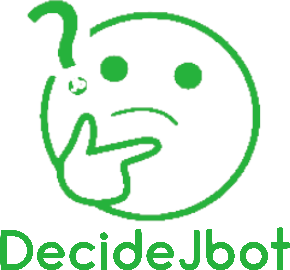


PROYECTO FIN DE CICLO



Autor

Julián Bienzobas Arnas

Tutor

Michel Cuartero Cimorra

DESARROLLO DE APLICACIONES MULTIPLATAFORMA

2021/2022

**Resumen**

Este TFG (Trabajo de Fin de Grado) ha sido realizado por Julián Bienzobas Arnas, estudiante del Grado Superior de Desarrollo de Aplicaciones Multiplataforma en el IES Tubalcain de Tarazona en el curso 2021/2022.

El proyecto titulado “DecideJbot” consta de dos aplicaciones, una aplicación móvil para Android y una aplicación Web desarrollada con el Framework Angular.

El proyecto se basa en funciones enfocadadas al azar, siendo las aplicaciones un instrumento que toma “decisiones” aleatoriamente por el usuario.

Las aplicaciones realizan funciones aleatorias como pueden ser la realización de un sorteo, el lanzamiento de un dado o una moneda, la rotación de una flecha o la generación de números aleatorios.

**Palabras clave**

Decisión, Sorteo, Participantes, Lista, Ruleta, Premios, Ganador, Actividad, Android, HTML, Javascript, Persistencia, Base de datos, Aleatorio, Historial, Resultados, Java, Azar, Rifa, Angular, Lotería, Moneda, Cara o Cruz, Dado, Flecha, Ganador, Premio, Elección, CSS, Android Studio, Visual Studio Code.

**Abstract**

This End of Degree Project has been carried out by Julián Bienzobas Arnas, student of the Higher Degree in Multiplatform Application Development at the IES Tubalcain of Tarazona in the 2021/2022 academic year.

The project entitled "DecideJbot" consists of two applications, a mobile application for Android and a Web application developed with the Angular Framework.

The project is based on randomly focused functions, with applications being an instrument that makes "decisions" randomly for the user.

Applications perform random functions such as drawing a lottery, rolling a die or coin, rotating an arrow, or generating random numbers.

**Keywords**

Decision, Draw, Participants, List, Roulette, Prizes, Winner, Activity, Android, HTML, Javascript, Persistence, Database, Random, History, Results, Java, Chance, Raffle, Angular, Lottery, Coin, Heads or Tails, Dice, Arrow, Winner, Prize, Choice , CSS, Android Studio, Visual Studio Code.

**TABLA DE CONTENIDOS**

[Descripción del proyecto 4](#_Toc106018913)

[**Contexto del proyecto** 4](#_Toc106018914)

[**Ámbito y entorno** 4](#_Toc106018915)

[**Análisis de la realidad** 5](#_Toc106018916)

[**Solución y justificación** 6](#_Toc106018917)

[**Destinatarios** 7](#_Toc106018918)

[**Objetivo del proyecto** 8](#_Toc106018919)

[**Objective of the Project** 9](#_Toc106018920)

[Acuerdo del proyecto 10](#_Toc106018921)

[**Requisitos funcionales y no funcionales** 10](#_Toc106018922)

[**Tareas** 11](#_Toc106018923)

[**Metodología a seguir para la realización del proyecto** 13](#_Toc106018924)

[**Planificación temporal de tareas** 13](#_Toc106018925)

[**Presupuesto** 14](#_Toc106018926)

[Análisis y diseño 15](#_Toc106018927)

[**Modelado de datos** 15](#_Toc106018928)

[**Análisis y diseño de la interfaz de usuario** 18](#_Toc106018929)

[**Análisis y diseño del sistema funcional** 57](#_Toc106018930)

[**Diseño de la arquitectura de la aplicación** 82](#_Toc106018931)

[**Tecnologías y herramientas usadas** 82](#_Toc106018932)

[**Arquitectura de componentes de la aplicación** 84](#_Toc106018933)

[Implementación e implantación del sistema 87](#_Toc106018934)

[**Implementación** 87](#_Toc106018935)

[**Pruebas** 87](#_Toc106018936)

[Documento de cierre 88](#_Toc106018937)

[**Documento de despliegue y configuración** 88](#_Toc106018938)

[**Manual de usuario** 89](#_Toc106018939)

[**Resultados obtenidos y conclusiones** 90](#_Toc106018940)

[**Seguimiento y desviación respecto a la planificación** 90](#_Toc106018941)

[Bibliografía 91](#_Toc106018942)

[Anexos 91](#_Toc106018943)

# **Descripción del proyecto**

## **Contexto del proyecto**

La idea de DecideJbot surge a raíz de la indecisión y como un reto personal.

