliegue.

Este IDE también soporta una amplia gama de extensiones que mejoran la productividad y la calidad del código, incluyendo herramientas de análisis de código y refactorización.

En el desarrollo de Votalo, Visual Studio 2022 se ha utilizado para la programación del backend en C# y .NET. La elección de esta herramienta se debe a su soporte avanzado para C# y el framework .NET, lo que facilita el desarrollo de API REST eficientes y escalables, necesarias para la funcionalidad del backend de Votalo.

En la Figura 3.2 podemos observar cómo es el entorno de desarrollo de Visual Studio 2022

[2] https://visualstudio.microsoft.com/es/vs/

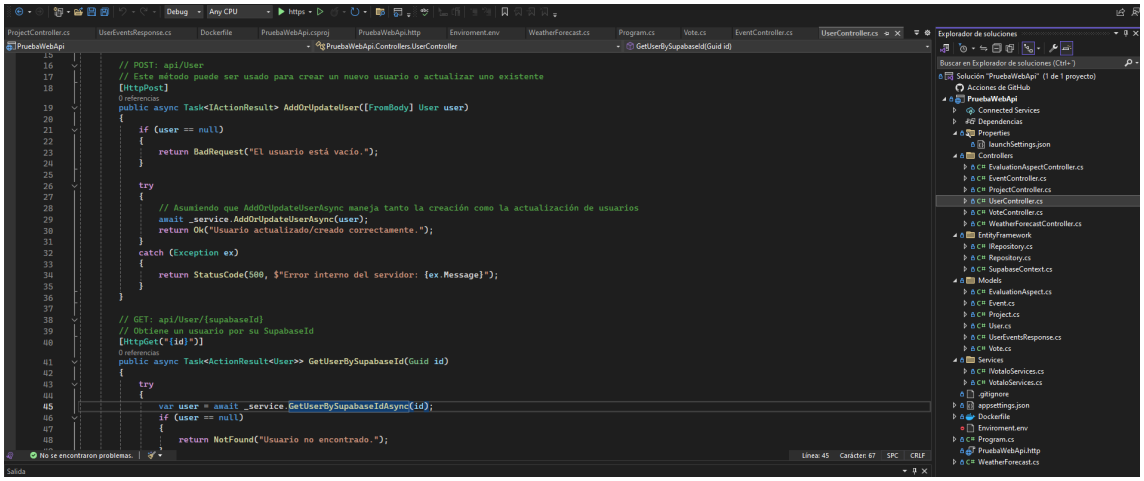

Figura 3.2: Interfaz de Visual Studio 2022

3.3 React

React[3] es una biblioteca de JavaScript de código abierto desarrollada por Facebook, utilizada para construir interfaces de usuario, especialmente para aplicaciones web de una sola página (SPA). Su enfoque en la creación de componentes reutilizables permite a los desarrolladores construir interfaces de manera modular y eficiente.

Figura 3.3: Código en React

En la Figura 3.3 podemos observar cómo se estructura el código de un componente en React

Una de las características mas destacadas de React es su sistema de manejo del estado y su virtual DOM, que optimiza las actualizaciones de la interfaz de usuario. Además, React tiene un amplio ecosistema de herramientas y librerías que facilitan la integración con otras tecnologías y la expansión de sus funcionalidades.

En el desarrollo de Votalo, se decidió utilizar React para la programación del frontend debido a las facilidades que ofrece para la construcción de interfaces de usuario dinámicas y responsivas. A pesar de no tener experiencia previa con esta biblioteca, su extensa documentación y la gran cantidad de recursos disponibles me permitieron aprender a utilizarla rapidamente.

[3] https://es.react.dev/

3.4 Jest

Jest[4] es un marco de pruebas en JavaScript desarrollado por Facebook, ampliamente utilizado para la creación y ejecución de pruebas unitarias, de integración y de extremo a extremo en aplicaciones JavaScript. Es especialmente popular en el desarrollo con React debido a su integración y facilidad de uso.

Una de las características más destacadas de Jest es su configuración mínima y facilidad de uso. La mayoría de los proyectos pueden comenzar a utilizar Jest sin necesidad de configuraciones complejas. Además, Jest ejecuta pruebas en paralelo por defecto, lo que mejora significativamente el tiempo de ejecución de las pruebas.

```
PS npm test

> learning-jest@1.0.0 test
> jest

 PASS  ./index.test.js
  FizzBuzz
    √ [3] should result in "fizz" (2 ms)
    √ [5] should result in "buzz" (1 ms)
    √ [15] should result in "fizzbuzz" (1 ms)
    √ [1,2,3] should result in "1, 2, fizz" (1 ms)

Test Suites: 1 passed, 1 total
Tests:       4 passed, 4 total
Snapshots:   0 total
Time:        0.586 s, estimated 1 s
Ran all test suites.
```

Figura 3.4: Salida al ejecutar tests con Jest

En la Figura 3.4 podemos observar la salida que muestra Jest al ejecutar varios tests correctamente.

En el proyecto Votalo, Jest se ha utilizado ampliamente para asegurar la calidad y funcionalidad del código a través de diferentes tipos de pruebas que serán explicadas con mas detalle en apartados posteriores.

3.5 Vite

Vite[5] es una herramienta de construcción de frontend extremadamente rápida. Vite proporciona una experiencia de desarrollo altamente optimizada, con arranques de servidor muy rápidos y compilaciones eficientes, utilizando módulos ES nativos.

Una de las ventajas clave de Vite es su capacidad para proporcionar recargas instantáneas de módulos (HMR), lo que permite a los desarrolladores ver los cambios de código reflejados inmediatamente en la aplicación en desarrollo, sin necesidad de una recompilación completa, lo cual agiliza enormemente el desarrollo.

[4] https://jestjs.io/es-ES/
[5] https://vitejs.dev/

En Votalo, Vite se utilizó para crear y gestionar el entorno de desarrollo de la aplicación en React. Su velocidad y eficiencia mejoraron significativamente la experiencia de desarrollo.

3.6 C#

C#[6] es un lenguaje de programación desarrollado por Microsoft, que forma parte del framework .NET. Es un lenguaje moderno, orientado a objetos y de propósito general, que está diseñado para ser simple y fácil de usar.

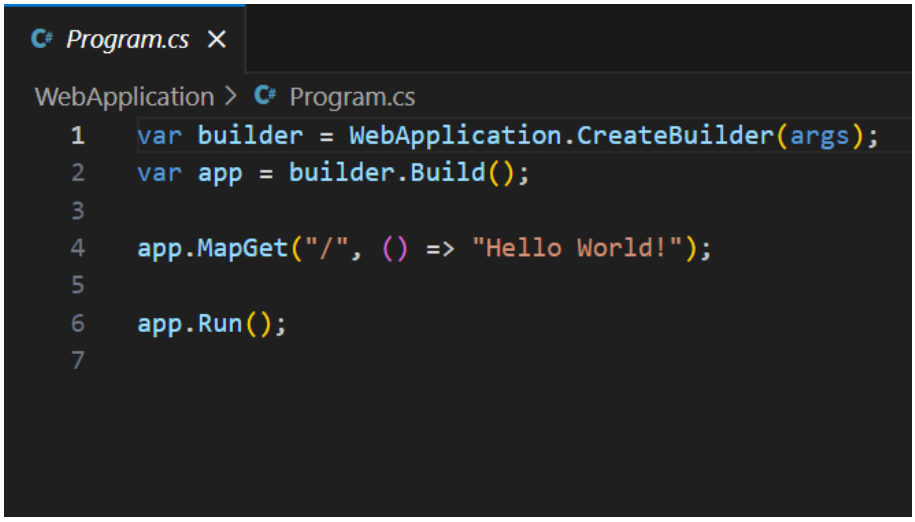

Figura 3.5: Código en C#

En la Figura 4.12 podemos observar cómo se estructura el código de un componente en C#.

Su integración con el framework .NET permite el acceso a una amplia gama de bibliotecas y herramientas que facilitan el desarrollo de aplicaciones robustas y escalables.

En el desarrollo de Votalo, se decidió utilizar C# para la programación del backend debido a sus capacidades avanzadas para la creación de API REST y su excelente rendimiento. C# junto con ASP.NET Core proporciona un entorno ideal para construir servicios web eficientes.

[6] https://dotnet.microsoft.com/es-es/languages/csharp

3.7 Docker

Docker[7] es una plataforma de código abierto que automatiza la implementación de aplicaciones dentro de contenedores de software. Los contenedores son entornos aislados que incluyen todo lo necesario para ejecutar una aplicación, como el código, las bibliotecas y las dependencias, lo que asegura que la aplicación se ejecute de manera consistente sin importar el entorno donde se despliegue.

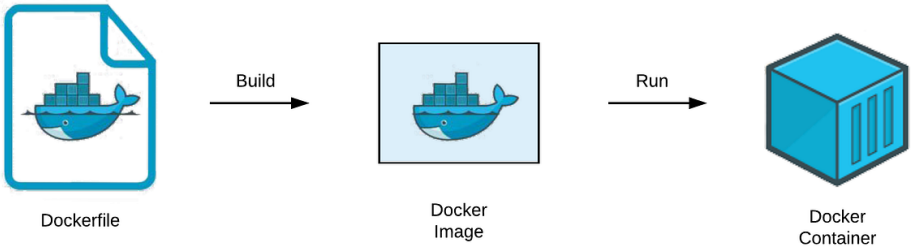

Figura 3.6: Funcionamiento de Docker

Una de las principales ventajas de Docker es su capacidad para simplificar y acelerar el proceso de desarrollo y despliegue de aplicaciones. Al encapsular una aplicación y sus dependencias en un contenedor, Docker elimina los problemas de "funciona en mi máquina", permitiendo a los desarrolladores centrarse en la creación de software sin preocuparse por las discrepancias del entorno.

En el desarrollo de Votalo, se decidió utilizar Docker para dockerizar el backend, facilitando su despliegue en diversos entornos. Al crear una imagen Docker del backend, fue posible subirlo a un servicio de hosting como fly.io. Esta solución permitió una implementación rápida del backend, asegurando que se ejecutara de la misma forma en todos los entornos, desde desarrollo hasta producción.

3.8 Supabase

Supabase[8] es una plataforma de código abierto que proporciona una alternativa a Firebase, ofreciendo una gama de servicios backend como bases de datos en tiempo real, autenticación de usuarios y almacenamiento. Construida sobre PostgreSQL, Supabase permite a los desarrolladores crear aplicaciones escalables y seguras con facilidad.

Una de las características más destacadas de Supabase es su facilidad de uso, combinada con la potencia y flexibilidad de PostgreSQL. Además, Supabase proporciona una excelente integración con React, lo que facilita la creación de aplicaciones web modernas y responsivas

[7] https://www.docker.com/
[8] https://supabase.com/

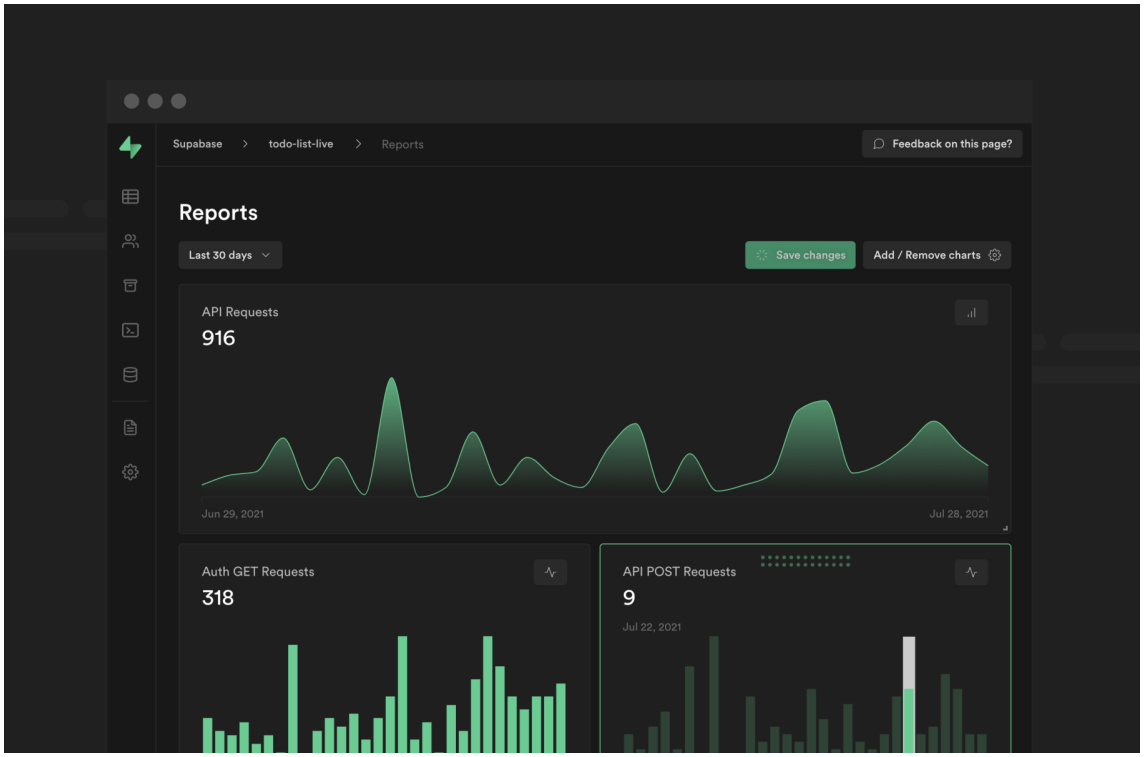

Figura 3.7: Dashboard de Supabase

En la Figura 3.7 podemos observar cómo es el entorno de Supabase.

En el desarrollo de Votalo, se decidió utilizar Supabase debido a varias razones. Primero, la familiaridad con la plataforma permitió un desarrollo más rápido y eficiente. Segundo, React ofrece un soporte excelente para la integración con Supabase, lo que facilitó la implementación de funcionalidades críticas como la autenticación de usuarios. Además, Supabase es una solución gratuita que proporciona servicios robustos y escalables sin incurrir en costos adicionales, lo que es ideal para proyectos en crecimiento. La elección de Supabase permitió gestionar la autenticación de usuarios de manera eficaz , mejorando la experiencia general de los usuarios de Votalo.

3.9 Cloudfare

Cloudflare[9] Pages es una plataforma de alojamiento web estática que permite a los desarrolladores desplegar sus sitios frontend de manera rápida y eficiente. Ofrece integración continua (CI) y despliegue continuo (CD) automatizados, lo que facilita el desarrollo y la actualización del sitio sin interrupciones. Además, proporciona un rendimiento mejorado y una gran escalabilidad gracias a la infraestructura global de Cloudflare.

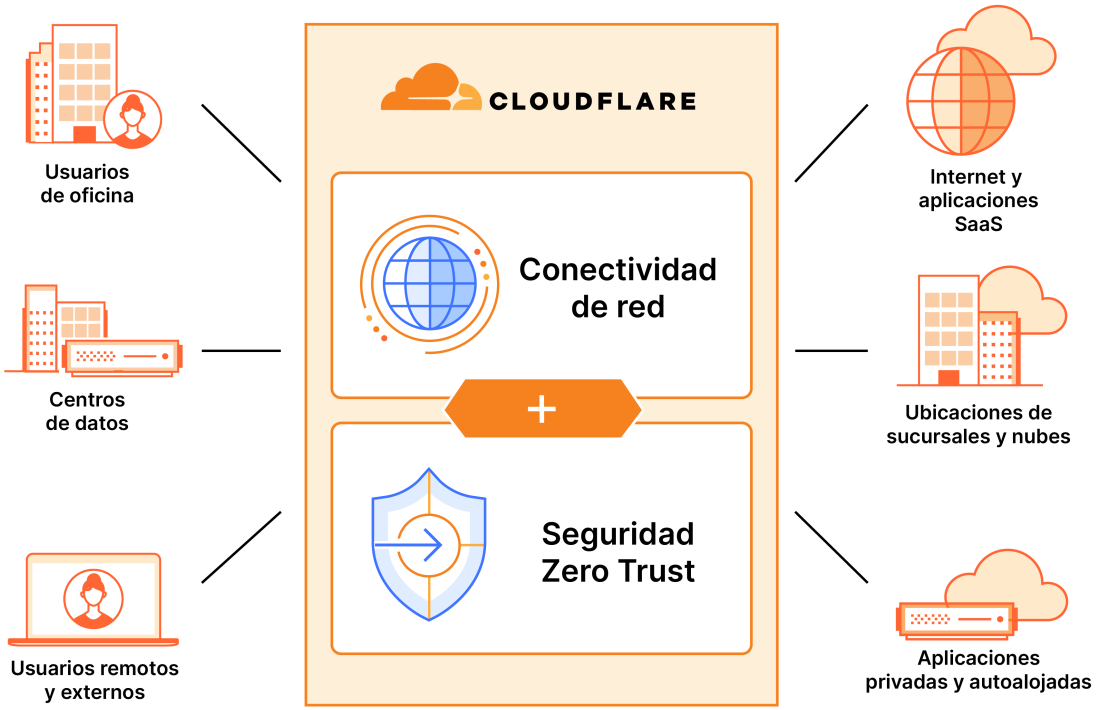

Figura 3.8: Aquitectura de Cloudfare

En el desarrollo de Votalo, se decidió utilizar Cloudflare Pages para alojar el frontend de la aplicación. Esta elección se debe a varias razones. Primero, la plataforma permite un despliegue rápido y sencillo del frontend construido en React. La integración con herramientas de CI/CD asegura que cada cambio en el código se despliegue automáticamente, reduciendo el tiempo de inactividad y mejorando la eficiencia del flujo de trabajo.

Además, Cloudflare Pages es una solución gratuita que proporciona un rendimiento y una seguridad excepcionales, lo que es ideal para un proyecto en crecimiento como Votalo. La utilización de Cloudflare Pages ha permitido que el frontend de Votalo sea altamente accesible, seguro y rápido, mejorando la experiencia de los usuarios finales.

[9] https://www.cloudflare.com/es-es/

3.10 Fly.io

Fly.io[10] es una plataforma de alojamiento en la nube que permite a los desarrolladores desplegar aplicaciones con facilidad.

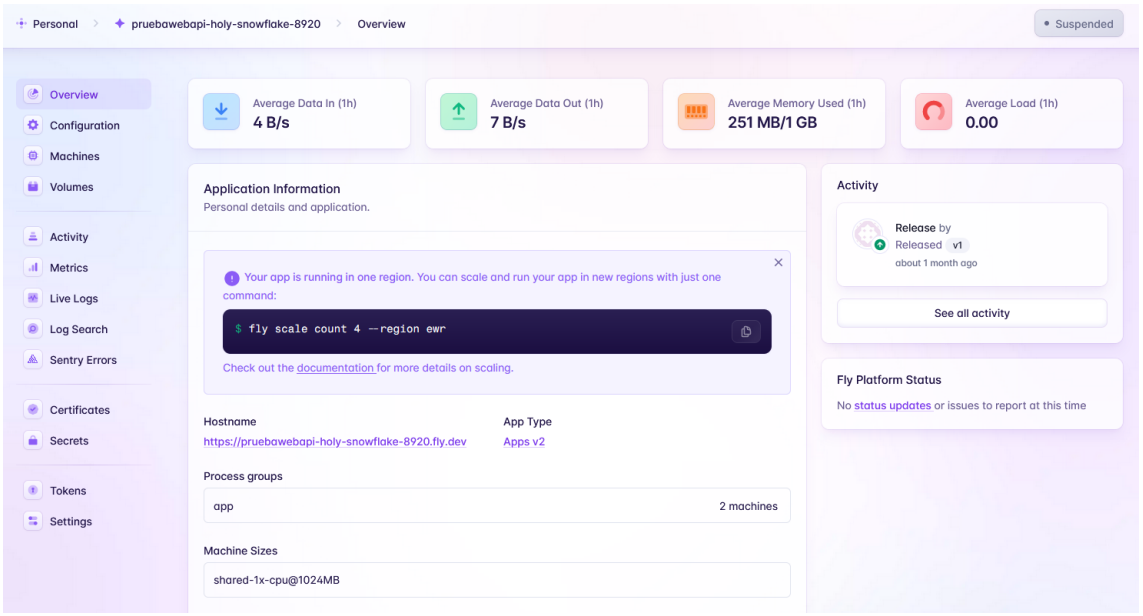

Figura 3.9: Dashboard de fly.io

En la Figura 3.9 podemos observar cómo es el entorno de Fly.io.

Una de las ventajas más destacadas de Fly.io es su modelo de servicio gratuito, que no hiberna por inactividad, lo que asegura que las aplicaciones estén siempre disponibles sin incurrir en costos adicionales. Además, Fly.io soporta contenedores Docker, lo que facilita la implementación y el escalado de aplicaciones.