Debido a ciertas indecisiones, allá por el año 2019 pasó por mi cabeza la idea de tomar algunas decisiones con ayuda de “la suerte” o el azar, además, con la llegada de la informática y la programación a mi vida, con mis gustos y ambiciones, surgió la idea y el reto personal de crear mi propia aplicación que tomase decisiones por mí. Así fue como nació “DecideJbot”

## **Ámbito y entorno**

Si con entorno nos referimos a los dispositivos, este proyecto está enfocado para casi todo tipo de entorno digital, ya sean dispositivos móviles con sistema operativo Android, o cualquier dispositivo que disponga de un navegador web.

Por otra parte, el proyecto y la aplicación pueden utilizarse en cualquier ámbito y entorno.

La aplicación y el proyecto “DecideJbot” surgió enfocado para ámbito personal, ya que era un proyecto y reto que quería realizar y utilizar personalmente.

Otro ámbito y entorno al que se puede dirigir y utilizar la aplicación y proyecto es el ámbito y entorno familiar, ya que, por ejemplo, se podría utilizar para decidir sobre las tareas del hogar.

Otros entornos, podrían ser entornos empresariales, y escolares, ya que la aplicación podría utilizarse, por ejemplo, para que una empresa lance y realice un sorteo para sus clientes, o, en el caso de las escuelas, tomar decisiones para formar grupos, decidir un delegado…

Por último, el proyecto tiene un enfoque relacionado con el ocio, ya que la aplicación dispone de secciones que se pueden utilizar en grupos de amigos, fiestas, juegos… Como puede ser la función de girar una ruleta o lanzar un dado.

## **Análisis de la realidad**

Investigando y probando aplicaciones similares o que puedan competir con “DecideJbot” se pueden hablar de algunas:

* Sortea2: Exclusiva para navegadores, con funciones dedicadas a sorteos y permite almacenar algún dato.
* AppSorteos: Exclusiva para navegadores. Incluye varios apartados que incluye mi proyecto, y otros que se implementarán en futuras actualizaciones, además de incluir una versión de pago.

Cómo punto fuerte para DecideJbot contra esta aplicación web, mi aplicación dispone de almacenamiento de datos.

* EchaloASuerte: Exclusiva para navegadores. La mayoría de las funciones que ofrece se incluyen en mi aplicación o se incluirán en futuras actualizaciones.

El punto fuerte de mi aplicación a comparación de esta es que la mía permite el almacenamiento de los participantes para los sorteos.

El diseño de mi aplicación comenzó basándose en el diseño de EchaloASuerte.

Respecto a web, puedo destacar las funciones y diseño de esas 3 aplicaciones. Existen varias aplicaciones web que realizan funciones similares o idénticas.

Respecto a aplicaciones móviles para Android he podido investigar y probar varias, encontrando por separado aplicaciones que realizan sorteos, generan números aleatorios o realizan distintos sorteos como podían ser de grupos, pero ninguna aplicación incluye en una sola todas las funciones que se han implementado en DecideJbot, así como no incluyen los mecanismos de guardado de datos de manera gratuita.

La realidad de este proyecto es, que pese a existir varias aplicaciones, ya sean en móvil o en navegador que realizan las funciones de DecideJbot, esta nació como una necesidad y reto personal, implementando todas las funciones que querría ver en una aplicación de “Sorteos y decisiones” en una única. Además, DecideJbot está pensada para que futuras versiones y la aplicación final implementen decisiones sobre temas específicos, ayudando al usuario a tomar decisiones en estos temas que podrían ser decidir una serie, una película, una actividad, un juego, un alimento… Además, de incluir un registro de todas las decisiones del usuario, ya sean sorteos variados o decisiones con un tema específico.

Otro punto positivo de DecideJbot es la posibilidad de poder tener una sesión o no, a gusto del usuario.

¿Quieres usar datos de manera global en móvil y en la aplicación web? Entonces, conviene tener y usar una sesión.

¿Quieres darle un uso casual y rápido a la aplicación? Entonces, puedes usarla sin sesión. Además de destacar los datos compartidos en la aplicación Web y móvil, para acceder a esos datos en cualquier dispositivo.

Lo último a comentar, analizando la aplicación, son sus inconvenientes.

Un inconveniente puede ser la cantidad de aplicaciones móvil y web que como ya se ha comentado, existen y realizan funciones similares.

Otro inconveniente, en este caso para el proyecto en sí, es la realización de 2 aplicaciones por separado (Web y Android), ya que eso significa doble trabajo y se podría ahorrar y solucionar realizando una única aplicación Hibrida, aunque para el caso de este proyecto, se ha decidido trabajar con 2 aplicaciones, independientes, pero con datos comunes.

## **Solución y justificación**

Cómo se comenta anteriormente, la idea de proyecto surge a raíz de ciertas indecisiones en distintas situaciones, además de la aparición del mundo de la programación y mi interés en este mundo.

Debido a las indecisiones que se comentan, empecé a utilizar el azar para poder tomar ciertas decisiones, y, para ser exactos, comencé a utilizar una web para ayudarme a tomar estas decisiones. Siendo exactos, la web utilizada era la de “Sortea2”.

A raíz de usar esa aplicación web e ir adquiriendo más conocimientos y más gustos e inquietudes en el mundo de la programación, me surgió la idea de dejar de utilizar esa web y poder crear mi propia aplicación, con funciones similares, pero con mi propio diseño, funciones y datos.

Así pues, fue donde nació la idea de crear mi aplicación, con ese objetivo y, además, como reto personal.

Para comenzar, DecideJbot comenzó siendo una única aplicación web realizada con PHP y que simplemente realizaba “Sorteos simples” y decidía sobre algún tema específico, que fueron series y música.

Posteriormente, y ya como parte de este trabajo de Fin de Grado, me surgió la idea de trabajar la aplicación Web con JavaScript, para solucionar temas de recargas de página y datos que eran un pequeño inconveniente de PHP.

Además, como parte del proyecto y necesidad para este, ya con conocimientos sobre Java y Android, nació y se comenzó a crear la aplicación móvil.

Finalmente, con conocimientos adquiridos en las prácticas y conociendo el Framework Angular, surgió la idea de adaptar la aplicación web a este Framework.

Respecto al almacenamiento de datos, estos se comenzaron almacenando en una base de datos MySQL, pero posteriormente, los datos se han terminado almacenando de manera local, en Android con SQLite y en web en el propio navegador con las funciones de LocalStorage. Además, implementando un almacenamiento mixto con una base de datos NoSQL de Firebase, llamada Firestore,

Con todos estos cambios y evoluciones se ha conseguido que DecideJbot sea un proyecto formado por dos aplicaciones, que realizan funciones basadas en la aleatoriedad y el azar, que pueden trabajar con sesiones de usuario pueden trabajar con datos almacenados localmente y externamente, teniendo en conjunto estos últimos en ambas aplicaciones.

## **Destinatarios**

DecideJbot no está destinada a un tipo de usuario específico, ya que es una aplicación que todo el mundo puede utilizar, algunos destinatarios podrían ser:

* Usuarios indecisos que pueden utilizarla para ayudarse a tomar sus decisiones.
* Usuarios que necesitan realizar un sorteo.
* Usuarios que quieren usar un elemento para obtener un valor aleatorio: Sorteo, flecha, dado, moneda…
* Usuarios que quieren realizar grupos.
* Usuarios que quieren asociar elementos.
* Usuarios que quieren disponer de un registro de actividades realizadas

## 

## **Objetivo del proyecto**

DecideJbot es una aplicación que toma decisiones por los usuarios, además de llevar un registro de esta.

Nos permite realizar desde un sorteo simple, indicando únicamente los participantes y los números de premios a repartir, permitiendo almacenar estos datos local, externamente o de manera conjunta para una posible reutilización, permitiéndonos visualizar los resultados de estos sorteos, además de poder almacenarlos y compartirlos.

Por otra parte, nos permite realizar sorteos más avanzados, incluyendo más opciones, como puede ser la programación de un sorteo en una fecha específica, haciendo así que los resultados de ese sorteo no se generen hasta la fecha específicada.

También, nos permite generar números aleatorios en un rango especificado, indicando un número mínimo y un máximo.

Por último, se permite realizar unas acciones específicas que harán uso del azar, como puede ser el lanzamiento de un dado, el lanzamiento de una moneda o la rotación de una flecha.