En el desarrollo de Votalo, se decidió utilizar Fly.io para alojar el backend de la aplicación. Esta elección se basa en varios factores. Primero, Fly.io ofrece un servicio gratuito que se ajusta perfectamente a las necesidades del proyecto, asegurando que el backend esté siempre activo y disponible para los usuarios sin costos adicionales. Segundo, el proceso de despliegue se simplificó gracias a la compatibilidad de Fly.io con Docker. Dockerizar el backend permitió empaquetar la aplicación y sus dependencias en un contenedor, asegurando una implementación sin problemas en Fly.io.

[10] https://fly.io/

3.11 Git

Git[11] es un sistema de control de versiones distribuido de código abierto que permite a los desarrolladores rastrear y gestionar los cambios en el código fuente a lo largo del tiempo. Es una herramienta fundamental en el desarrollo de software moderno, ya que facilita la colaboración, el manejo de versiones y la integración continua.

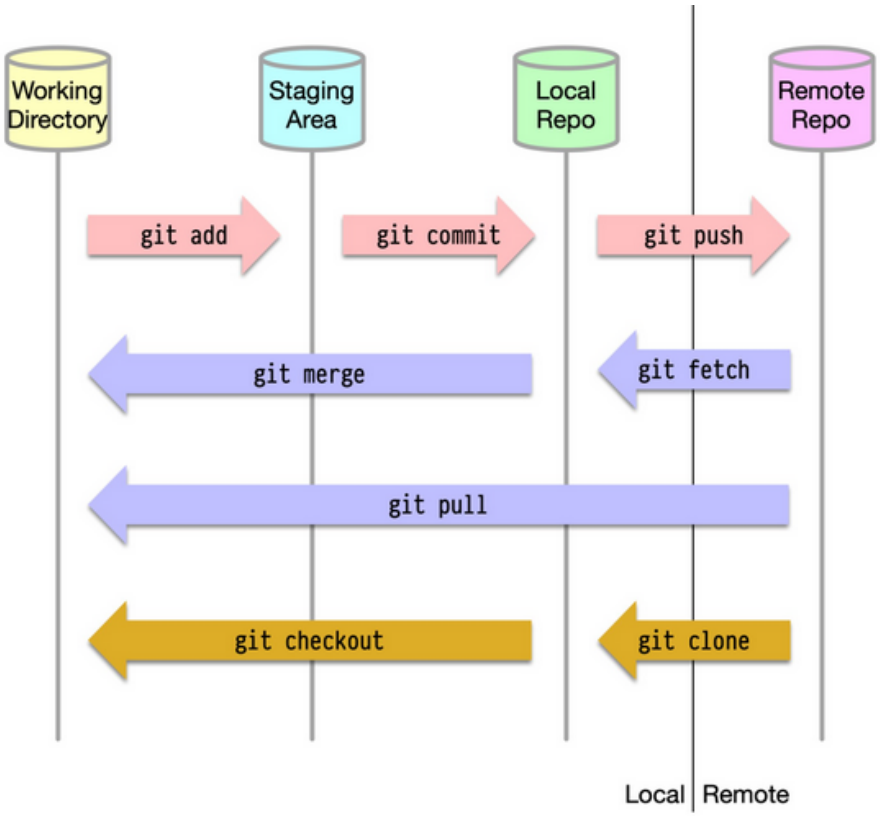

Figura 3.10: Funcionamiento de Git

En el desarrollo de Votalo, se utilizó Git como sistema de control de versiones tanto para el frontend como para el backend, aunque en este proyecto he trabajado solo. Git ha ayudado a gestionar el código fuente, permitiendo realizar un seguimiento detallado de todos los cambios realizados durante el desarrollo. Además, Git facilitó la integración de cambios y la recuperación de versiones anteriores en caso de ser necesario, proporcionando una mayor seguridad y confianza en el desarrollo del proyecto.

[11] https://git-scm.com/

CAPÍTULO 4
Desarrollo de la solución

4.1 Metodología

4.1.1. Proceso realizado

El desarrollo de Votalo se basó en la metodología Lean Startup[8], centrándose en la creación de un Producto Mínimo Viable[9] y la realización de experimentos iterativos. Esta metodología se combinó con un enfoque ágil, utilizando sprints para planificar y ejecutar el desarrollo del proyecto.

La estrategia de MVP-Experimento, basada en el método Lean Startup, implicó la creación de versiones iniciales del producto con las características mínimas necesarias para resolver problemas específicos de los usuarios y validar hipótesis críticas. Cada MVP se desarrolló en ciclos iterativos de un mes y una semana, denominados sprints. Este enfoque permitió al equipo lanzar rápidamente versiones funcionales del producto, recolectar datos y feedback de los usuarios, y utilizar esta información para guiar los desarrollos futuros.

El proyecto se dividió en dos sprints, cada uno con una duración de un mes y una semana. Al final de cada sprint, se generó un MVP que se utilizó para validar las funcionalidades desarrolladas y obtener retroalimentación para el desarrollo futuro.

Este enfoque iterativo no solo permitió una rápida adaptación a las necesidades cambiantes del usuario, sino que también minimizó el riesgo de desarrollar funcionalidades innecesarias o mal alineadas con el mercado. Al integrar la retroalimentación continua y los ciclos de validación rápidos, Votalo se desarrolló de manera eficiente, alineándose con las expectativas del mercado y asegurando una mayor probabilidad de éxito una vez lanzado.

4.1.2. Gestión del trabajo

Para la planificación y seguimiento del proyecto, la tecnología usada fue Trello, ya que es una aplicacion que ya hemos utilizado en la carrera, y que proporciona una forma muy sencilla de organizar el trabajo a la hora de desarrollar software. Se creó un tablero con las siguientes columnas:

- **Backlog**
  - Para mantener una lista de todas las tareas pendientes de ser especificadas y desarrolladas.
- **Especificar requisitos**
  - Donde se detallaban los requisitos de cada tarea para que quedara claro el propósito de cada UT.
- **Programar**
  - Después de haber definido sus requisitos, las tareas se pasaban a programar durante el tiempo que se estuviera desarrollando el trabajo de programación asociado. Cuando ese trabajo se acababa, se pasaba a la sección de pruebas.
- **Pruebas**
  - En esta sección se trataba de probar las UT que acababan de salir de programación para comprobar su correcto comportamiento en todas las circunstancias posibles. Se realizaban comprobaciones tanto a nivel de código como a nivel de interacción del usuario con la interfaz de la aplicación para comprobar el correcto funcionamiento de las funcionalidades de Votalo.
- **Terminado**
  - Después de todo el proceso, las UT acaban en la sección de Terminado, donde se da por cerrada una UT.

Mas adelante se mostrará más en detalle el uso de esta herramienta en los diferentes sprints.

4.2 Requisitos

4.2.1. Requisitos Funcionales

Los requisitos funcionales de Votalo se definen a través de Casos de Uso (CU) que describen las interacciones entre los usuarios y el sistema. En la figura 4.1, se presenta un Diagrama de Casos de Uso y una breve descripción de cada uno.

Diagrama de Casos de Uso

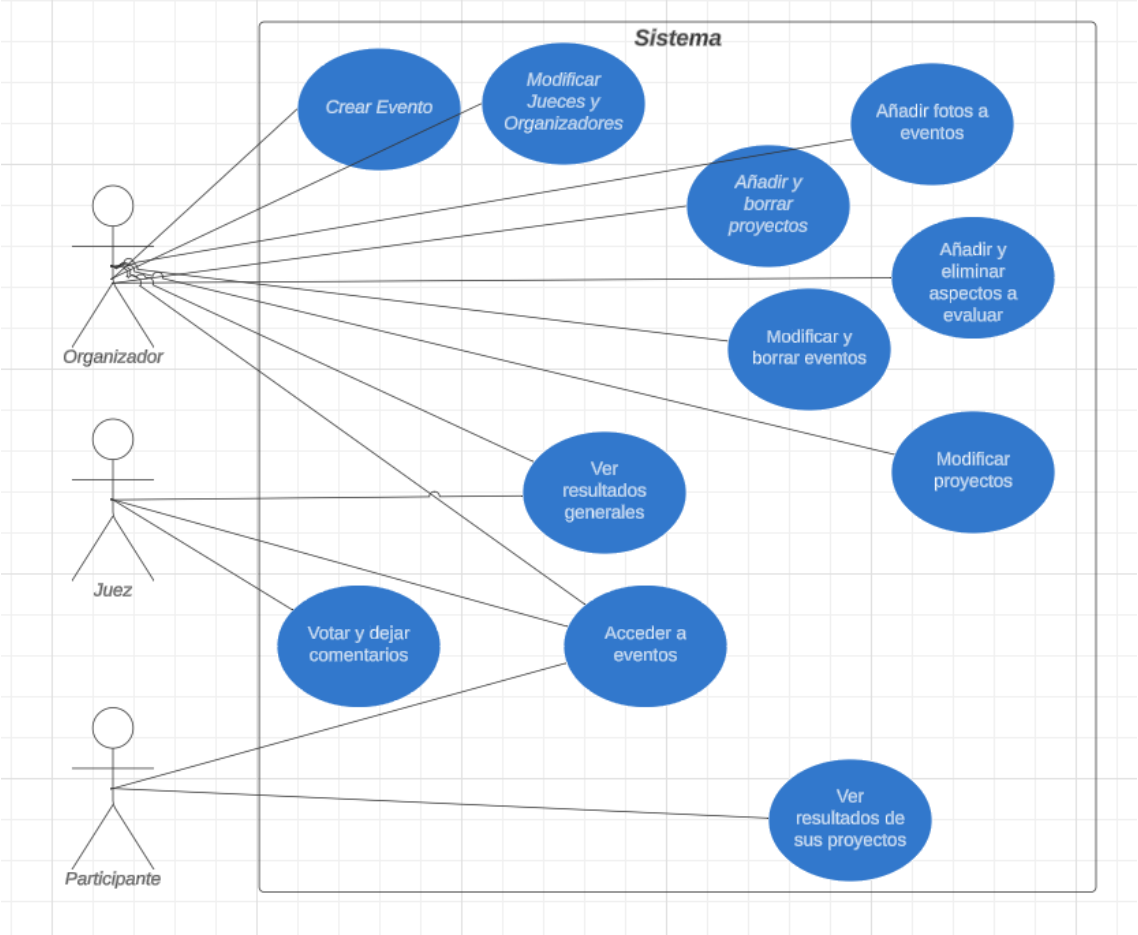

Figura 4.1: Diagrama de Casos de Uso

Descripción de Casos de Uso

1. **Crear Evento:**
   - Actor(es): Organizador
   - Descripción: Permite a los organizadores crear nuevos eventos ingresando detalles como el nombre del evento, descripción, imagen, fecha y lugar.
2. **Modificar Jueces y Organizadores:**
   - Actor(es): Organizador
   - Descripción: Permite a los organizadores modificar la lista de jueces y organizadores del evento.
3. **Añadir y borrar proyectos:**
   - Actor(es): Organizador
   - Descripción: Permite a los organizadores añadir nuevos proyectos al evento y borrar los existentes.
4. **Añadir y eliminar aspectos a evaluar:**
   - Actor(es): Organizador
   - Descripción: Permite a los organizadores definir y eliminar los aspectos a evaluar del evento.

5. **Añadir fotos a eventos:**
   - Actor(es): Organizador
   - Descripción: Permite a los organizadores añadir fotos a los eventos para mejorar la visualización y presentación.
6. **Modificar y borrar eventos:**
   - Actor(es): Organizador
   - Descripción: Permite a los organizadores modificar los detalles de los eventos y borrarlos si es necesario.
7. **Modificar proyectos:**
   - Actor(es): Organizador
   - Descripción: Permite a los organizadores modificar los detalles de los proyectos dentro de un evento.
8. **Acceder a eventos:**
   - Actor(es): Participantes, Jueces y Organizadores
   - Descripción: Permite a los usuarios acceder a los eventos a los que ha sido añadido.
9. **Ver resultados generales:**
   - Actor(es): Jueces y Organizadores
   - Descripción: Permite a los usuarios ver los resultados generales de los eventos, asi como ver las notas y comentarios de todos los proyectos.
10. **Ver resultados de sus proyectos:**
    - Actor(es): Participante
    - Descripción: Permite a los participantes ver los resultados de sus proyectos en el evento.
11. **Votar y dejar comentarios:**
    - Actor(es): Jueces
    - Descripción: Permite a los jueces evaluar los proyectos, asignar puntuaciones y dejar comentarios en cada aspecto a evaluar.

4.2.2. Requisitos No Funcionales

1. **Rendimiento:**
   - La aplicación debe ser capaz de manejar múltiples solicitudes concurrentes.
   - Las actualizaciones de datos deben reflejarse en tiempo real para todos los usuarios.
2. **Usabilidad:**
   - La interfaz de usuario debe ser intuitiva y fácil de navegar.
   - Deben proporcionarse guías y explicaciones claras en cada sección de la aplicación para facilitar su uso.
3. **Seguridad:**
   - La aplicación debe asegurar que solo usuarios autorizados puedan acceder y modificar los eventos.
   - Los datos de los usuarios y eventos deben estar protegidos contra accesos no autorizados.

4. **Disponibilidad:**
   - La aplicación debe estar disponible y operativa el 99.9 % del tiempo, evitando tiempos de inactividad prolongados.
5. **Compatibilidad:**
   - La aplicación debe ser compatible con los principales navegadores web y dispositivos móviles.

4.3 Diseño

La arquitectura de Votalo sigue un modelo de tres capas compuesto por la capa de presentación, la capa de lógica de negocio y la capa de acceso a datos. Este diseño modular facilita la escalabilidad, el mantenimiento y la reutilización de componentes.

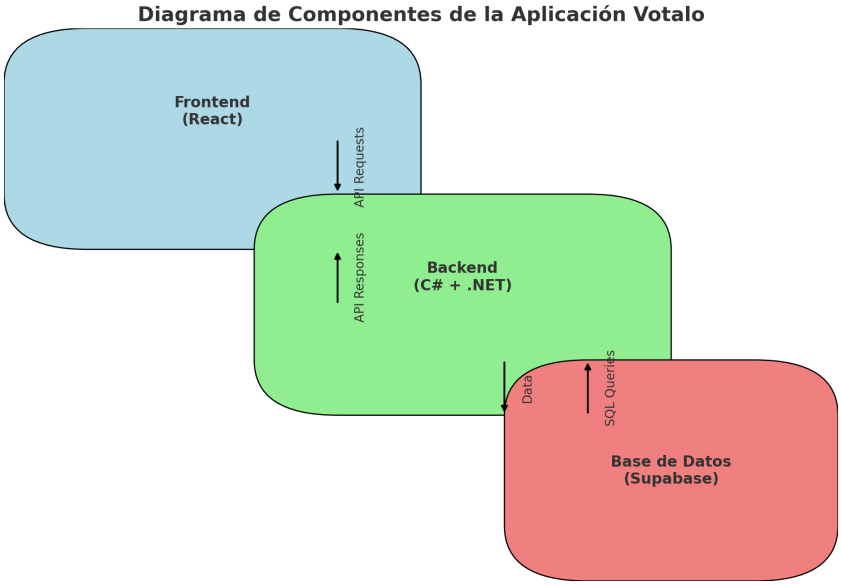

Figura 4.2: Diagrama de Componentes

En la figura 4.2 se puede ver ilustrada la arquitectura de nuestra aplicación y como se relacionan las diferentes capas entre ellas.

Capa de Presentación

La capa de presentación está desarrollada utilizando React.js para el frontend. Esta biblioteca permite la creación de interfaces de usuario dinámicas y responsivas, facilitando la interacción del usuario con la aplicación.

Capa de Lógica de Negocio

La capa de lógica de negocio está implementada en C# utilizando ASP.NET Core. Esta capa contiene la lógica central de la aplicación, gestionando las reglas de negocio, el procesamiento de datos y la comunicación entre la capa de presentación y la capa de acceso a datos. Los controladores de API RESTful se utilizan para manejar las solicitudes HTTP del frontend y proporcionar respuestas adecuadas.

Capa de Acceso a Datos

La capa de acceso a datos utiliza Supabase. Supabase gestiona la base de datos en tiempo real, proporcionando servicios de autenticación, almacenamiento y consultas optimizadas. Esta capa se encarga de la persistencia de datos y la ejecución de consultas a la base de datos.

La base de datos de Votalo está estructurada para soportar las funcionalidades principales de la aplicación. Las tablas más relevantes incluyen:

- **User:** Almacena información de los usuarios, incluyendo credenciales y roles (juez, organizador, participante).
- **Event:** Contiene detalles sobre los eventos, como nombre, descripción, fecha y organizadores.
- **Project:** Registra los proyectos participantes en cada evento, incluyendo título, descripción y miembros del equipo.
- **Vote:** Guarda las votaciones realizadas por los jueces, incluyendo los criterios evaluados y las puntuaciones asignadas.
- **EvaluationAspect:** Define los aspectos de evaluación que serán utilizados en las votaciones, permitiendo una evaluación detallada y personalizada.

La estructura de la BD se corresponde a la definida en el diagrama de clases. Este muestra las principales clases y sus relaciones como se puede ver en la figura 4.3:

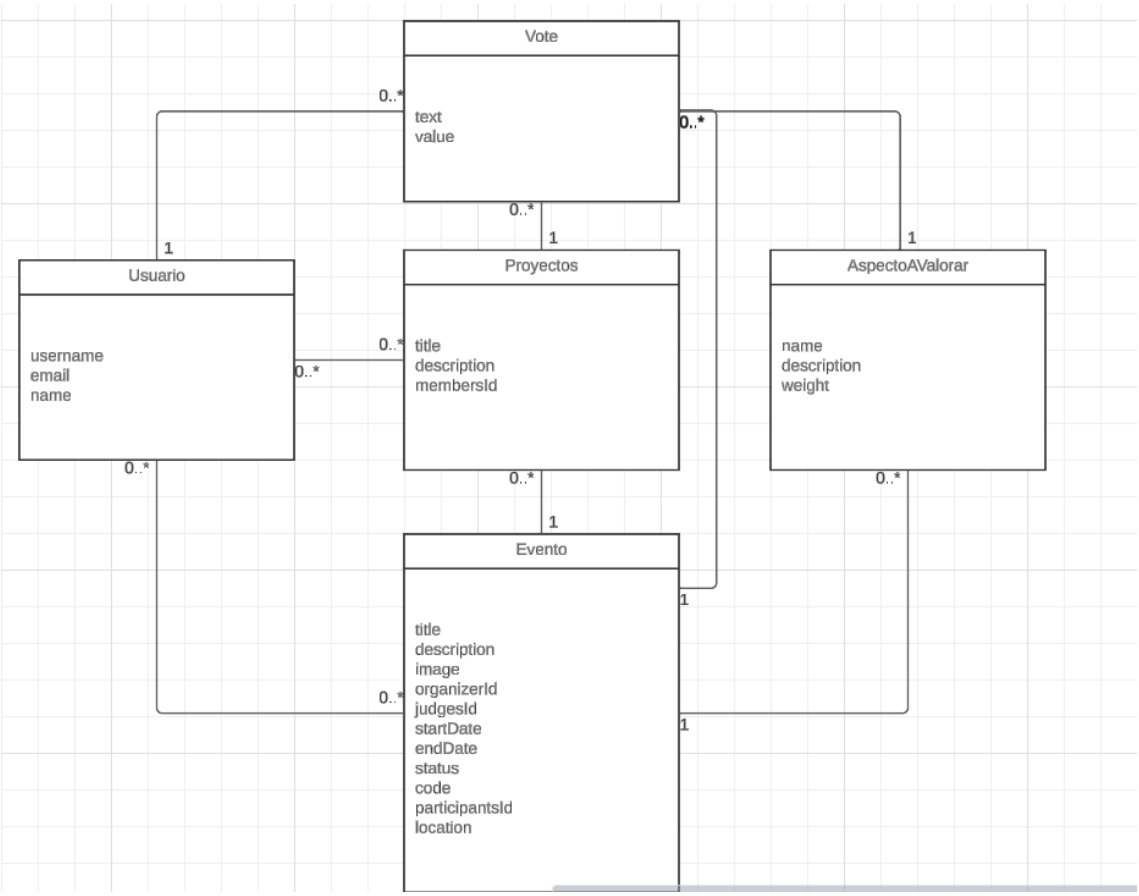

Figura 4.3: Diagrama de clases

- **Usuario:** Contiene los atributos `username`, `email`, y `name`. Un usuario puede participar en múltiples eventos y proyectos, y puede actuar como juez en votaciones.
- **Evento:** Contiene los atributos `title`, `description`, `image`, `organizerId`, `judgesId`, `startDate`, `endDate`, `status`, `code`, `participantsId`, y `location`. Un evento puede tener múltiples proyectos y aspectos a valorar.
- **Proyectos:** Contiene los atributos `title`, `description`, y `membersId`. Cada proyecto pertenece a un evento específico y puede recibir múltiples votos.
- **Vote:** Contiene los atributos `text` y `value`. Un voto está relacionado con un proyecto, un evento, un aspecto a valorar, y el juez (usuario) que realiza la votación.
- **AspectoAValorar:** Contiene los atributos `name`, `description`, y `weight`. Cada aspecto a valorar está asociado a un evento y puede tener múltiples votos asociados.

Las relaciones entre estas clases están definidas como sigue:

- Un **Usuario** puede participar en múltiples **Eventos**.
- Un **Evento** puede tener múltiples **Proyectos**.
- Un **Proyecto** puede recibir múltiples **Votos**.
- Un **AspectoAValorar** está asociado a múltiples **Votos** y a un solo **Evento**.
- Un **Usuario** puede emitir múltiples **Votos** y participar en múltiples **Proyectos**.

4.4 Programación

Para el desarrollo de Votalo, se decidió desarrollar el frontend en React, ya que, aunque no tenía experiencia con esta tecnología, es una biblioteca moderna enfocada al desarrollo web, con una gran cantidad de librerías y documentación que facilitan el desarrollo.

El backend se desarrolló en C#, ya que es un lenguaje que ofrece muchas facilidades para el desarrollo de API REST y con el cual ya estaba familiarizado.

En esta sección se describen las anécdotas del desarrollo, la aplicación de patrones de diseño, refactorings y desafíos enfrentados durante la creación de la aplicación Votalo. La aplicación se ha separado en frontend y backend, con el frontend actuando como la capa de presentación que consume datos de la API REST del backend. El backend, se encarga de realizar las llamadas a la base de datos y se diseñó siguiendo el patrón Repository, muy útil para facilitar el acceso a la base de datos.

4.4.1. Frontend

El desarrollo del frontend de Votalo presentó numerosos desafíos. Al iniciar este proyecto, no tenía experiencia previa con Vite ni con React, lo cual representó un reto considerable. Además, la selección de librerías tanto para estilos como para añadir funcionalidades también fue un problema, ya que nunca había utilizado estas herramientas antes. A continuación, se describen algunos de los problemas que surgieron durante el desarrollo y cómo fueron abordados.

Registro de Usuarios

Uno de los primeros desafíos fue implementar un sistema de registro de usuarios seguro, con contraseñas encriptadas y datos protegidos. Anteriormente, en otros proyectos no comerciales, se utilizaba una validación de usuarios muy laxa. Sin embargo, dado que Votalo está destinada a ser comercializada, era crucial asegurar la autenticidad y seguridad de los usuarios.

Supabase facilitó enormemente esta tarea, proporcionando servicios de autenticación que utilizamos para validar usuarios con Google. Esto también permitió que las sesiones de los usuarios se almacenaran en cookies, evitando que tuvieran que iniciar sesión cada vez que accedieran a la aplicación. Sin embargo, surgió un problema: el registro de usuarios con Google no permitía modificar las tablas de usuario para añadir nuevos campos, limitando la información a nombre, email y avatar.

Para solucionar esto, se añadió un trigger con una función que crea un nuevo registro en una tabla pública cada vez que se registra un nuevo usuario, permitiendo añadir campos adicionales como los eventos en los que ha participado.

```
const SignUpForm = () => {
    const navigate = useNavigate();
    const { setUser } = useAuth();
    const handleGoogleSignUp = async () => {
        const { error } = await supabase.auth.signInWithOAuth({
            provider: 'google',
            options: {
                redirectTo: '/',
            },
        });

        if (error) {
            alert(`Error: ${error.message}`);
            return;
        }
    };
```

Figura 4.4: Código para registrar usuarios

En la figura 4.4 se muestra el código implementado con la libreria de Supabase para registrar usuarios mediante la cuenta de Google

Manejo de Estados Globales

Otro desafío importante al que me enfrenté, debido a mi falta de familiaridad con React, fue el manejo de estados globales.

En Votalo, esto es indispensable para varias partes. En primer lugar, para guardar el usuario logueado, ya que en diferentes componentes de la aplicación debe verificarse el usuario y obtener su email para realizar diversas comprobaciones, como su rol dentro de cada evento. Al ser este un dato que no se espera que cambie constantemente, se decidió utilizar el contexto de la propia librería de React. De esta manera, se almacena el usuario en un componente, y con este mismo componente envolvemos toda la aplicación. De esta forma, con la creación de un custom Hook llamado `useAuth`, podíamos acceder al

usuario logueado desde cualquier parte de la aplicación, evitando así tener que pasar este parámetro por todos los componentes.

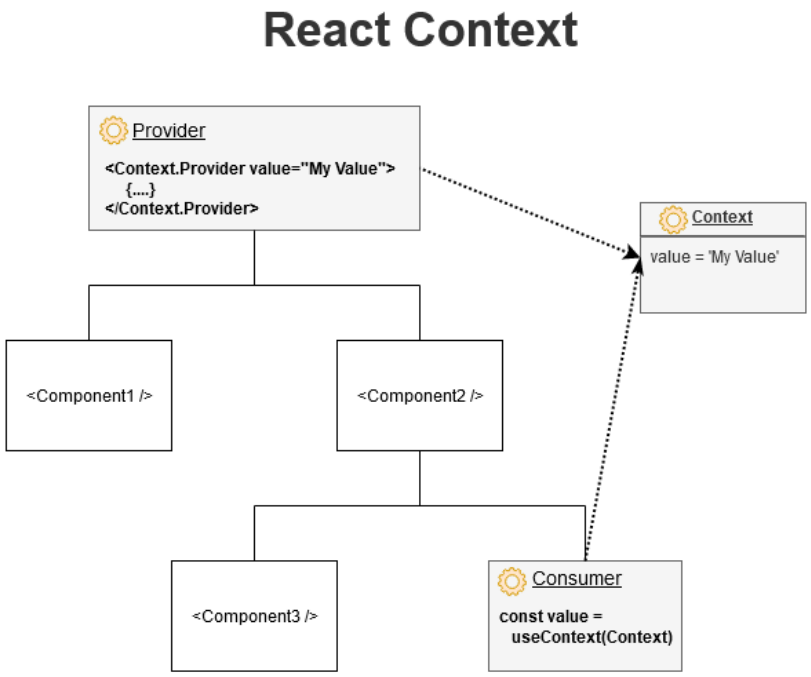

Figura 4.5: Estructura de estado global con React Context

En la figura 4.5 se muestra la estructura que utiliza el context, y cómo su uso permite acceder a esta variable desde cualquier componente.

Pero esto no era lo único que quería tener como estado global, ya que dentro de cada evento teníamos diferentes aspectos como el propio evento, los proyectos de este evento, los aspectos a evaluar y los votos.

Todo esto se utiliza constantemente en las diferentes ventanas del evento, por lo que era necesario almacenarlo en un contexto global. A diferencia del usuario, estos valores sí que se espera que cambien constantemente, por lo que implementar el mismo método que teníamos para el usuario no era la decisión más eficiente. Por esto se decidió utilizar la biblioteca Zustand, la cual nos permite generar estados globales de manera sencilla y está optimizada para almacenar valores que cambian constantemente. De esta forma, cada vez que accedíamos a un evento, se cargaban los datos de este y se almacenaban en las diferentes stores, facilitando mucho la implementación de las diferentes pantallas.

```ts
// stores/useVoteStore.js
import { create } from 'zustand';
import { Vote } from '../types/vote';

type State = {
    votes: Vote[];
    setVotes: (votes: Vote[]) => void;
    updateVote: (aspectId: number, vote: Vote) => void;
};

const useVoteStore = create<State>((set) => ({
    votes: [],
    setVotes: (votes) => set({ votes }),
    updateVote: (aspectId, updatedVote) => set(state => ({
        votes: state.votes.map(vote => vote.evaluationAspectId === aspectId ? { ...vote, ...updatedVote } : vote),
    })),
}));

export default useVoteStore;
```

Figura 4.6: Código del store de votos creado con Zustand

En la figura 4.6 puede observarse la creación de una store global para los votos de un evento utilizando la librería Zustand[1], estableciendo el estado de los votos en un array, y los metodos setVotes y updateVote.

Al implementarlo por primera vez, se detectó un problema: estos valores solo se actualizaban al recargar la página, ya que los datos se cargaban con la creación del componente. Esto suponía muchos problemas, tanto a nivel de seguridad (por ejemplo, un usuario que anteriormente fuera juez podría seguir votando siempre y cuando no recargara la página) como a nivel de usabilidad (si se añadían nuevos proyectos, nuevos aspectos a evaluar o nuevos votos, estos no se veían reflejados hasta que se recargara la pantalla, perdiendo así el dinamismo y la constante retroalimentación que se buscaba). Ante este problema, se decidió utilizar nuevamente la librería de Supabase, que nos permite suscribirnos a canales para captar las actualizaciones de datos en las tablas en tiempo real. De esta forma, se consiguió que todos los datos se actualizaran en tiempo real, logrando una experiencia de usuario mucho mejor y mejorando la seguridad de la aplicación.

La forma en la que funciona es que al cargar el componente por primera vez, se hace una llamada a nuestro backend, que nos devuelve los valores en ese instante, y estos valores se guardan en las diferentes stores. Luego, se suscribe a los cambios en las tablas necesarias utilizando la librería de Supabase en React, y al detectar cambios, estos actualizan los valores de las stores, ya sea añadiendo, modificando o eliminando elementos. Este proceso tuvo bastantes complicaciones, como la documentación de Supabase desactualizada o el intento de diferentes formas de manejar los datos, pero creo que fue una de las mejoras más significativas, o de los refactorings más importantes de la aplicación.

Navegación en React

React está diseñado para generar SPA (Single Page Applications), pero la inexperiencia con esta tecnología me llevó, en los primeros ciclos de desarrollo, a abusar del `navigate` de React para las diferentes ventanas o partes de mi aplicación. Esto provocaba que la aplicación tuviera que recargarse constantemente y complicaba la implementación de diversas características. Después de ver los resultados del primer experimento, y al estar ya más familiarizado con React, se refactorizó gran parte del código para evitar el uso

[1] https://zustand-demo.pmnd.rs/

de `navigate` y explotar mucho más el poder de React para generar SPA. Esto conllevó la necesidad de separar o estructurar de mejor manera muchos componentes de la aplicación, para evitar que hubiera código muy largo o complicado. Al final, se obtuvo una web mucho más rápida y fluida, lo cual fue otra mejora increíblemente significativa.

```
const EventDetailForm = () => {
    const renderContent = () => {
                    </div>
                );
            case 'projectDetails':
                return selectedProject && event ? (
                    <ProjectDetailView event={event} project={selectedProject} goBack={handleBackToList} />
                ) : (
                    <div>Selecciona un proyecto para ver detalles.</div>
                );
            case 'judges':
                return event ? <JudgesManagement event={event} updateEvent={setEvent} /> : null;
            case 'evaluationAspects':
                return event ? <EvaluationAspectsManagement evaluationAspects={evaluationAspects} setEvaluationAspects={setEvaluationAspects} event={event} /> : null;
            case 'organizers':
                return event ? <OrganizerManagement event={event} updateEvent={setEvent} /> : null;
            case 'scores':
                return <ScoresDisplay projects={projectScores} isOrganizer={isOrganizer} isJudge={isJudge} onViewComments={handleViewComments} />;
            case 'comments':
                return viewingComments ? (
                    <ProjectComments project={viewingComments} goBack={handleCloseComments} />
                ) : (
                    <div>No se ha seleccionado ningún proyecto.</div>
                );
            default:
                return <div>Selecciona una opción</div>;
        }
    };
```

Figura 4.7: Fragmento de código para cambiar de ventana

Como se puede ver en la figura 4.7, en una misma ventana se renderiza un contenido diferente dependiendo del valor de una variable, lo que evita que la aplicación se esté recargando constantemente.

Diseño y Estilos

Otra parte que resultó problemática fue el tema del diseño y los estilos de la aplicación. Finalmente, se decidió utilizar una librería como TailwindCSS[2] en lugar de estilos en archivos .css, como se hace generalmente. Esto se debe a que esta librería facilita enormemente la aplicación de estilos y proporciona estilos preestablecidos que simplifican mucho el trabajo. El problema es que se acaba teniendo un código más largo al no utilizar archivos separados para los estilos. Sin embargo, en nuestro caso, esto no es algo preocupante, ya que al no ser una aplicación enorme en cuanto a diferentes componentes y demás, las ventajas que nos ofrece esta librería son mucho mayores al hecho de que el código se alargue un poco.

```
{editMode && (
    <div className="fixed inset-0 bg-black bg-opacity-50 flex justify-center items-center">
        <div className="bg-white px-10 pb-10 shadow-lg rounded-lg max-w-3xl w-full mx-4 overflow-auto" style={{ maxHeight: '80vh' }}>
            <form onSubmit={handleSubmit} className="space-y-4">
                <div className="sticky top-0 left-0 right-0 bg-white px-4 py-2 flex justify-between items-center border-b z-50">
                    <h2 className="text-xl font-bold">Edit Event</h2>
                    <div className="flex space-x-2">
                        <button onClick={() => setEditMode(false)} className="p-2 rounded-full bg-gray-300 hover:bg-gray-400">
                            <svg className="w-6 h-6 text-gray-600" fill="none" stroke="currentColor" viewBox="0 0 24 24" xmlns="http:/
                        </button>
                        <button type="submit" className="p-2 rounded-full bg-blue-500 hover:bg-blue-600">
                            <svg className="w-6 h-6 text-white" fill="none" stroke="currentColor" viewBox="0 0 24 24" xmlns="http://ww
                        </button>
```

Figura 4.8: Fragmento de código utilizando TailwindCSS

En la figura 4.8 puede observarse el uso de TailwindCSS para aplicar estilos a los diferentes componentes de la aplicación, añadiendolos dentro del campo çlassName".

[2] https://tailwindcss.com/

Refactorings Significativos

Un refactoring significativo fue la reestructuración del manejo de estados globales y la mejora de la navegación en React. La implementación de Zustand y la suscripción a canales de Supabase transformaron la forma en que la aplicación manejaba datos dinámicos, mejorando significativamente la experiencia del usuario. Además, la transición hacia una verdadera SPA redujo los tiempos de carga y mejoró la fluidez de la aplicación.

Estos cambios fueron fundamentales para garantizar que la aplicación Votalo ofreciera una experiencia de usuario óptima, segura y eficiente.

4.4.2. Backend

En cuanto a la programación del backend, aunque estaba familiarizado con C# y .NET, era la primera vez que programaba una API REST y también la primera vez que desarrollaba una aplicación separada en frontend y backend. Aunque al principio me resultó un poco confuso, gracias a los conocimientos adquiridos durante la carrera pude adaptarme rápidamente.

Primero, se creó una carpeta con las entidades definidas, manteniendo los mismos atributos que en la base de datos.

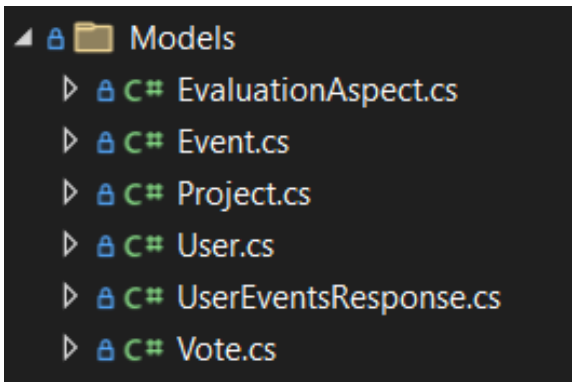

Figura 4.9: Modelos de entidades del backend

En la figura 4.9 se muestra el contenido de la carpeta Models que contiene todas las entidades.

```
12 referencias
public class Vote
{
    0 referencias
    public int id { get; set; }
    2 referencias
    public string text { get; set; }
    2 referencias
    public string judgeId { get; set; }
    3 referencias
    public string projectId { get; set; }
    4 referencias
    public string eventId { get; set; }
    4 referencias
    public int evaluationAspectId { get; set; }
    2 referencias
    public float value { get; set; }

}
```

Figura 4.10: Definición de entidades en el backend

En la figura 4.10 se puede ver el contenido de la clase Vote.

Para la conexión a la base de datos y el consumo de datos de esta, se implementó el patrón Repository, el cual facilita enormemente el trabajo y separa la lógica de la aplicación, haciendo que futuros refactorings o nuevas características sean mucho más sencillas de implementar.

Por un lado, tenemos el contexto (context), que es donde se establece la conexión con la base de datos y se definen las entidades. También tenemos el código del patrón repository, el cual define todas las operaciones a realizar con la base de datos, como la inserción, actualización, eliminación y obtención de datos (getters).

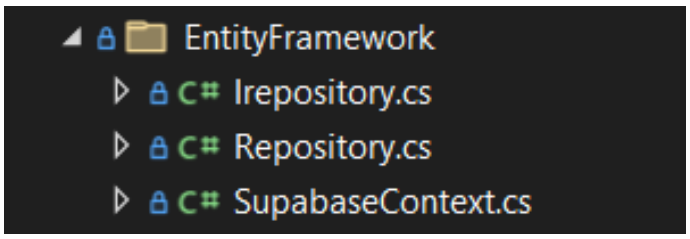

Figura 4.11: Código del patrón repository

En la figura 4.11 se muestra el contenido de la carpeta EntityFrameword que contiene todas las clases necesarias para la implementación del patrón Repository.

```
6 referencias
public class Repository : IRepository
{
    private readonly DbContext _dbContext;

    5 referencias
    public Repository(DbContext dbContext)
    {
        _dbContext = dbContext;
    }

    3 referencias
    public async Task AddAsync<T>(T entity) where T : class
    {
        await _dbContext.Set<T>().AddAsync(entity);
    }

    3 referencias
    public async Task<IEnumerable<T>> GetAllAsync<T>() where T : class
    {
        return await _dbContext.Set<T>().ToListAsync();
    }

    7 referencias
    public async Task<T> GetByIdAsync<T>(object id) where T : class
```

Figura 4.12: Carpeta EntityFramework

En la figura 4.12 se muestra una parte de la implementación del patrón Repository.

Por otro lado, tenemos el Servicio, que contiene una entidad del repository. En este se declaran todas las funciones que necesitan acceder a la base de datos a través de este patrón.

Se decidió que un único servicio (service) contendría las funciones para todas las entidades, en vez de hacer un servicio por cada entidad. Aunque esto puede ser un poco más complejo de implementar, facilita el trabajo futuro al no tener que estar trabajando con tantas entidades diferentes.

A continuación, en la figura 4.13 se muestra una función implementada en el servicio, la cual añade o actualiza un proyecto:

```
2 referencias
public async Task AddOrUpdateProjectAsync(Models.Project project)
{
    using (var context = new SupabaseContext())
    {
        var existingProject =  context.Project.FirstOrDefault(p => p.id == project.id);

        if (existingProject != null)
        {
            context.Entry(existingProject).CurrentValues.SetValues(project);
        }
        else
        {
            context.Project.Add(project);
        }

        await context.SaveChangesAsync();
    }
}
```

Figura 4.13: Función para añadir o actualizar un proyecto en el servicio

Finalmente, tenemos los Controladores (Controllers), que son los endpoints de nuestro backend, y se crea uno por cada entidad.

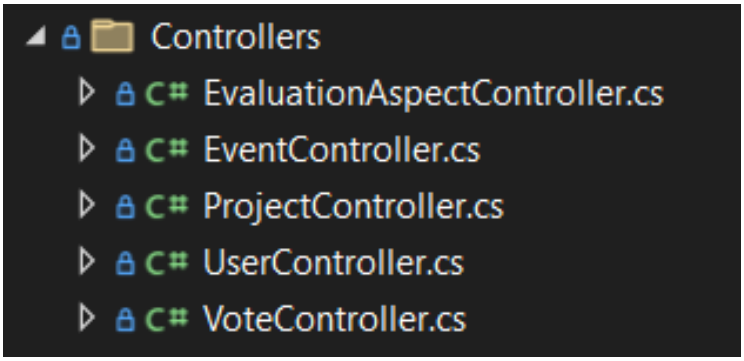

Figura 4.14: Controlador de entidad en el backend

En cada uno de estos controladores se crean diferentes rutas a las cuales el frontend podrá realizar peticiones POST, PUT, GET y DELETE. Aquí se maneja la lógica de lo que se realizará en cada ruta.