Además, la aplicación permitirá al usuario trabajar con sus datos creados por él o por la aplicación, permitiéndole crear, programarlos, borrarlos, combinarlos o visualizarlos con un uso optativo de sesión.

## **Objective of the Project**

DecideJbot is an application that makes decisions for users, in addition to keeping track of it.

It allows us to carry out from a simple draw, indicating only the participants and the numbers of prizes to be distributed, allowing us to store this data locally, externally or jointly for possible reuse, allowing us to visualize the results of these draws, in addition to being able to store and share them.

On the other hand, it allows us to make more advanced draws, including more options, such as scheduling a draw on a specific date, thus making the results of that draw not generated until the specific date.

Also, it allows us to generate random numbers in a specified range, indicating a minimum and a maximum number.

Finally, it is allowed to perform specific actions that will make use of chance, such as the throwing of a die, the throwing of a coin or the rotation of an arrow.

In addition, the application will allow the user to work with their data created by him or by the application, allowing him to create, program, delete, combine or visualize them with an optional use of session.

# **Acuerdo del proyecto**

## **Requisitos funcionales y no funcionales**

|  |
| --- |
| **Requisitos funcionales** |
| El usuario puede decidir si iniciar o no sesión |
| El usuario puede realizar un sorteo |
| El usuario puede guardar los resultados de un sorteo (Externamente) |
| El usuario puede acceder a los resultados generados de un sorteo. |
| El usuario puede compartir resultados. |
| El usuario puede añadir participantes a un sorteo. |
| El usuario puede reemplazar participantes de un sorteo. |
| El usuario puede eliminar participantes de un sorteo. |
| El usuario puede guardar un sorteo (Local, externamente o de manera conjunta) |
| El usuario puede programar la fecha de realización de un sorteo. |
| El usuario puede lanzar un dado. |
| El usuario puede generar números aleatorios. |
| El usuario puede lanzar una moneda. |
| El usuario puede girar una flecha. |
| El usuario puede gestionar sus resultados (Crear, ver, editar y eliminar) |

|  |
| --- |
| **Requisitos no funcionales** |
| La aplicación móvil es exclusiva para Android. |
| La aplicación web se puede utilizar en cualquier dispositivo con un navegador web. |
| Los datos se pueden almacenar localmente:   * Android (SQLite con Sugar ORM) * Web (LocalStorage) (Almacenamiento por navegador) |
| Los datos se pueden almacenar externamente:   * Firestore de Firebase. |
| * Las aplicaciones generan resultados aleatoriamente. |

## **Tareas**

|  |  |  |
| --- | --- | --- |
| **App Móvil** | | |
| Autenticación de usuarios | Registro | |
| Inicio de sesión | |
| Inicio de sesión con Google | |
| Sorteos | Sorteo simple | Añadir un título |
| Añadir participantes |
| Borrar todos los participantes |
| Guardar sorteo localmente |
| Ver sorteos locales |
| Cargar sorteos locales (Añadir/Reemplazar participantes) |
| Eliminar sorteos locales |
| Permitir decidir si realizar el sorteo con una cuenta atrás |
| Guardar sorteo externamente |
| Implementar sonidos |
| Generador de números aleatorios | Diseño |
| Lógica |
| Dado | |
| Flecha | |
| Cara o cruz | |
|  | Resultados | Guardar resultados (Externamente) |

|  |  |
| --- | --- |
| **App Web** | |
| Usuarios | Registro |
| Inicio de sesión |
| Inicio de sesión con Google |

|  |  |  |
| --- | --- | --- |
| Sorteos | Sorteo simple | Añadir un título |
| Añadir participantes |
| Borrar todos los participantes |
| Guardar sorteo localmente |
| Ver sorteos locales |
| Cargar sorteos locales (Añadir/Reemplazar participantes) |
| Eliminar sorteos locales |
| Permitir decidir si realizar el sorteo con una cuenta atrás |
| Guardar sorteo externamente |
| Cargar sorteos externos (Añadir/Reemplazar participantes) |
| Implementar sonidos |
| Generador de números aleatorios | Diseño |
| Lógica |
| Sorteo avanzado | Diseño de los pasos |
| Lógica |
| Programar fecha |
| Resultados | Guardar resultados (Externamente) |
| Ver resultados |
| Actualizar resultados |
| Eliminar resultados |
| Adelantar fecha de resultados a la actual |

## **Metodología a seguir para la realización del proyecto**

Para realizar el proyecto he trabajado con la metodología Kanban, realizando un tablero con las tareas que hay que realizar en todo el proyecto, las tareas en proceso y las tareas terminadas.