Cada controlador cuenta con una entidad del servicio mencionado anteriormente, de manera que cada vez que necesita acceder a la base de datos, lo hace a través de este.

A continuación, en la figura 4.15 se muestra una parte del EventController, el cual define lo que hará la ruta `/api/Event/user-events/userId` al ser llamada con el método GET. En este caso, es un método complejo diseñado para devolver una lista con todos los eventos en los cuales participa un usuario. Como puede verse, todo el acceso a la base de datos se hace mediante el objeto servicio.

```
[Route("api/[controller]")]
[ApiController]
0 referencias
public class EventController : ControllerBase
{
    VotaloServices _service = new VotaloServices(new Repository(new SupabaseContext()));

    // GET: api/Event/user-events/{userId}
    [HttpGet("user-events/{userId}")]
    0 referencias
    public async Task<ActionResult<UserEventsResponse>> GetUserEvents(string userId)
    {
        try
        {
            var organized = await _service.GetEventsByOrganizerIdAsync(userId);
            var judged = await _service.GetEventsByJudgeIdAsync(userId);
            var participated = await _service.GetEventsByParticipantIdAsync(userId);

            var response = new UserEventsResponse
            {
                OrganizedEvents = organized,
                JudgedEvents = judged,
                ParticipatedEvents = participated
            };

            return Ok(response);
        }
        catch (Exception ex)
        {
            return StatusCode(500, $"Internal server error: {ex.Message}");
        }
    }
```

Figura 4.15: Definición de la ruta para obtener eventos de un usuario

Una herramienta que fue de gran ayuda para el desarrollo del backend fue Swagger[3], la cual agilizó mucho el proceso, permitiendo probar rápidamente todos los endpoints de mi API para comprobar si funcionaban correctamente, sin necesidad de generar código en el frontend con el único objetivo de comprobar si mi backend funcionaba correctamente. En la figura 4.16 se puede ver una parte de la interfaz de Swagger.

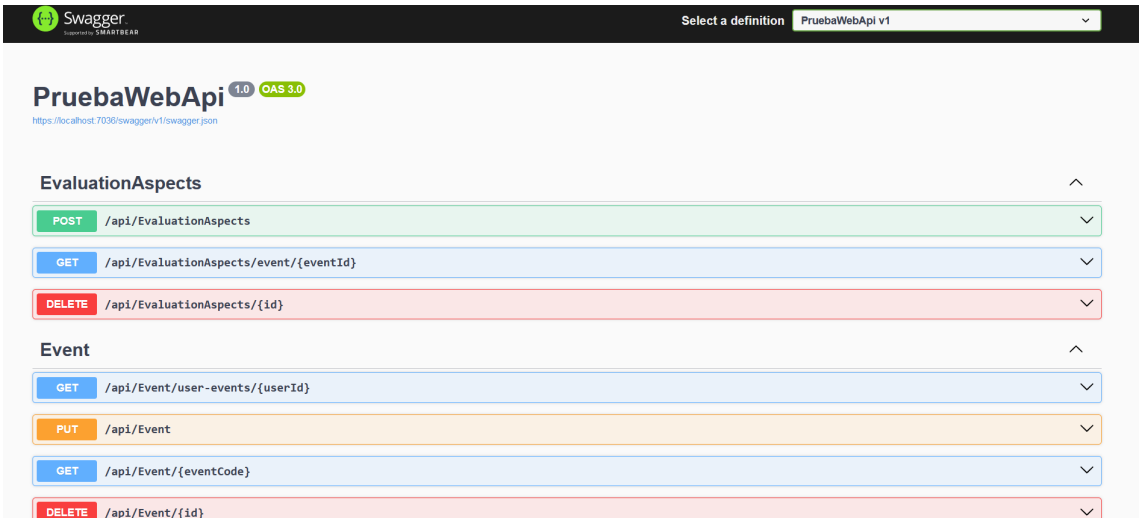

Figura 4.16: Interfaz de Swagger para probar endpoints

4.5 Pruebas

En esta sección se documentan las pruebas aplicadas a la aplicación Votalo para asegurar su correcto funcionamiento y calidad. Se describen las pruebas de aceptación, pruebas de integración y pruebas unitarias realizadas durante el desarrollo del proyecto.

4.5.1. Pruebas de Aceptación

Las pruebas de aceptación se llevaron a cabo para verificar que la aplicación cumpliera con los requisitos funcionales y no funcionales definidos en la fase de planificación. Estas pruebas fueron validadas por el propio desarrollador, y los experimentos realizados con los usuarios finales sirvieron para comprobar que se habían validado correctamente.

Objetivo

Verificar que la aplicación cumpla con los requisitos y expectativas del usuario final.

Procedimiento

1. **Selección de Casos de Uso:** Se seleccionaron casos de uso representativos que abarcan las principales funcionalidades de la aplicación, tales como la creación de eventos, la votación de proyectos y la gestión de comentarios.

2. **Definición de Criterios de Aceptación:** Se definieron criterios claros y específicos para cada caso de uso, como la capacidad de crear un evento con éxito, la posibi-

[3] https://swagger.io/

lidad de que los jueces voten y comenten, y la visualización en tiempo real de los resultados de las votaciones.

3. **Validación por el Desarrollador:** El desarrollador realizó las pruebas de aceptación, validando cada caso de uso según los criterios definidos.

4. **Comprobación mediante Experimentos:** Los usuarios finales probaron la aplicación utilizando escenarios predefinidos, y sus experiencias sirvieron para comprobar que las funcionalidades validadas funcionaban correctamente en un entorno real.

Ejemplos de Pruebas de Aceptación

- **Caso de Uso:** Creación de un Evento
  - **Criterios de Aceptación:** El usuario debe poder crear un evento proporcionando un título, descripción, fecha de inicio, fecha de finalización, imagen y localización.
  - **Resultado Esperado:** El evento se crea correctamente y aparece en la lista de eventos del usuario.
  - **Resultado Obtenido:** El evento se creó correctamente con todos los datos añadidos.
- **Caso de Uso:** Votación de Proyectos
  - **Criterios de Aceptación:** Los jueces deben poder votar y dejar comentarios en los proyectos.
  - **Resultado Esperado:** Las votaciones y comentarios se registran correctamente y se reflejan en tiempo real.
  - **Resultado Obtenido:** La funcionalidad de votación y comentarios funcionó correctamente.

A continuación se presenta la lista de las Pruebas de Aceptación aplicadas a la aplicación Votalo:

- Verificar que el usuario pueda añadir fotos a un evento sin errores.
- Asegurar que las fotos añadidas se muestren correctamente en el evento.
- Confirmar que el usuario pueda modificar los detalles de un evento existente.
- Asegurar que el usuario pueda eliminar un evento y que este no aparezca en la lista de eventos.
- Verificar que el usuario pueda editar los detalles de un proyecto.
- Asegurar que los cambios en los proyectos se guarden correctamente.
- Confirmar que el usuario pueda visualizar los resultados de las votaciones de sus proyectos.
- Asegurar que los resultados se muestren de manera clara y comprensible.
- Verificar que la interfaz de usuario esté bien diseñada y sea intuitiva.

- Asegurar que los estilos finales se apliquen correctamente en toda la aplicación.
- Verificar que el usuario pueda crear un nuevo evento con todos los detalles necesarios.
- Asegurar que el evento creado se añada correctamente a la lista de eventos.
- Confirmar que el usuario pueda añadir, modificar y eliminar jueces y organizadores de un evento.
- Verificar que los cambios en los jueces y organizadores se reflejen correctamente en el evento.
- Asegurar que los usuarios puedan acceder a los eventos a los que están invitados.
- Verificar que se pueda acceder a la información completa del evento.
- Confirmar que el usuario pueda añadir nuevos proyectos a un evento.
- Verificar que el usuario pueda eliminar proyectos y que estos no aparezcan en la lista de proyectos.
- Verificar que el usuario pueda definir y eliminar aspectos específicos a evaluar en los proyectos.
- Asegurar que los cambios en los aspectos a evaluar se guarden correctamente.
- Confirmar que los usuarios puedan votar en los proyectos y dejar comentarios.
- Verificar que los votos y comentarios se registren y se muestren correctamente.
- Asegurar que los usuarios puedan ver los resultados generales de las votaciones.
- Verificar que los resultados se presenten de manera clara y precisa.
- Confirmar que los usuarios puedan iniciar sesión y cerrar sesión utilizando Google.
- Verificar que el proceso de autenticación con Google funcione correctamente y sin errores.

4.5.2. Pruebas de Integración

Las pruebas de integración se realizaron para asegurar que los diferentes módulos y componentes de la aplicación funcionaran correctamente cuando se integraran en conjunto. Estas pruebas son cruciales para detectar problemas de compatibilidad y comunicación entre módulos.

Objetivo

Asegurar que los módulos integrados funcionen correctamente como una unidad completa.

Procedimiento

1. **Identificación de Módulos Críticos:** Se identificaron los módulos críticos que necesitaban ser integrados, como el backend con el frontend y la integración de servicios externos como Supabase.
2. **Definición de Casos de Prueba:** Se definieron casos de prueba específicos para las interacciones entre módulos, como la creación de eventos desde el frontend y su almacenamiento en la base de datos a través del backend.
3. **Ejecución de Pruebas:** Se realizaron pruebas para verificar que los módulos se comunicaran correctamente y que las funcionalidades integradas se ejecutaran sin errores.

Ejemplos de Pruebas de Integración

- **Prueba de Integración:** Creación de un Evento
  - **Descripción:** Verificar que la creación de un evento desde el frontend se almacene correctamente en la base de datos a través del backend.
  - **Procedimiento:** Crear un evento desde la interfaz del usuario y comprobar en la base de datos si se ha almacenado correctamente.
  - **Resultado Esperado:** El evento se almacena correctamente en la base de datos.
  - **Resultado Obtenido:** El evento se almacenó correctamente en la base de datos.
- **Prueba de Integración:** Actualización en Tiempo Real
  - **Descripción:** Verificar que las actualizaciones de votaciones y comentarios se reflejen en tiempo real en todos los clientes conectados.
  - **Procedimiento:** Realizar una votación en un proyecto desde un dispositivo y comprobar en otro dispositivo si la actualización se refleja en tiempo real.
  - **Resultado Esperado:** La votación se actualiza en tiempo real en todos los dispositivos conectados.
  - **Resultado Obtenido:** Las actualizaciones en tiempo real funcionaron correctamente después de ajustar la configuración de suscripciones de Supabase.

A continuación se presenta una lista de las Pruebas de Integración aplicadas a la aplicación Votalo:

- Verificar la integración entre el frontend y el backend para la creación de eventos.
- Comprobar que la actualizacion de los datos se hace en tiempo real.
- Asegurar que las modificaciones y eliminaciones de eventos se reflejen correctamente entre el frontend y el backend.
- Comprobar que las fotos añadidas a los eventos se almacenan y se recuperan correctamente desde el backend.
- Confirmar que los detalles de los proyectos se pueden editar y guardar correctamente a través del frontend y que se actualizan en el backend.
- Asegurar que los resultados de las votaciones de proyectos se calculan y se muestran correctamente en el frontend.

- Comprobar la autenticación y autorización de usuarios mediante Google y su integración con el backend.
- Asegurar que la asignación, modificación y eliminación de jueces y organizadores se reflejan correctamente en el sistema.
- Verificar que los usuarios pueden acceder a los eventos y que toda la información relevante se carga correctamente desde el backend.
- Confirmar que la adición y eliminación de proyectos se integran correctamente con la base de datos a través del backend.
- Asegurar que los aspectos a evaluar en los proyectos se pueden añadir y eliminar correctamente y que se reflejan en el sistema.
- Verificar que las votaciones y comentarios de los usuarios se registran correctamente en la base de datos y se muestran en el frontend.
- Asegurar que los resultados generales de las votaciones se calculan correctamente y se presentan en el frontend.

4.5.3. Pruebas Unitarias

Las pruebas unitarias se realizaron para validar que cada método de clases del código funcionara correctamente. Estas pruebas son fundamentales para asegurar la calidad del código y facilitar el mantenimiento futuro.

Objetivo

Verificar que cada unidad de código funcione correctamente de manera aislada.

Procedimiento

1. **Selección de Unidades de Prueba:** Se seleccionaron funciones y métodos críticos para ser probados de manera unitaria.
2. **Escritura de Casos de Prueba:** Se escribieron casos de prueba específicos para cada unidad utilizando Jest como herramienta de testing.
3. **Ejecución de Pruebas:** Se ejecutaron las pruebas unitarias automáticamente mediante un pipeline de integración continua.
4. **Análisis de Resultados:** Se analizaron los resultados de las pruebas y se corrigieron los errores detectados.

Ejemplos de Pruebas Unitarias

- **Prueba Unitaria:** Función de Autenticación
  - **Descripción:** Verificar que la función de autenticación valide correctamente las credenciales del usuario.
  - **Procedimiento:** Probar la función con diferentes combinaciones de credenciales válidas e inválidas.

- **Resultado Esperado:** La función debe aceptar credenciales válidas y rechazar las inválidas.

- **Resultado Obtenido:** La función de autenticación pasó todas las pruebas, asegurando que solo los usuarios con credenciales válidas pueden registrarse.

A continuación, en la figura 4.17 se puede ver una parte del código implementado con Jest[4], tecnología usada que sirve para comprobar que tanto la función para validar el email como la función para validar la contraseña funcionan correctamente. Se definen diferentes casos para cada función, especificando cuál es el valor esperado y comparándolo con el resultado obtenido.

```
// src/views/EmailAuth.test.tsx
import { validateEmail, validatePassword } from './EmailAuth';


describe('validateEmail', () => {
    test('returns true for a valid email', () => {
        expect(validateEmail('test@example.com')).toBe(true);
    });

    test('returns false for an invalid email', () => {
        expect(validateEmail('invalid-email')).toBe(false);
    });

    test('returns false for an email without domain', () => {
        expect(validateEmail('test@')).toBe(false);
    });

    test('returns false for an email without "@" symbol', () => {
        expect(validateEmail('testexample.com')).toBe(false);
    });
});

describe('validatePassword', () => {
    test('returns true for a valid password', () => {
        expect(validatePassword('password1')).toBe(true);
    });

    test('returns false for a password shorter than 8 characters', () => {
        expect(validatePassword('pass1')).toBe(false);
    });

    test('returns false for a password without numbers', () => {
        expect(validatePassword('password')).toBe(false);
    });
```

Figura 4.17: Código de pruebas unitarias para validar email y contraseña

Tras ejecutar los tests, se puede comprobar que el resultado es exitoso, como muestra la figura 4.18:

[4] https://jestjs.io/es-ES/

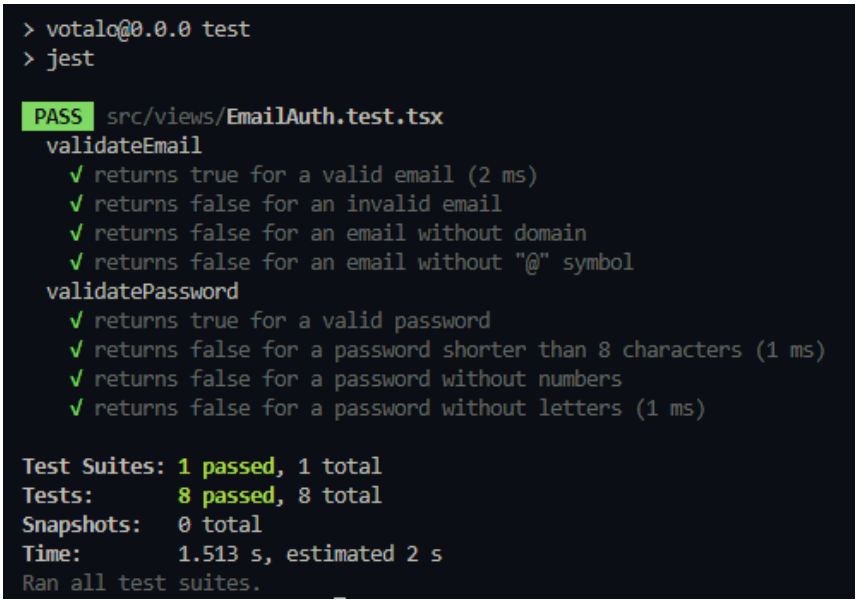

Figura 4.18: Resultados de las pruebas unitarias

- **Prueba Unitaria:** Cálculo de Porcentajes de Evaluación

  - **Descripción:** Verificar que la función `calculatePercentages` calcule correctamente los porcentajes de los aspectos de evaluación.

  - **Procedimiento:** Probar la función con diferentes combinaciones de pesos para los aspectos de evaluación.

  - **Resultado Esperado:** La función debe calcular correctamente los porcentajes basados en los pesos proporcionados.

  - **Resultado Obtenido:** La función `calculatePercentages` pasó todas las pruebas, asegurando que los porcentajes se calculan correctamente.

Esta es una parte critica de la aplicación, ya que el mal funcionamento de esta conllevaría fallos en todo el proceso de votación y a la hora de mostrar los resultados.

A continuación, en la figura 4.19 se puede ver una parte del código implementado con Jest que sirve para comprobar que la función `calculatePercentages` funciona correctamente, definiendo diferentes casos para cada uno y comparando el valor esperado con el resultado obtenido.

```
describe('calculatePercentages', () => {
    test('calculates percentages correctly', () => {
        const evaluationAspects: EvaluationAspect[] = [
            { id: 1, name: 'Aspect 1', description: 'Description 1', weight: 30, eventId: "1" },
            { id: 2, name: 'Aspect 2', description: 'Description 2', weight: 70, eventId: "1" }
        ];

        const { percentages, totalWeight } = calculatePercentages(evaluationAspects);

        expect(totalWeight).toBe(100);
        expect(percentages[1]).toBe('30.00');
        expect(percentages[2]).toBe('70.00');
    });

    test('handles zero total weight', () => {
        const evaluationAspects: EvaluationAspect[] = [
            { id: 1, name: 'Aspect 1', description: 'Description 1', weight: 0, eventId: "1" },
            { id: 2, name: 'Aspect 2', description: 'Description 2', weight: 0, eventId: "1" }
        ];

        const { percentages, totalWeight } = calculatePercentages(evaluationAspects);

        expect(totalWeight).toBe(0);
        expect(percentages[1]).toBe('0');
        expect(percentages[2]).toBe('0');
    });
```

Figura 4.19: Código en Jest

Tras ejecutar las pruebas podemos comprobar que el resultado es exitoso, como muestra la figura 4.18:

```
> jest src/views/Ponderation.test.tsx

 PASS  src/views/Ponderation.test.tsx
  calculatePercentages
    √ calculates percentages correctly (3 ms)
    √ handles zero total weight

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        1.449 s, estimated 2 s
Ran all test suites matching /src\\views\\Ponderation.test.tsx/i.
```

Figura 4.20: Resultado del test

- **Prueba Unitaria:** Cálculo de Puntuaciones de Proyectos

  - **Descripción:** Verificar que la función `calculateScores` calcule correctamente las puntuaciones de los proyectos basadas en los votos (valores entre 0 y 10) y las ponderaciones de los aspectos de evaluación.
  - **Procedimiento:** Probar la función con diferentes combinaciones de proyectos, aspectos de evaluación y votos.
  - **Resultado Esperado:** La función debe calcular correctamente las puntuaciones totales de los proyectos y sus aspectos.
  - **Resultado Obtenido:** La función `calculateScores` pasó todas las pruebas, asegurando que las puntuaciones se calculan correctamente.

A continuación, en la figura 4.21, se puede ver una parte del código implementado con Jest que sirve para comprobar que la función `calculateScores` funciona correctamente, definiendo diferentes casos para cada uno y comparando el valor esperado con el resultado obtenido.