Utilicé la aplicación Trello para poder realizar el Kanban, creándome una plantilla con las 3 columnas (Pendientes, En proceso y Finalizadas), y una vez creadas, añadí todas las tareas y subtareas a la columna de “Pendientes”.

Analicé las tareas, para decidir cuáles eran prioritarias y las moví a la columna “En proceso” para ir realizando las tareas o subtareas planteadas.

Alguna vez, se trabajaban varias tareas a la vez, ya que, al realizar dos aplicaciones, realizaba las tareas para ambas aplicaciones simultáneamente.

Una vez se termina una tarea, se indica la fecha de finalización y se mueve a la columna “Finalizada”, aunque podría darse la situación de tener que realizar una nueva subtarea sobre esa tarea y que se volviese a mover a la columna “En proceso”.

## **Planificación temporal de tareas**

Para planificar el tiempo que se le quiere dedicar a cada tarea, lo primero que hay que tener en cuenta es la fecha final de entrega de la versión del proyecto.

Siendo la fecha de entrega el día 17 de Junio de 2022, y comenzando el proyecto aproximadamente el 4 de Abril de 2022, lo primero que hay que tener en cuenta es el orden en el que se quieren realizar las tareas, ya que unas pueden depender de otras, y para esto, se utilizó la metodología.

La primera tarea, era la planificación del proyecto, teniendo una idea clara de cómo iba a ser, que funcionalidades iba a tener y cumplir…

Una vez planificado el proyecto, se pudo planificar todas las tareas, y se podía aproximar un tiempo a cada tarea. Cabe destacar, que los tiempos no iban a ser exactos, y que podrían llevar más o menos tiempo dependiendo de las complicaciones u otros aspectos que puedan afectar a la tarea.

Otra de las primeras tareas, una vez planificado el proyecto y teniéndolo en mente, era realizar un primer diseño de las aplicaciones, realizando mockups de las pantallas que tendrán las aplicaciones.

Una vez planeado el diseño de las aplicaciones, el diseño se puede comenzar a desarrollar y programar.

El planteamiento que he realizado para el el desarrollo de las tareas del proyecto ha sido realizar el diseño de una pantalla y posteriormente, planear, diseñar y programar su lógica, haciéndolo después de nuevo en la otra aplicación.

En algunas ocasiones, “pausaba” el desarrollo de una tarea y desarrollaba otra simultáneamente.