Figura 4.21: Código en Jest

Tras correr los test podemos comprobar que el resultado es exitoso, como muestra la figura 4.22:

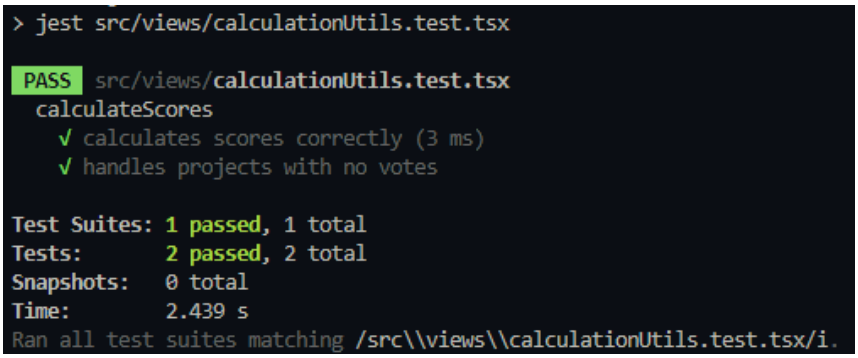

Figura 4.22: Resultado del test

4.5.4. Prueba de Carga

Se realizaron pruebas de carga utilizando la herramienta Jmeter[5], tecnología usada para evaluar la robustez y eficiencia de la aplicación. En estas pruebas, se simularon 980 peticiones en un período de 10 segundos, representando una carga considerablemente alta. Los resultados demostraron la robustez de la aplicación, ya que no se registraron fallos en ninguna de las peticiones. Además, se observó que los tiempos de respuesta de la web fueron excelentes, incluso bajo una carga tan intensa, como se puede observar en la figura 4.23.

[5] https://jmeter.apache.org/

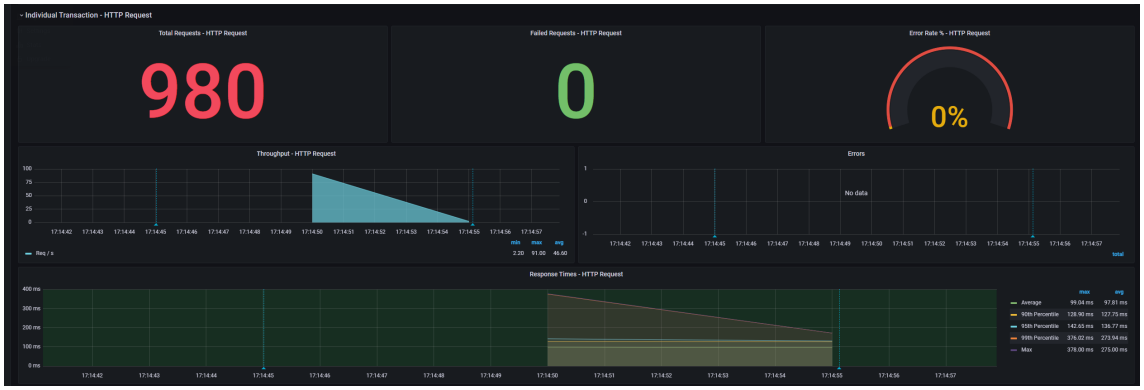

Figura 4.23: Resultados de la prueba de carga, mostrando el rendimiento y los tiempos de respuesta de la aplicación bajo alta demanda.

4.5.5. Conclusión de las Pruebas

La aplicación de estas pruebas permitió asegurar la calidad y fiabilidad de Votalo. Las pruebas de aceptación validadas por el desarrollador, con la comprobación adicional de los experimentos, aseguraron que la aplicación cumpliera con las expectativas del usuario final. Las pruebas de integración garantizaron que los módulos funcionaran correctamente juntos, y las pruebas unitarias aseguraron la calidad de cada componente individual. Gracias a estas pruebas, se pudo identificar y resolver problemas de manera temprana, mejorando la experiencia final del usuario y la robustez

CAPÍTULO 5
Descripción de la aplicación

Esta guía está dirigida a los diferentes tipos de usuarios de la aplicación Votalo: organizadores de eventos, jueces y participantes. Cada tipo de usuario tiene diferentes responsabilidades y accesos dentro de la aplicación, y a continuación se detalla el uso de la aplicación para cada uno de ellos.

5.0.1. Organizadores de Eventos

Los organizadores de eventos son los usuarios responsables de la creación y gestión de los eventos dentro de Votalo. A continuación, se detallan los pasos que deben seguir:

1. **Crear un Evento:** Los organizadores deben proporcionar los siguientes datos para crear un evento:
   - Título del evento
   - Descripción del evento
   - Localización del evento
   - Fecha de inicio
   - Fecha de fin
   - Imagen del evento
2. **Acceder a sus Eventos:** Una vez creado el evento, los organizadores pueden acceder a sus eventos desde el panel principal y seleccionar el evento creado.
3. **Gestionar el Evento:** Dentro del evento, los organizadores pueden:
   - **Añadir más Organizadores:** Otros organizadores pueden ser añadidos al evento, quienes también tendrán la capacidad de editar todos los detalles del evento.
   - **Añadir Jueces:** Los jueces son responsables de votar y comentar sobre cada aspecto a evaluar de los proyectos.
   - **Añadir Aspectos a Evaluar:** Los organizadores deben definir los aspectos a evaluar, proporcionando una descripción y la ponderación de cada uno.
   - **Crear Proyectos:** Se pueden crear diferentes proyectos, añadiendo el título, la descripción y la lista de participantes de cada proyecto.
   - **Modificar o Eliminar Elementos:** Si es necesario, los organizadores pueden modificar o eliminar proyectos, eventos, jueces, organizadores y aspectos a evaluar.

4. **Ver Resultados:** A medida que se realizan las votaciones, los organizadores pueden ver los resultados en el panel de puntuaciones.

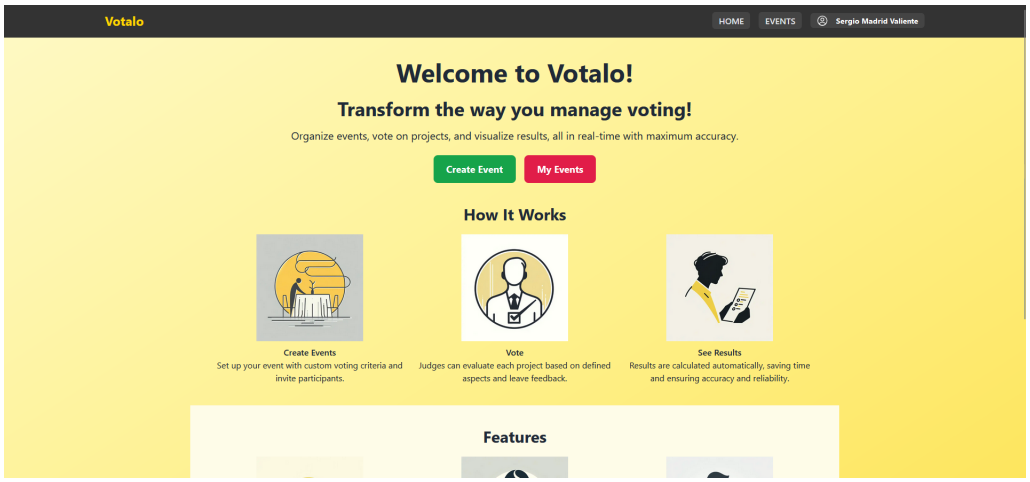

Figura 5.1: Pantalla principal de la aplicación.

En la Figura 5.1, se muestra la pantalla principal, donde los usuarios pueden crear y acceder a sus eventos.

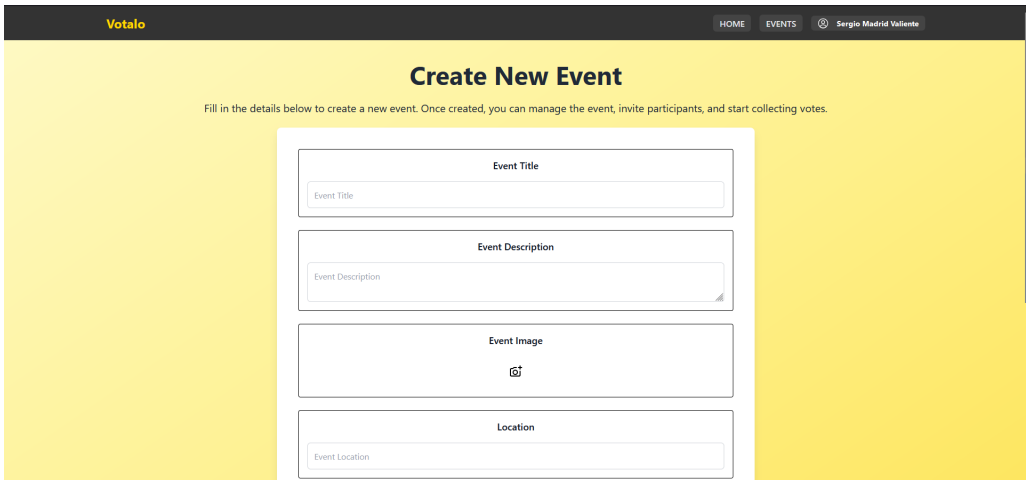

Figura 5.2: Formulario de creación de eventos

En la Figura 5.2, se muestra el formulario de creación de eventos, donde los organizadores proporcionan los detalles del evento.

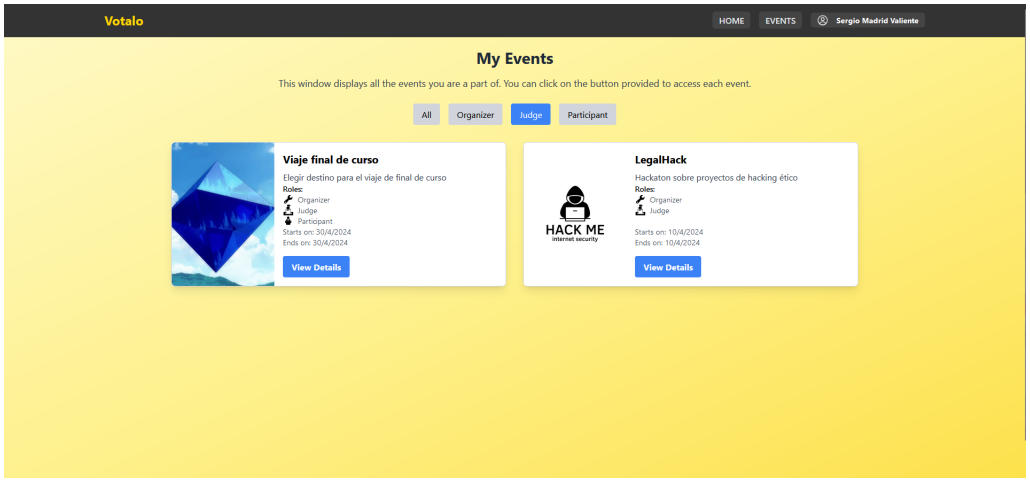

Figura 5.3: Vista de los eventos

En la Figura 5.3, se muestra la vista de los eventos de los que forma parte, permitiendo acceder a ellos, accesible desde el panel principal.

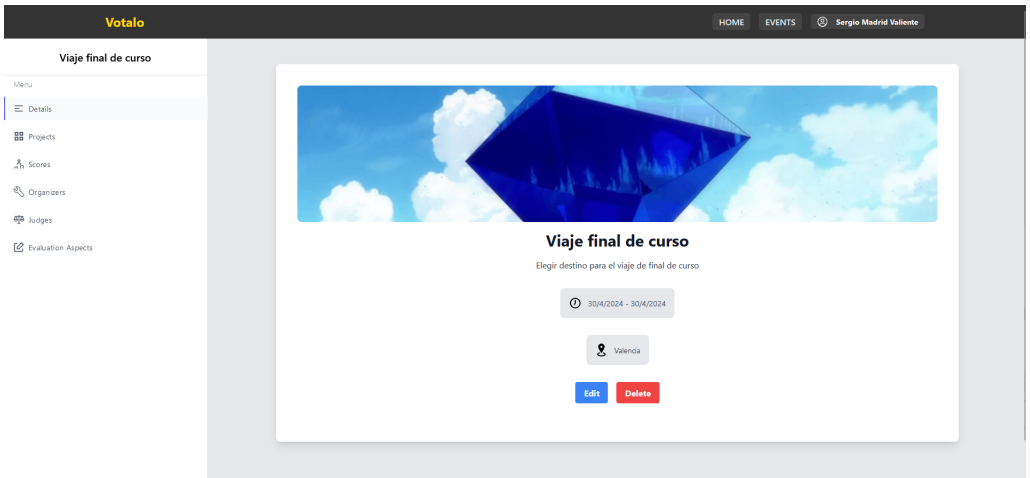

Figura 5.4: Página de detalles del evento

En la Figura 5.4, se muestra la página de detalles del evento, donde los organizadores pueden gestionar el evento y sus componentes.

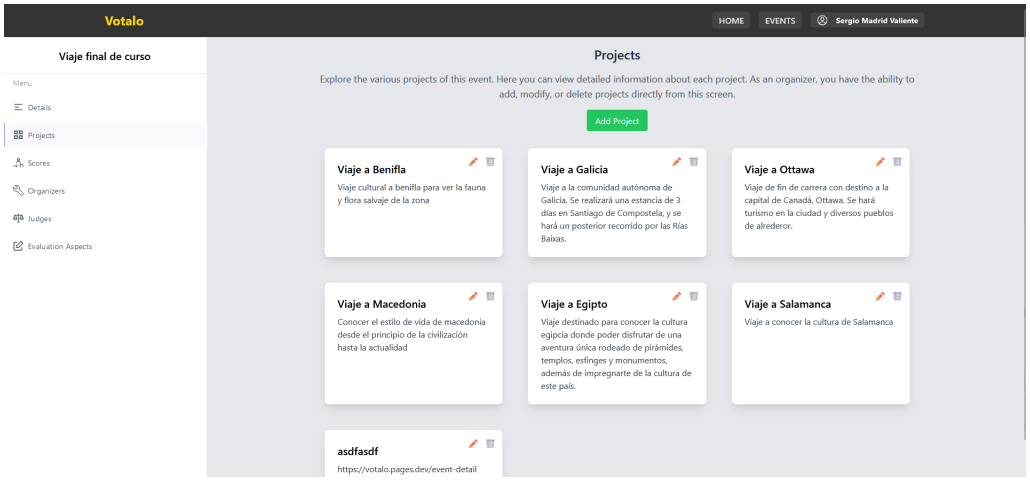

Figura 5.5: Vista de los proyectos

En la Figura 5.5, se muestra la vista de los proyectos dentro de un evento específico.

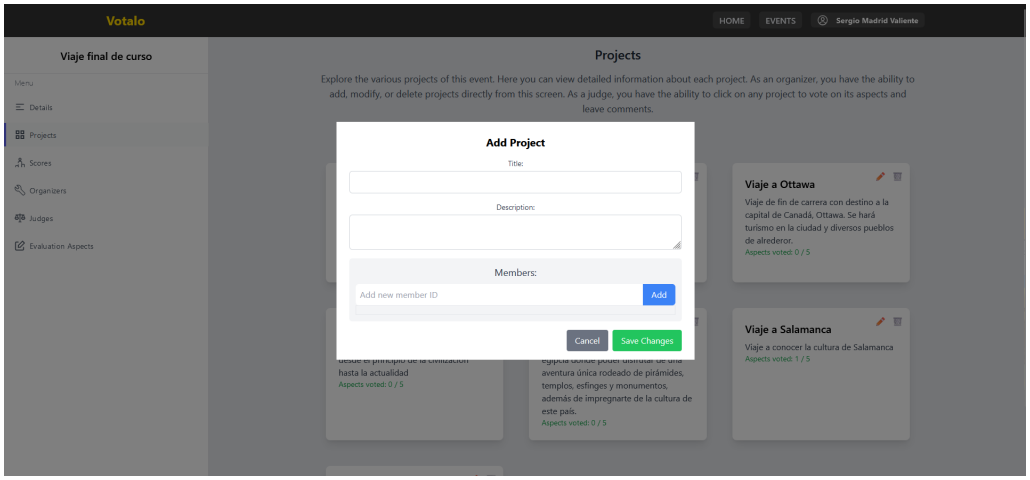

Figura 5.6: Formulario para añadir nuevos proyectos al evento

En la Figura 5.6, se muestra el formulario para añadir nuevos proyectos al evento, con detalles como título, descripción y participantes.

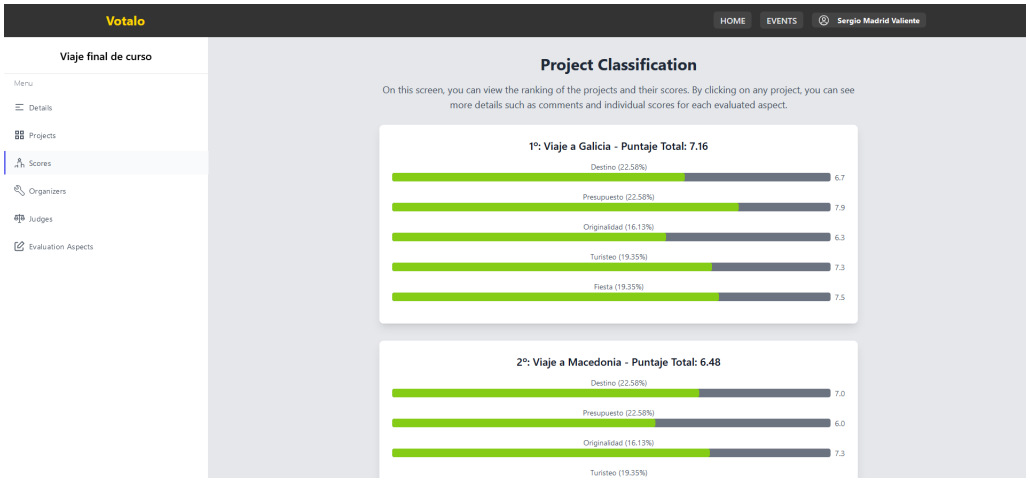

Figura 5.7: Panel de puntuaciones del evento

En la Figura 5.7, se muestra el panel de puntuaciones del evento, donde los organizadores y jueces pueden ver los resultados de las votaciones.

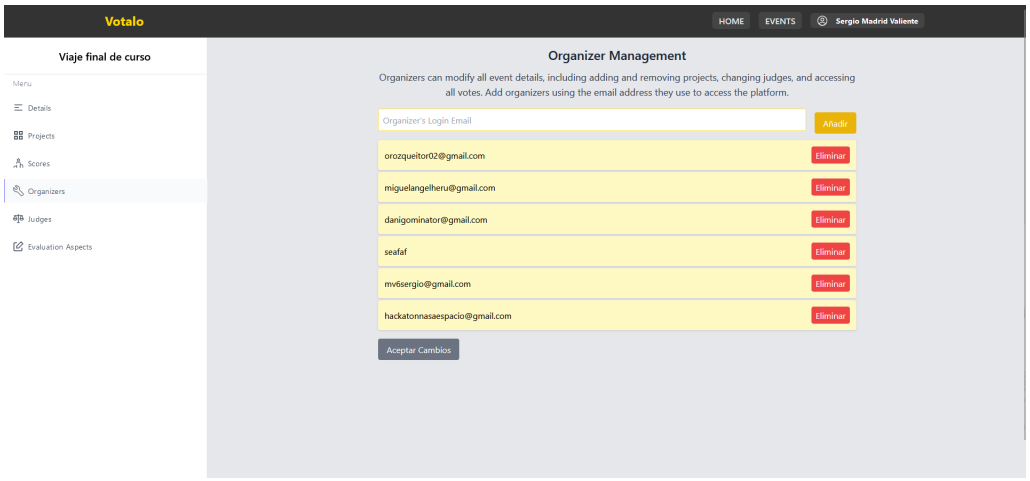

Figura 5.8: Gestión de organizadores del evento

En la Figura 5.8, se muestra la gestión de organizadores del evento, donde se pueden añadir o modificar los organizadores.

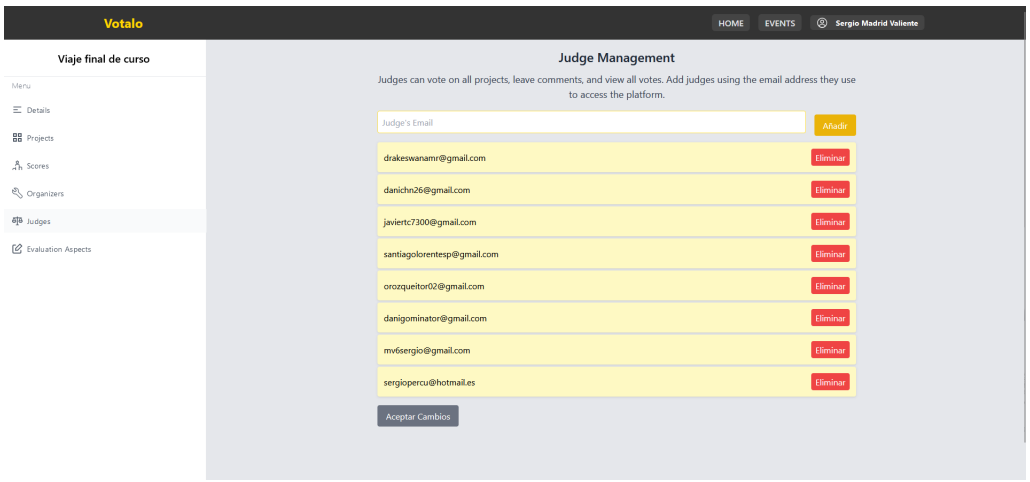

Figura 5.9: Gestión de jueces del evento

En la Figura 5.9, se muestra la gestión de jueces del evento, donde se pueden añadir o modificar los jueces.

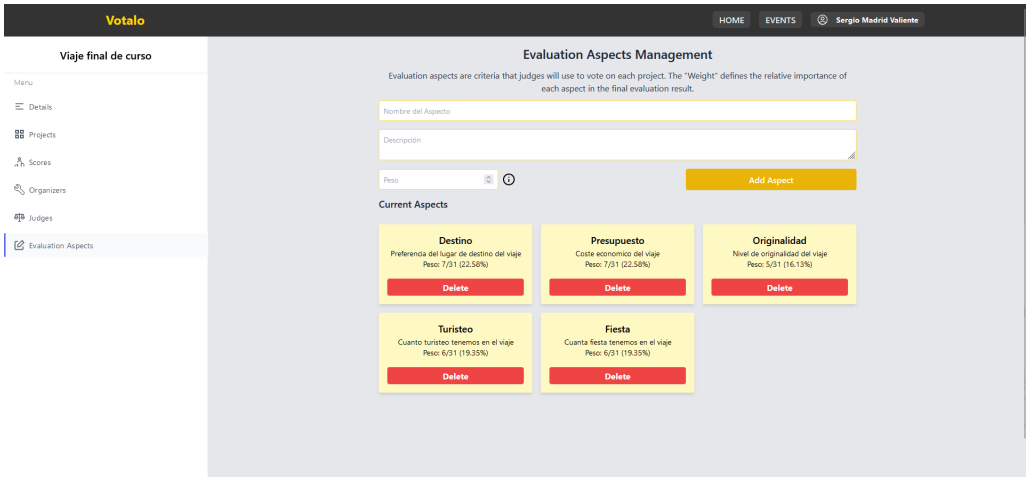

Figura 5.10: Gestión de aspectos a evaluar en el evento

En la Figura 5.10, se muestra la gestión de aspectos a evaluar en el evento, donde los organizadores pueden definir los criterios de evaluación.

5.0.2. Jueces

Los jueces son usuarios encargados de evaluar y comentar sobre los proyectos en función de los aspectos definidos por los organizadores. Los pasos para los jueces son los siguientes:

1. **Acceder a sus Eventos:** Los jueces pueden acceder directamente a sus eventos desde el panel principal.

2. **Seleccionar un Evento:** Dentro de la pantalla de eventos, el juez selecciona el evento en el cual debe votar.

3. **Votar en Proyectos:** Una vez dentro del evento, en la ventana de proyectos, el juez puede hacer clic en cada proyecto, lo que le permitirá:

   - Votar sobre el proyecto en cada uno de los aspectos a evaluar.
   - Comentar sobre los aspectos a evaluar.

4. **Ver Resultados:** Los jueces también pueden ver los resultados de las votaciones de todos los proyectos.

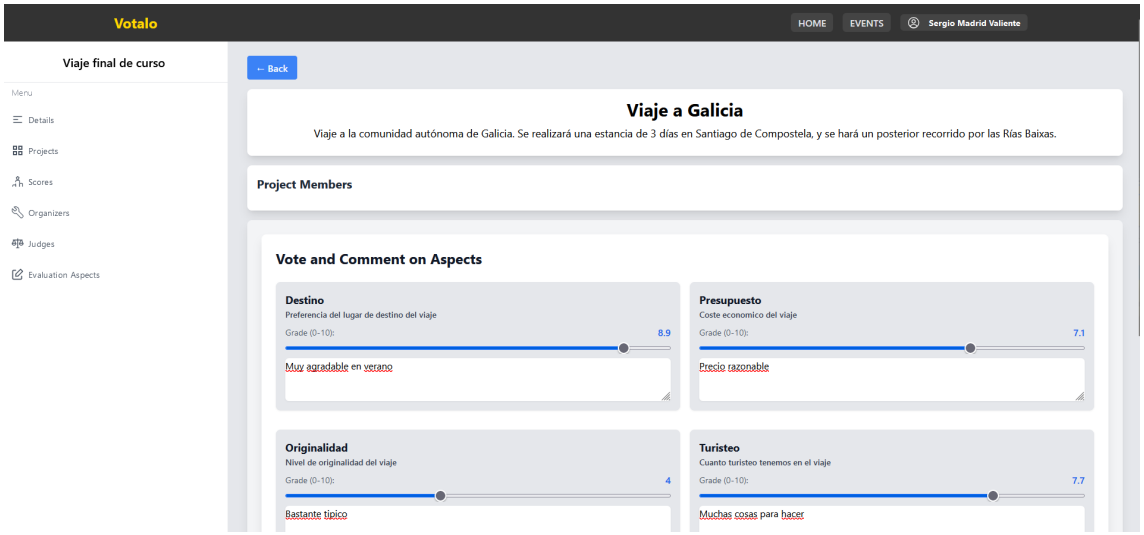

Figura 5.11: Interfaz de votación para jueces

En la Figura 5.11, se muestra la interfaz de votación para jueces, donde pueden evaluar y comentar sobre los proyectos.

5.0.3. Participantes

Los participantes son usuarios que forman parte de uno o más proyectos dentro de un evento. Sus responsabilidades y accesos son los siguientes:

1. **Acceder a sus Eventos:** Los participantes pueden acceder a sus eventos desde el panel principal.

2. **Seleccionar un Evento:** Desde el panel de eventos, el participante selecciona el evento correspondiente.

3. **Ver Detalles del Evento:** Dentro del evento, el participante puede:

   - Ver los detalles del evento.
   - Ver otros proyectos del evento.
   - Ver las calificaciones y comentarios de los aspectos a evaluar únicamente de los proyectos en los que participa.

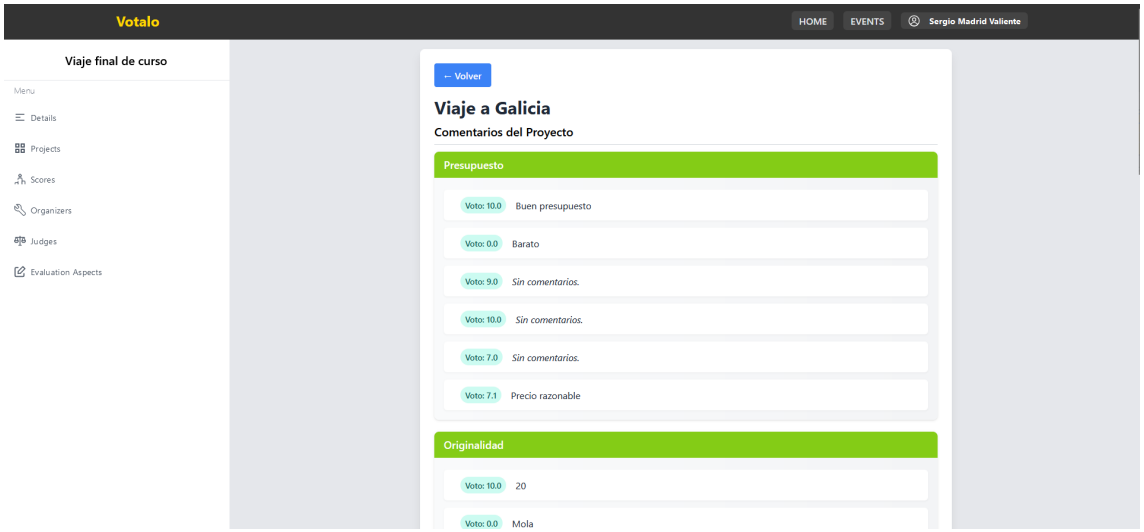

Figura 5.12: Vista de calificaciones y comentarios para participantes.

En la Figura 5.12, se muestra la vista de calificaciones y comentarios para participantes, mostrando las notas junto con los comentarios de sus proyectos.

Esta guía proporciona una visión detallada del uso de la aplicación Votalo para cada tipo de usuario, asegurando que todos comprendan sus roles y responsabilidades dentro de la plataforma.

CAPÍTULO 6
Cronología

Una vez definidos los requisitos, el objetivo era organizar el proyecto para cumplir con las fechas de los dos experimentos.

Se acordó que la primera versión de la aplicación debía estar lista aproximadamente para la última semana de abril para realizar el primer experimento, que consistía en hacer una encuesta a los early adopters después de utilizar la aplicación.

Durante el desarrollo de la carrera se han conocido varias aplicaciones y formas de organizar el trabajo al desarrollar software, pero en nuestro caso se decidió utilizar Trello para organizar las tareas.

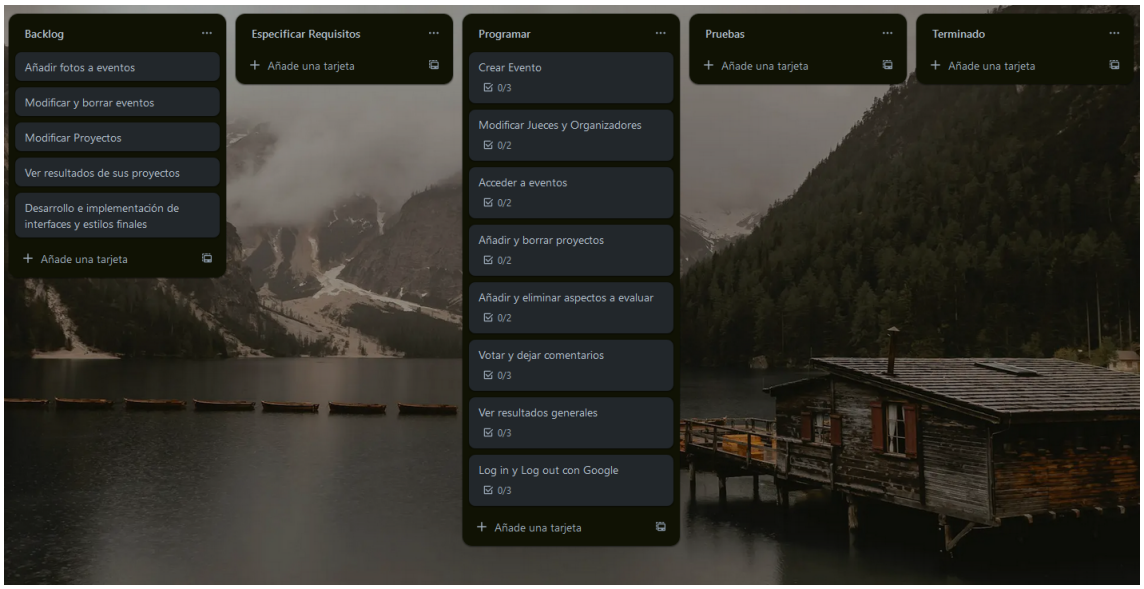

Figura 6.1: Estado del tablero al comenzar el primer sprint

El estado del tablero al comenzar el primer sprint, después de haber especificado los requisitos de las UTs que formarían parte del mismo es el mostrado en la figura 6.1.

El primer sprint se centró en implementar las funcionalidades básicas necesarias para que la aplicación fuera funcional. El MVP resultante de este sprint permitió:

- Crear eventos.
- Añadir jueces y otros organizadores.
- Añadir y borrar proyectos del evento.
- Añadir aspectos a evaluar con sus respectivas ponderaciones.
- Permitir a los jueces votar y dejar comentarios en los proyectos.

- Acceder a los eventos introduciendo su código manualmente.
- Actualizar todos los datos en tiempo real.

A pesar de que este MVP contenía las funcionalidades esenciales, la interfaz de usuario era básica y necesitaba mejoras significativas.

Las estimaciones de tiempo fueron correctas, por lo que todo el trabajo pudo ser terminado a tiempo para el primer experimento. Una vez finalizado el primer sprint, se añadieron algunas UTs extra gracias a los comentarios de los participantes del experimento, para añadir y mejorar ciertas funcionalidades.

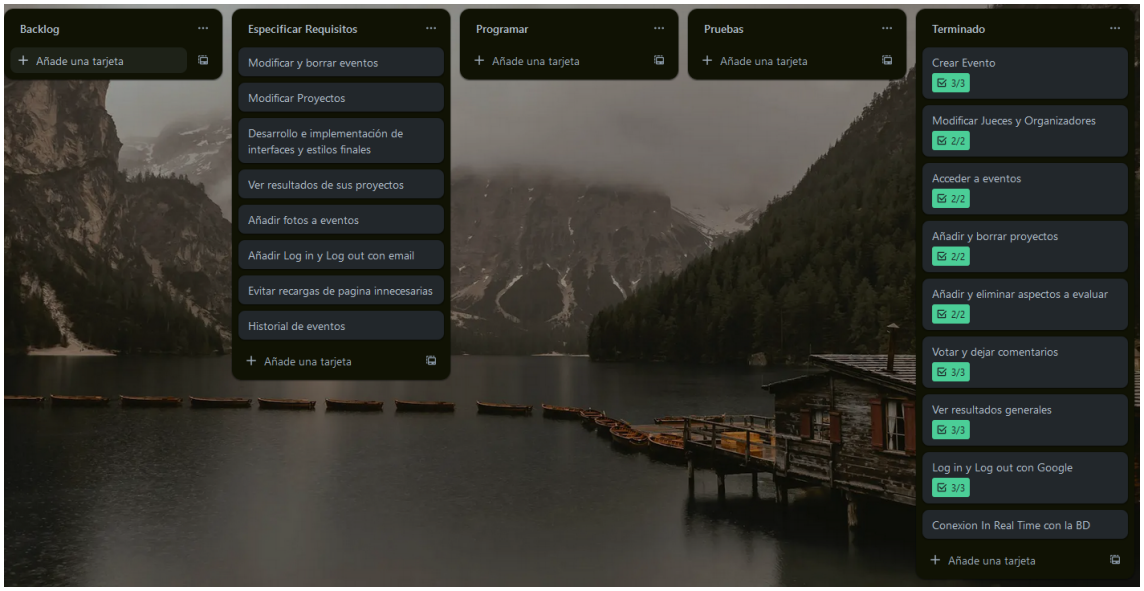

Figura 6.2: Estado del tablero al comenzar el segundo sprint

La figura 6.2 muestra el estado del tablero al empezar el segundo sprint. La UT más costosa sin duda fue la de desarrollo e implementación de interfaces y estética finales, ya que englobaba el desarrollo de los mockups y las mejoras de interfaz de toda la aplicación. Como se ha mencionado varias veces a lo largo del documento, el primer sprint se centró en implementar funcionalidades complejas y no tanto en la estética de la aplicación.

El segundo sprint se centró en agregar nuevas características y mejorar la interfaz de usuario. Las mejoras y adiciones en el MVP del segundo sprint incluyeron:

- Una ventana de historial de eventos, permitiendo acceder a los eventos sin necesidad de introducir el código manualmente.
- Vista de los participantes de cada proyecto.
- Funcionalidades para modificar y borrar eventos, así como modificar proyectos.
- Capacidad para añadir fotos a los eventos.
- Mejora significativa en el diseño y la usabilidad de la aplicación.

Las mejoras en el diseño incluyeron una paleta de colores atractiva y una pantalla principal que explicaba las ventajas de la aplicación y cómo usarla. Se guió mejor al usuario entre las diferentes ventanas, se redujo el tiempo de carga, y se estructuraron y explicaron mejor las funcionalidades dentro de cada ventana. Además, se mejoró la forma de votar y dejar comentarios, haciéndolo mucho más intuitivo, y se solucionaron problemas menores.

Las estimaciones de tiempo para este segundo sprint también fueron correctas y se pudieron finalizar todas las tareas para la fecha prevista, que era la primera semana de junio.

6.0.1. Primer Experimento

Con la primera versión de Votalo lista, se reunió a un conjunto diverso de early adopters para que interactuaran con la aplicación y así poder obtener sus impresiones y sugerencias para futuras mejoras.

La realización de este experimento implicó a un grupo de ocho personas de edades entre 20 y 25 años, y fueron separadas por roles dentro de la dinámica de votación de proyectos: dos de ellos actuaron como organizadores del evento, cuatro asumieron el papel de jueces y los dos restantes tomaron el rol de participantes en un projecto.

Antes de la prueba, elaboré una encuesta que abordaba diferentes aspectos de "Votalo", preguntando sobre la facilidad de uso de la aplicación, la atractividad de sus interfaces y su utilidad percibida, entre otras cuestiones relevantes.

La estructura de la encuesta fue la que muestro en la Figura 6.3:

Pregunta	Escala de importancia				
	En Absoluto	No mucho	NS/NC	Mayormente si	Mucho
¿Qué tan intuitiva encontraste la interfaz de la aplicación para realizar votaciones?	1	2	3	4	5
¿Te resultó sencillo navegar por las distintas secciones de la aplicación?	1	2	3	4	5
¿Encuentras claras y comprensibles las instrucciones para votar y dejar comentarios?	1	2	3	4	5
¿Consideras que la aplicación mejora el proceso de votación en comparación con otros métodos que has utilizado?	1	2	3	4	5
¿Qué tan probable es que recomiendes esta aplicación a un colega o a un organizador de eventos?	1	2	3	4	5
¿Crees que el tiempo de carga y la respuesta de la aplicación fueron adecuados?	1	2	3	4	5
¿Estás satisfecho con la experiencia general de uso de la aplicación?	1	2	3	4	5
¿Funcionaron todas las características de la aplicación como esperabas?	Si o No				
¿Qué características crees que podrían añadirse para mejorar la aplicación?	Respuesta abierta				
¿Tienes alguna sugerencia específica para mejorar la interfaz de usuario?	Respuesta abierta				
¿Experimentaste algún error o problema técnico mientras utilizabas la aplicación?	Respuesta abierta				

Figura 6.3: Estructura de la encuesta

Como ya se ha comentado, el primer grupo de early adopters fue formado por 8 personas jóvenes, de las cuales cinco eran estudiantes de ingeniería informática, uno es estudiante del doctorado de Bioinformatica, y los dos restantes trabajan como profesores.

La idea del experimento era darles la herramienta y los diferentes roles de cada uno, simulando estar en un contexto en el cual se esta intentando proponer un destino para un viaje, y se han estado realizando exposiciones sobre los diferentes destinos, y ver como se desenvolvían con la herramienta sin darles ninguna indicación.

El objetivo de la encuesta era recopilar la mayor información posible sobre los posibles aspectos a mejorar para la siguiente versión de la aplicación. Por eso se añadieron preguntas tanto numéricas como de respuesta abierta.

Figura 6.4: ¿Qué tan intuitiva encontraste la interfaz?

Como era de esperar, la interfaz de la aplicación no fue del todo amigable con el usuario, ya que en este primer MVP, se priorizó el desarrollar funciones mas complejas de la lógica de la aplicación, con el objetivo de afinar las estimaciones de tiempo de trabajo lo antes posible, aplicando la ideología fail-fast.

Figura 6.5: ¿Te resultó sencillo navegar por las distintas secciones?

En este caso la mayoría de las personas coincidieron en que era relativamente fácil navegar entre las diferentes funcionalidades de la aplicación, aunque también hubo gente que no lo vio tan sencillo.

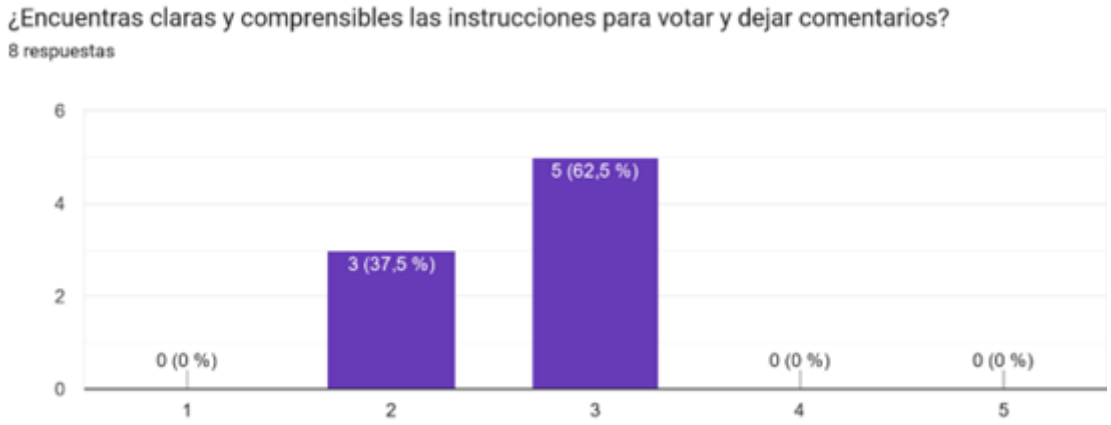

Figura 6.6: ¿Encuentras claras y comprensibles las instrucciones para votar y dejar comentarios?