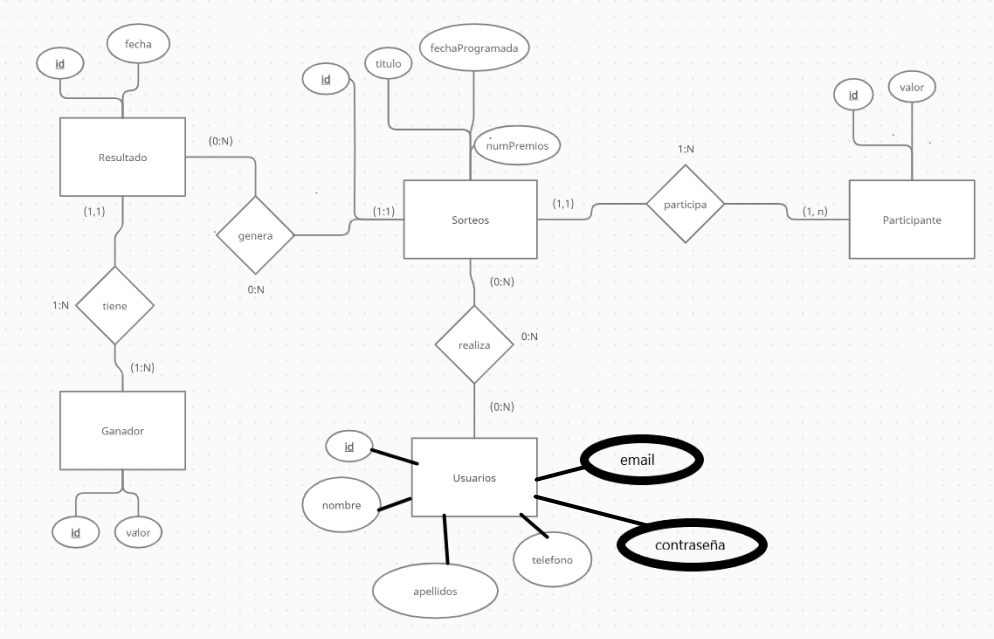
## **Presupuesto**

|  |  |  |
| --- | --- | --- |
| **DecideJbot v1.0** | | ***15/06/2022*** |
|  | | |
| **Descripción** | **Unidades** | **Ingresos** |
| Horas trabajadas | 360 | + 2880 € |
| Testers |  | * 500 € |
| Diseñadores |  | * 300 € |
|  | | |
| Publicación de la app móvil en Google Play |  | * 25 € |
| Publicación de la app Web |  | * 55 € / año |
|  | | |
| Seguro/Mantenimiento |  | + 300 € |
|  |  |  |
| Total (SIN IVA) |  | 2300 € |
| IVA | 21% | + 483€ |
|  | | |
| **TOTAL** |  | **2783 €** |

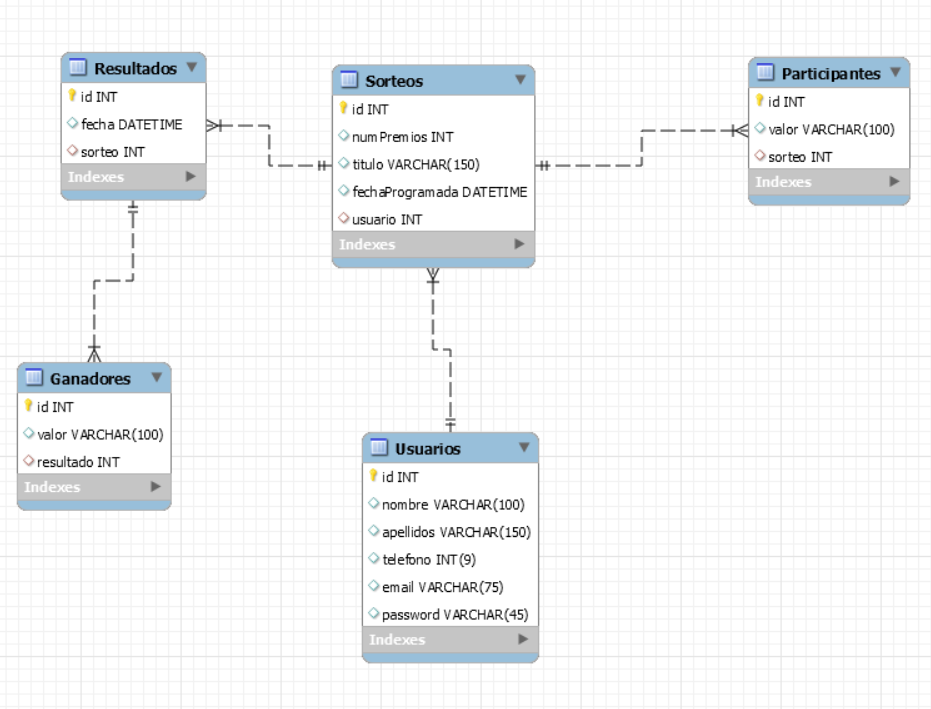
# **Análisis y diseño**

## **Modelado de datos**

**Modelo entidad relación:**



**Modelo Relacional:**



**Casos de uso:**



|  |  |
| --- | --- |
| CU-1 | Iniciar sesión |
| **Descripción:**  El usuario quiere iniciar sesión. | |
| **Actores:**  Usuario | |
| **Precondiciones:**   * El usuario debe estar registrado * Se requiere conexión a internet. | |
| **Caso normal de uso:**   * El usuario introduce sus datos. * El usuario pulsa “Iniciar sesión” | |
| **Poscondiciones:**  El usuario es redirigido a la pantalla inicial con la sesión iniciada. | |
| **Alternativa:**  Aparecen mensajes de error. | |

|  |  |
| --- | --- |
| CU-2 | Acceder sin sesión |
| **Descripción:**  El usuario quiere utilizar la aplicación sin iniciar sesión. | |
| **Actores:**  Usuario | |
| **Caso normal de uso:**   * El usuario pulsa en la aplicación móvil “Acceder sin sesión” | |
| **Poscondiciones:**  El usuario es redirigido a la pantalla inicial sin sesión iniciada. | |

|  |  |
| --- | --- |
| CU-3 | Registrarse |
| **Descripción:**  El usuario quiere acceder a la pantalla de Registro. | |
| **Actores:**  Usuario | |
| **Caso normal de uso:**  El usuario pulsa el botón “Registrarse” | |
| **Poscondiciones:**  El usuario es redirigido a la pantalla de registro. | |

|  |  |
| --- | --- |
| CU-4 | Acceder con Google |
| **Descripción:**  El usuario quiere acceder a la aplicación con la cuenta de Google vinculada. | |
| **Actores:**  Usuario | |
| **Precondiciones:**   * El usuario debe disponer de una cuenta de Google. * Se requiere conexión a internet. | |
| **Caso normal de uso:**   * El usuario selecciona “Acceder con Google” * El usuario selecciona la cuenta de Google con la que desea acceder. | |
| **Poscondiciones:**  El usuario es redirigido a la pantalla inicial con la sesión iniciada. | |

## **Análisis y diseño de la interfaz de usuario**

La primera parte del análisis y diseño de la interfaz de usuario está formada por unos mockups de las pantallas y las páginas de las aplicaciones. Los mockups son una primera idea de como van a ser estas pantallas y páginas, de que están formadas, y se acompaña de una pequeña descripción de como actuarían sus componentes, por si es necesario conocerlo para el diseño.

**Aplicación móvil:**

Al iniciar la aplicación, aparece una “Splash Screen” (Pantalla de carga) con el logo de la aplicación, que cambiará de pantalla automáticamente a los 5 segundos.

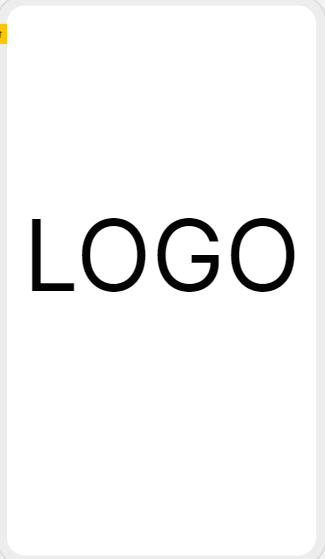
****

Ilustración 1: Splash Screen

Tras la pantalla de carga, en caso de que el usuario no tenga una sesión iniciada, aparecerá la pantalla de inicio de sesión:

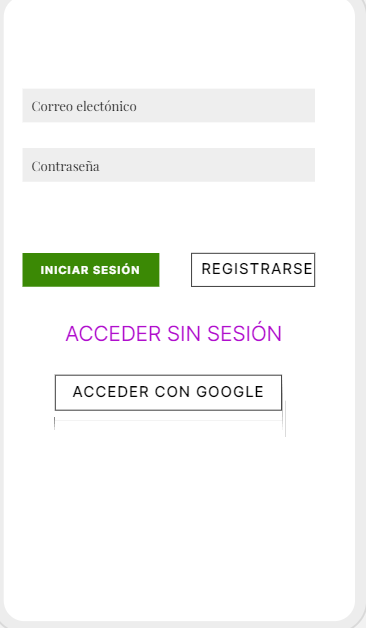


Ilustración 2: Pantalla de inicio de sesión

Si en la pantalla de inicio de sesión, el usuario pulsa el botón de “Registrarse”, accederá a la pantalla de registro

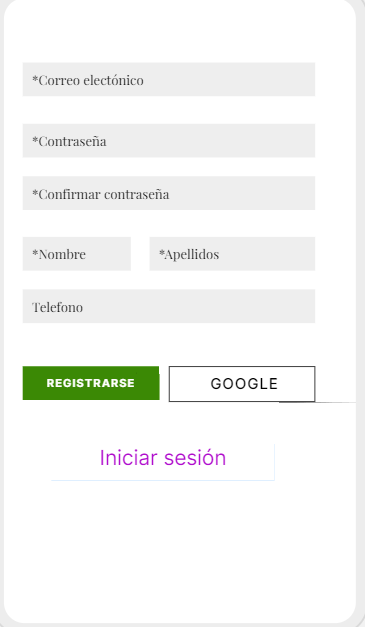


Ilustración 3: Pantalla de registro de usuario

Tras la pantalla de carga, en caso de que el usuario tenga iniciada sesión, o tras iniciar sesión, aparecerá la “Main Activity” o pantalla principal, con el logo de la aplicación y un menú a las secciones de la aplicación.