Al igual que en la primera pregunta, el resultado era esperable, ya que el estudiar una mejor forma de guiar al usuario a la hora de votar esta planteado para hacer en la siguiente parte del desarrollo de la aplicación.

Figura 6.7: ¿Funcionaron todas las características de la aplicación como esperabas?

Aquí podemos ver que mayoritariamente todo funcionó como se esperaba, aunque hubo un error puntual que hizo que algunas personas no estuvieran de acuerdo.

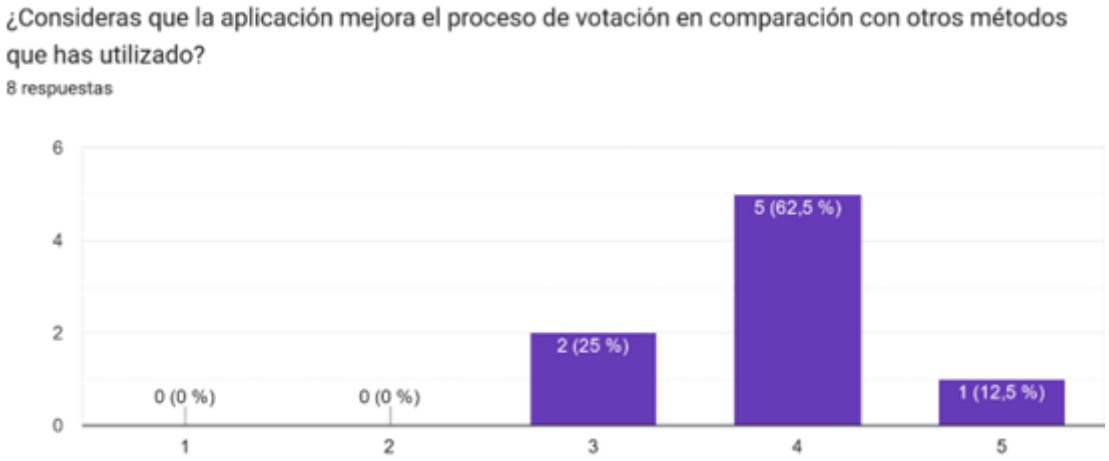

Figura 6.8: ¿Consideras que la aplicación mejora el proceso de votación en comparación con otros métodos?

En estos resultados se ve que la gran mayoría de personas encontró útil el objetivo de la aplicación.

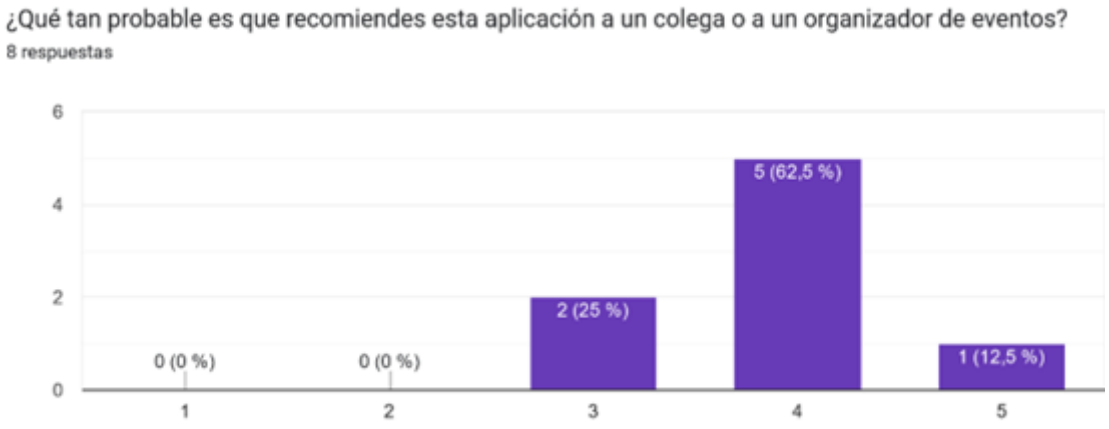

Figura 6.9: ¿Qué tan probable es que recomiendes esta aplicación?

La mayor parte afirmó que se la recomendaría a alguien que necesitara de un servicio similar a este.

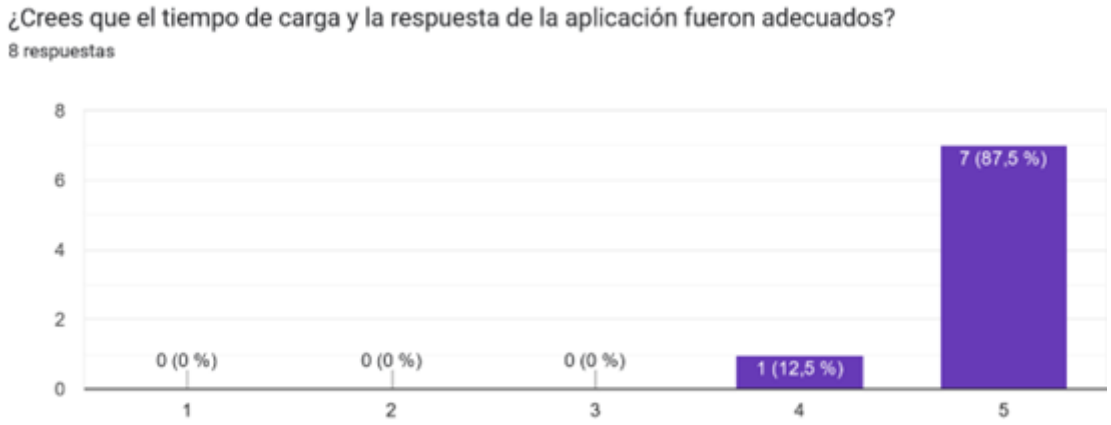

Figura 6.10: ¿Crees que el tiempo de carga y la respuesta de la aplicación fueron adecuados?

En este caso la gente estuvo muy satisfecha con el rendimiento de la web, como era de esperar, ya que entre todo el trabajo realizado este primer sprint, se encuentran muchas tareas enfocadas al buen funcionamiento de la lógica de la aplicación.

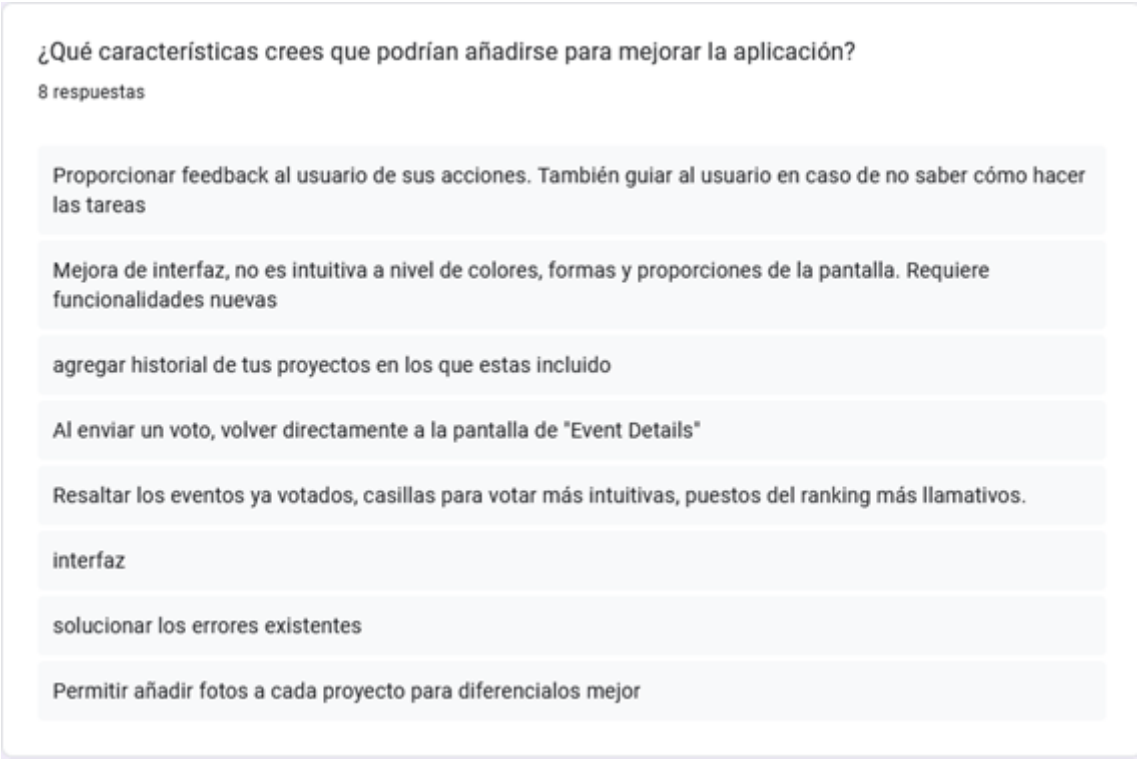

Figura 6.11: ¿Qué características crees que podrían añadirse para mejorar la aplicación?

En esta pregunta pudimos recibir un feedback muy valioso. Entre las sugerencias podemos destacar dos que se repiten más. La primera es la mejora de la interfaz y la forma de guiar al usuario por la aplicación, que como ya hemos dicho varias veces, era de esperar. Y la segunda es la implementación de un historial de eventos, mediante el cual pueda acceder rápidamente a todos los eventos de los que formo parte sin tener que introducir el código cada ver. Esta es una funcionalidad que ya estaba planeada para

añadir en el siguiente sprint, pero al ver que varias personas resaltan su importancia nos hace replantearnos la forma de acceder a los eventos.

Además de estas dos sugerencias, me pareció una gran idea el resaltar los proyectos ya votados, para ayudar a los jueces en su tarea. Era algo que no se había pensado y es increíblemente útil.

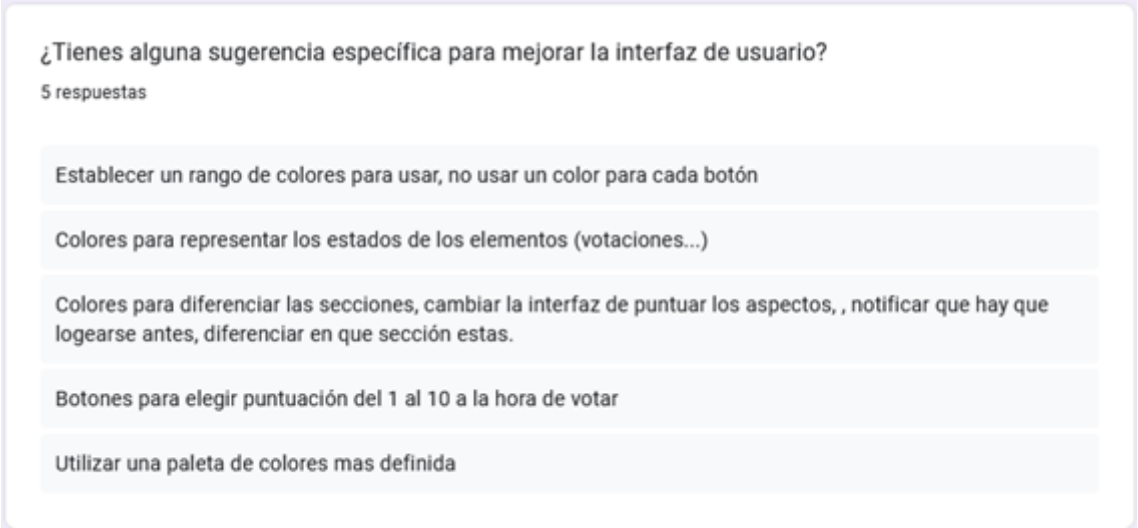

Figura 6.12: ¿Tienes alguna sugerencia específica para mejorar la interfaz de usuario?

Esta pregunta era muy importante, ya que me ayudara a todo el trabajo a realizar en el siguiente sprint. Podemos ver que mucha gente sugiere cambiar la paleta de colores.

Otra propuesta muy interesante es el cambiar como elegir la puntuación a la hora de votar, ya que era algo que estaba poco desarrollado, y tras las sugerencias se ha replanteado el cómo debería ser esta interfaz.

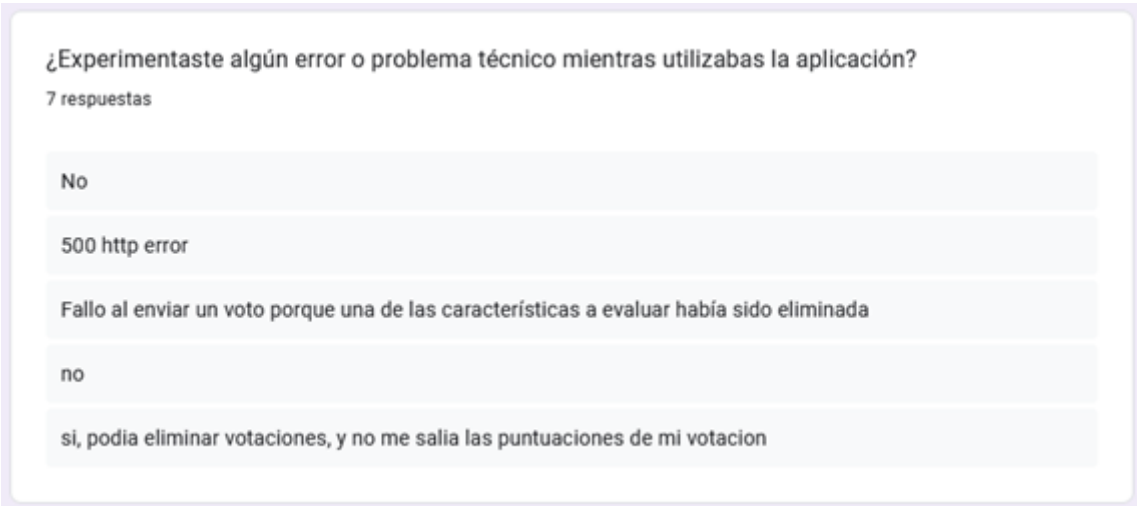

Figura 6.13: ¿Experimentaste algún error o problema técnico mientras utilizabas la aplicación?

Aquí se puede ver que mayoritariamente no hubo errores, quitando los mencionados en la encuesta, que vienen de un uso de la aplicación que no había sido previsto, y que tiene una solución sencilla.

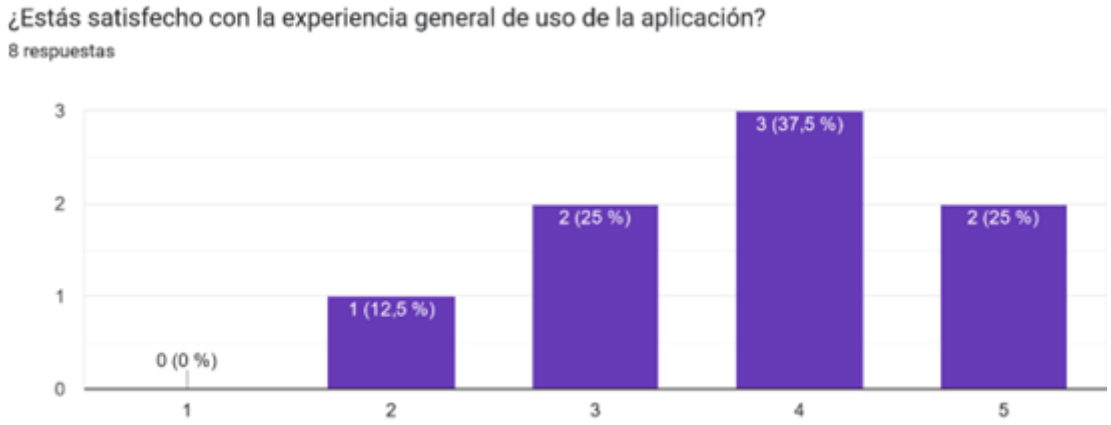

Figura 6.14: ¿Estás satisfecho con la experiencia general de uso de la aplicación?

Finalmente podemos diferencia de opiniones sobre la experiencia de uso, aunque la mayoría fueron positivas, hay varias que no lo son tanto y que se deben a una interfaz de usuario poco intuitiva.

Lo que sacamos de esta encuesta es la gran capacidad de mejora que tiene Votalo, ya que incluso sin haber enfocado el desarrollo de este primer sprint en facilitar la experiencia de uso de un usuario primerizo, los resultados han sido mayoritariamente positivos. Por lo que al enfatizar en los aspectos de mejora de interfaz e implementar alguna funcionalidad necesaria más, Votalo podrá ofrecer un servicio mucho mejor.

6.0.2. Segundo Experimento

Tras completar la segunda versión de Votalo, se utilizó para realizar una actividad con los alumnos de PSW de tercero de carrera. Estos habían realizado unos videos para su asignatura y, utilizando Votalo, se encargaron de votar los proyectos de sus compañeros.

En este experimento participaron un total de 52 personas, de las cuales 33 completaron la encuesta de satisfacción. Cada uno de los alumnos asumió el rol de juez para votar los videos de los demás grupos, y también fueron añadidos como participantes en sus propios proyectos, de manera que no podían votar sobre sus propios videos.

El objetivo del experimento era que los alumnos utilizaran las diferentes funciones de la aplicación y validaran su funcionamiento, así como las mejoras de interfaz y de usabilidad implementadas en este segundo sprint. Todo esto mientras se automatizaba en gran medida el proceso de votación en comparación con los métodos utilizados en años anteriores.

Se les pasó a los early adopters una encuesta similar a la del primer experimento para recopilar la mayor cantidad de información posible sobre aspectos a mejorar y sobre el estado actual de la aplicación. A continuación, se mostrarán los resultados.

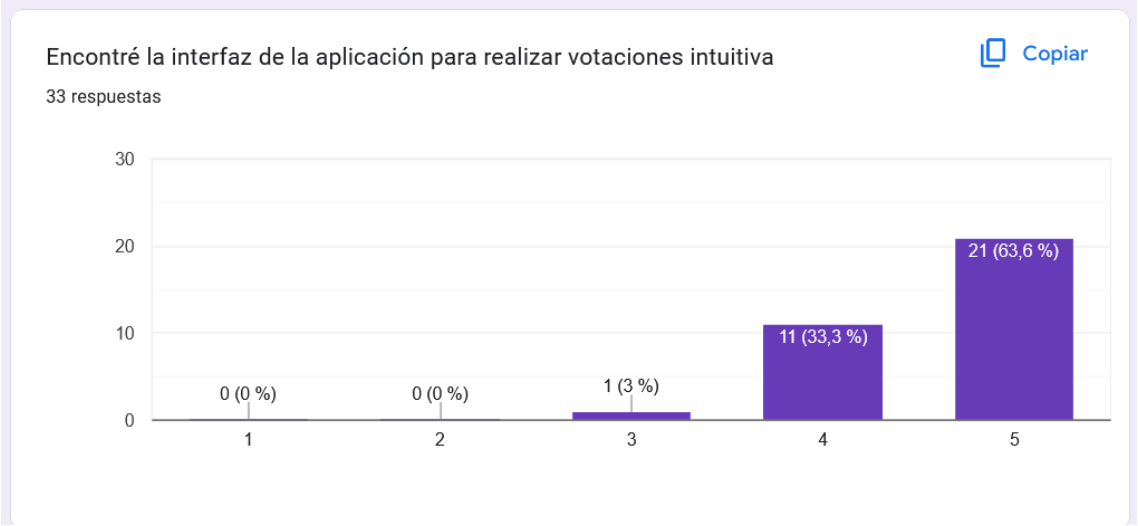

Figura 6.15: Interfaz Intuitiva

Prácticamente todos los usuarios encontraron la aplicación intuitiva, lo que demuestra que se cumplió el objetivo del segundo sprint y que las mejoras en la interfaz y usabilidad fueron efectivas.

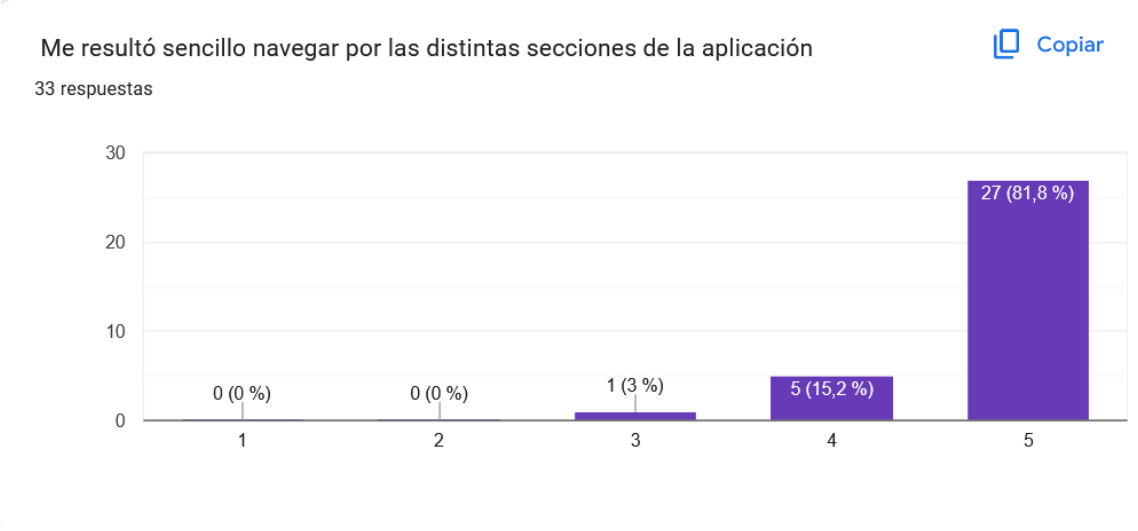

Figura 6.16: Facilidad de Navegación

De nuevo, prácticamente todos coincidieron en que fue sencillo navegar por la aplicación, lo cual es un feedback muy positivo.

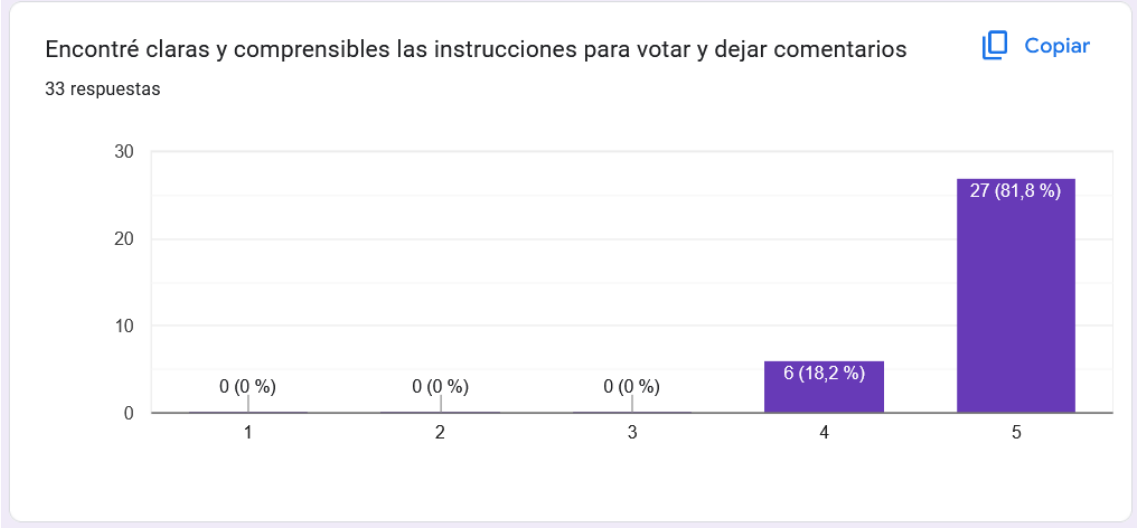

Figura 6.17: Claridad de Instrucciones

Esta era otra parte crucial a solucionar para este segundo experimento, y como puede verse, la gran mayoría de los alumnos estuvieron totalmente de acuerdo en que las instrucciones eran claras y comprensibles.

Figura 6.18: Funcionamiento de las Características

En este caso, solo 2 alumnos votaron negativamente, lo cual debe revisarse para comprobar en qué circunstancias la aplicación no funcionó correctamente.

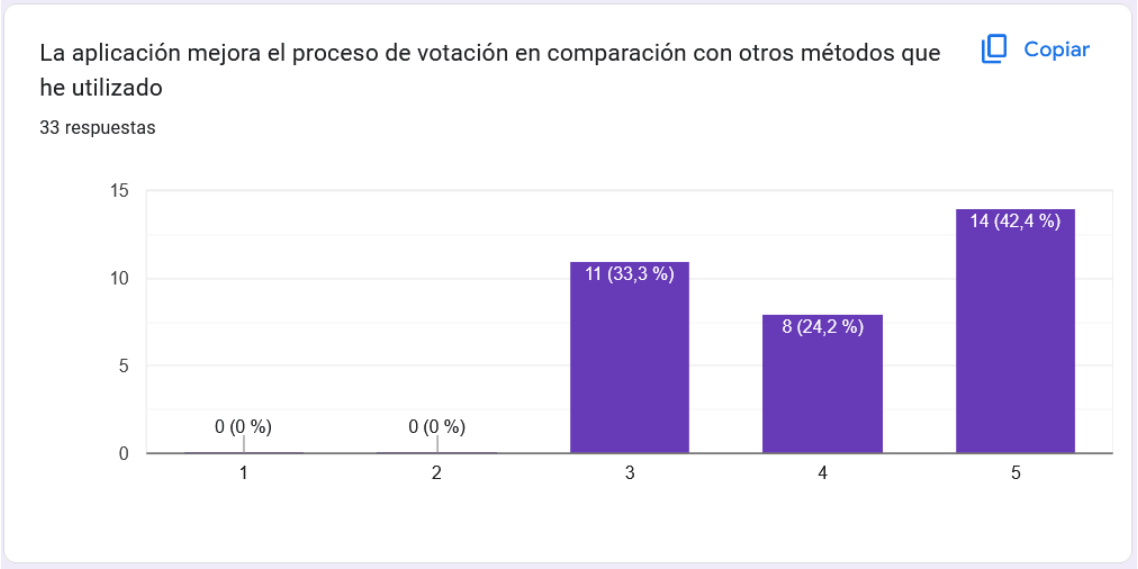

Figura 6.19: Mejora del Proceso de Votación

En este caso, las votaciones no fueron tan positivas. Si bien es cierto que muchos dijeron que sí mejora el proceso de votación respecto a otros métodos, hubo algunos que se mantuvieron neutrales. Esto puede deberse a que la aplicación ahorra trabajo principalmente a los organizadores de los eventos, mientras que los jueces y usuarios no perciben tanto beneficio comparado con otros métodos.

Figura 6.20: Recomendación de la Aplicación

Aquí vemos que, de nuevo, la mayoría de respuestas son bastante positivas, por lo que podemos concluir que el desarrollo de la aplicación va por buen camino.

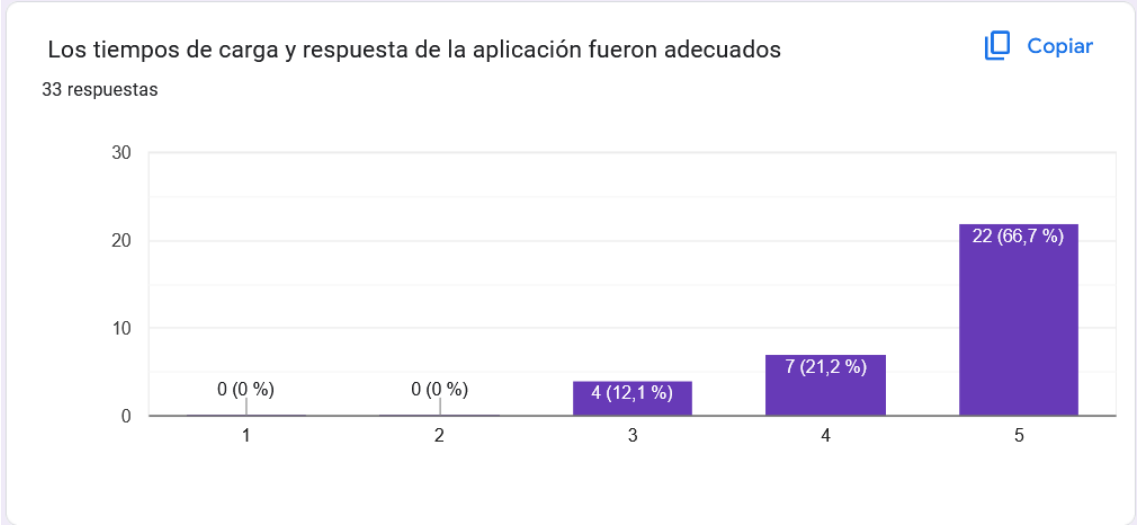

Figura 6.21: Tiempos de Carga

En este caso, para la mayoría de las personas los tiempos de carga fueron buenos, aunque algunos no estuvieron del todo de acuerdo. Esto es esperable ya que estamos utilizando la capa gratuita de Supabase, la cual no ofrece una gran cantidad de recursos.

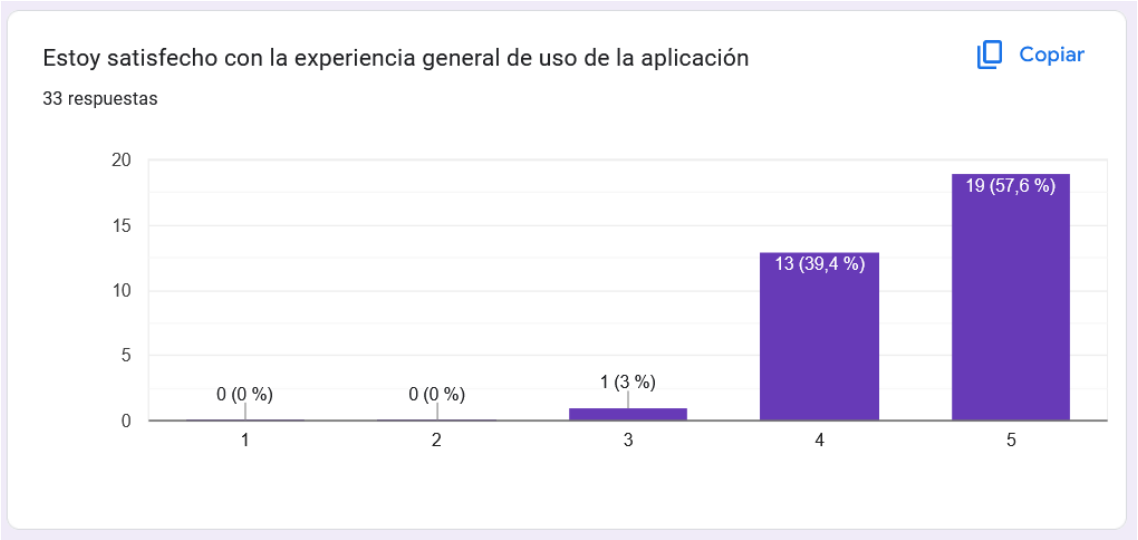

Figura 6.22: Satisfacción General

En cuanto a la satisfacción de los usuarios, se puede ver que también fue muy positiva, demostrando que las mejoras realizadas en este segundo sprint fueron efectivas.

En cuanto a las respuestas de la pregunta abierta, se recibieron una gran cantidad de sugerencias y mejoras para el desarrollo futuro de Votalo, tantas que sería imposible mostrarlas todas en este documento. Sin embargo, estas sugerencias han sido de gran ayuda para identificar los aspectos que sería interesante incluir en el futuro.

De esta encuesta podemos concluir que el desarrollo de Votalo va por buen camino y que estamos ofreciendo un servicio útil e innovador. Para el desarrollo futuro, nos centraremos en implementar nuevas opciones, como cambios en el perfil de usuario, y en solucionar errores puntuales que puedan surgir actualmente.

CAPÍTULO 7
Conclusiones

El desarrollo de Votalo ha sido una experiencia muy enriquecedora. Es emocionante ver cómo una aplicación evoluciona desde las primeras sesiones de brainstorming y bocetos iniciales hasta su finalización.

Se puede afirmar que se han cumplido con objetivos propuestos, ya que la aplicación ha sido validada en dos experimentos, el primero con 8 usuarios, y el segundo con mas de 50 usuarios. Hemos logrado desarrollar una aplicación que cumple con todas las funcionalidades especificadas y ha recibido críticas muy positivas en cuanto a su utilidad y eficiencia.

Al principio del proyecto, surgieron muchos problemas y complicaciones debido a la falta de familiaridad con algunas de las tecnologías utilizadas y la limitada experiencia en proyectos de este tipo. Sin embargo, a medida que avanzaba el proyecto, estos problemas se fueron resolviendo y nos familiarizamos más con las tecnologías, lo que permitió realizar el trabajo de manera más eficiente.

Desde el primer experimento, Votalo recibió buenas críticas en cuanto a su funcionalidad, aunque el diseño inicial no fue bien recibido. Esto nos motivó a seguir trabajando y mejorando. Finalmente, las mejoras implementadas resultaron en críticas muy positivas, destacando tanto la funcionalidad como el diseño de la aplicación, llegando a tener actualmente una aplicación totalmente funcional y con una interfaz amigable para los usuarios.

A nivel personal, el haber podido finalizar este proyecto me ha ayudado a aprender más sobre el desarrollo web, especialmente en lo que respecta a la separación entre frontend y backend y el uso de nuevas tecnologías punteras. Este proyecto me permitió aplicar y reforzar los conocimientos adquiridos durante mis estudios y explorar nuevas herramientas y metodologías de desarrollo.

El desarrollo de Votalo no habría sido posible sin los conocimientos adquiridos en la carrera de Ingeniería Informática. La manera en que se ha enfocado el proceso de desarrollo de la aplicación, incluyendo las fases por las que ha pasado el proyecto, ha sido posible gracias a los estudios. Asignaturas como Proyecto de Ingeniería de Software (PIN), Proyecto de Software (PSW) y Diseño de Software (DDS) han sido esenciales para el desarrollo de este proyecto. No solo se reforzaron los conocimientos obtenidos, sino que también se ampliaron y se pusieron en práctica, comprobando su importancia en el desarrollo real de un software. Conceptos como las unidades de trabajo, mockups, hitos y MVP han sido fundamentales tanto en la generación de la idea de negocio y el modelo de negocio como en el proceso de desarrollo de la aplicación.

En conclusión, Votalo es un proyecto que ha demostrado ser viable y prometedor, con gran potencial de crecimiento y liderazgo en su sector. La experiencia adquirida y los

conocimientos aplicados durante este proyecto serán invaluable en futuros desarrollos y proyectos profesionales.

CAPÍTULO 8
Trabajo futuro

Para asegurar la viabilidad comercial y el éxito continuo de Votalo, se delinean varias áreas de trabajo futuro. Primero, es fundamental la expansión de funcionalidades. Añadir características adicionales que aumenten la utilidad de la aplicación, como herramientas analíticas que permitan a los organizadores y participantes evaluar el rendimiento y obtener insights sobre las votaciones, es crucial. Implementar elementos de gamificación también puede mejorar la experiencia del usuario y aumentar la participación.

Otra tarea adicional importante para el futuro es la verificación de los requisitos no funcionales. Durante el desarrollo actual, no ha sido posible verificar estos requisitos, que incluyen aspectos como el rendimiento, la seguridad, la escalabilidad y la mantenibilidad de la aplicación. Este trabajo pendiente deberá ser una prioridad en futuras iteraciones del desarrollo de Votalo.

Otra área clave es la interfaz de usuario. Continuar mejorando la interfaz basada en la retroalimentación obtenida asegurará una mejor guía para los usuarios nuevos y mejoras en la accesibilidad y usabilidad de la aplicación.

La internacionalización y el multilingüismo también son esenciales para expandir el alcance de Votalo. Ampliar el soporte de idiomas para incluir más lenguas facilitará el acceso a una audiencia global.

Para comercializar y hacer crecer Votalo, es crucial desarrollar una estrategia de marketing robusta que aumente la visibilidad de la aplicación, destacando sus ventajas y diferenciadores frente a la competencia. Explorar modelos de monetización viables, como suscripciones premium, tarifas por evento y publicidad dentro de la aplicación, asegurará una fuente de ingresos sostenible. Además, buscar y establecer acuerdos con organizaciones que frecuentemente organizan eventos, como universidades, empresas y organizaciones sin fines de lucro, fomentará el uso de Votalo.

Finalmente, es importante la innovación continua. Establecer un ciclo continuo de recopilación de sugerencias de los usuarios y la implementación de actualizaciones periódicas mantendrá la aplicación relevante y acorde con las necesidades de los usuarios. Al abordar estas áreas de trabajo futuro, Votalo no solo mejorará su oferta actual, sino que también se asegurará un crecimiento sostenible a lo largo del tiempo.

Bibliografía

[1] Erdem Gürel and Merba Tat. SWOT Analysis: A Theoretical Review. *The Journal of International Social Research*

[2] Ash Maurya. *Running Lean: Iterate from Plan A to a Plan That Works*. O'Reilly Media, Sebastopol, CA, 2012.

[3] HubSpot. TAM, SAM & SOM: What Do They Mean & How Do You Calculate Them? Consultado en https://blog.hubspot.com/marketing/what-is-tam-sam-som.

[4] Foundation Inc. TAM, SAM, SOM: How to Calculate Them for Your Industry. Consultado en https://foundationinc.co/lab/tam-sam-som.

[5] Expert Market Research. Events Industry Market Size, Share, Growth Statistics 2032. Consultado en https://www.expertmarketresearch.com/reports/events-industry-market.

[6] Allied Market Research. Events Industry Market Share And Size, Growth Statistics 2032. Consultado en https://www.alliedmarketresearch.com/events-industry-market.

[7] Market Research Future. Events Industry Market Size, Share and Growth Forecast By 2032. Consultado en https://www.marketresearchfuture.com/reports/events-industry-market.

[8] Eric Ries. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Business, New York, 2011.

[9] Agile Alliance. What is a Minimum Viable Product (MVP)? Consultado en https://www.agilealliance.org/glossary/mvp.

=1

APÉNDICE A
OBJETIVOS DE DESARROLLO SOSTENIBLE

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**			X	
ODS 4. **Educación de calidad.**	X			
ODS 5. **Igualdad de género.**		X		
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**		X		
ODS 9. **Industria, innovación e infraestructuras.**	X			
ODS 10. **Reducción de las desigualdades.**		X		
ODS 11. **Ciudades y comunidades sostenibles.**				X
ODS 12. **Producción y consumo responsables.**				X
ODS 13. **Acción por el clima.**				X
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**			X	
ODS 17. **Alianzas para lograr objetivos.**		X		

Figura A.1: Grado de relación del proyecto Votalo con los ODS

ODS 1. Fin de la pobreza: No procede, ya que el proyecto Votalo no está relacionado con la erradicación de la pobreza.

ODS 2. Hambre cero: No procede, ya que el proyecto no aborda la erradicación del hambre.

ODS 3. Salud y bienestar: Bajo, porque aunque no es el objetivo principal, la organización y gestión eficiente pueden contribuir al bienestar general.

ODS 4. Educación de calidad: Alto, porque Votalo es utilizado en un contexto educativo para mejorar la organización y evaluación de proyectos.

ODS 5. Igualdad de género: Medio, ya que fomenta la participación igualitaria en los procesos de votación y evaluación.

ODS 6. Agua limpia y saneamiento: No procede, ya que el proyecto no aborda temas de agua y saneamiento.

ODS 7. Energía asequible y no contaminante: No procede, porque el proyecto no está relacionado con la energía.

ODS 8. Trabajo decente y crecimiento económico: Medio, ya que facilita el trabajo de organización de eventos y puede contribuir al crecimiento económico en sectores específicos.

ODS 9. Industria, innovación e infraestructuras: Alto, porque Votalo es una herramienta innovadora que mejora las infraestructuras de evaluación y votación en eventos.

ODS 10. Reducción de las desigualdades: Medio, ya que permite una evaluación justa y estructurada, reduciendo sesgos.

ODS 11. Ciudades y comunidades sostenibles: No procede, ya que el proyecto no está directamente relacionado con este objetivo.

ODS 12. Producción y consumo responsables: No procede, porque el proyecto no aborda la producción y el consumo.

ODS 13. Acción por el clima: No procede, ya que no está relacionado con la acción climática.

ODS 14. Vida submarina: No procede, porque el proyecto no aborda temas marinos.

ODS 15. Vida de ecosistemas terrestres: No procede, ya que el proyecto no está relacionado con los ecosistemas terrestres.

ODS 16. Paz, justicia e instituciones sólidas: Bajo, porque promueve procesos justos y estructurados en la evaluación.

ODS 17. Alianzas para lograr objetivos: Medio, porque fomenta la colaboración entre estudiantes, profesores y participantes en eventos.