La sección principal o “Home” dispone de un botón en formato de carta para iniciar o cerrar sesión y otro botón para acceder a la web.



Ilustración 4: Main Activity

En caso de pulsar el botón “Sorteos” del menú de la pantalla principal, aparecerá el fragmento de la sección de sorteos.



Ilustración 5: Fragmento Sorteos

En la sección de sorteos, al pulsar el botón de “Sorteo simple” nos llevará a la pantalla de Sorteo Simple.

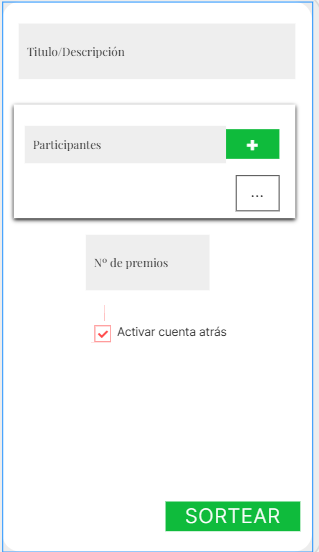


Ilustración 6: Pantalla Sorteo Simple

En el sorteo simple, si falta algún dato de completar, aparecerá una seekbar con un mensaje de error.

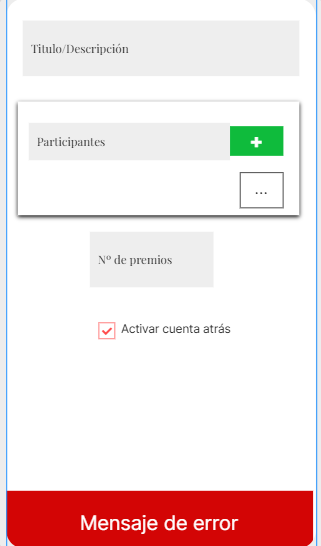


Ilustración 7: Sorteo Simple con error

En el sorteo simple, al pulsar en los tres puntitos, aparecerá un menú para poder gestionar el sorteo (Guardar, cargar o borrar)

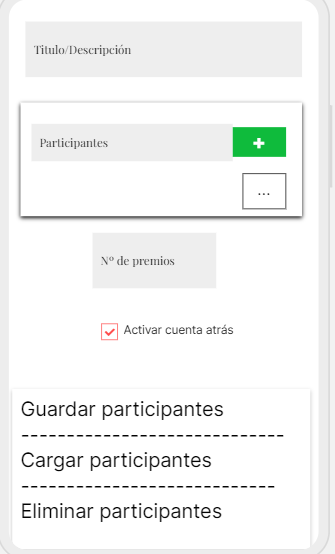


Ilustración 8: Menú en Sorteo Simple

En el sorteo simple, si están todos los campos rellenos correctamente, al dar en el botón “Sortear” se generarán y mostrarán los resultados listados en un dialogo.



Ilustración 9: Lista de ganadores en Sorteo Simple

En el sorteo simple, al seleccionar los tres puntitos y seleccionar la opción “Guardar participantes” aparecerá un menú con el que el usuario podrá decidir si guardar los datos del sorteo localmente, externamente o de ambas maneras.

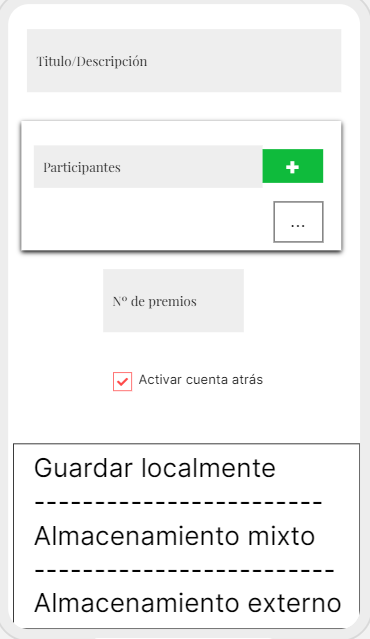


Ilustración 10: Menú guardar en Sorteo Simple

En el sorteo simple, al seleccionar los tres puntitos y seleccionar la opción “Cargar listas de participantes” aparecerá un cuadro de diálogo con la lista de sorteos, permitiendo al usuario cargar las listas de participantes de estos añadiéndolas a la lista actual o reemplazando la lista actual.

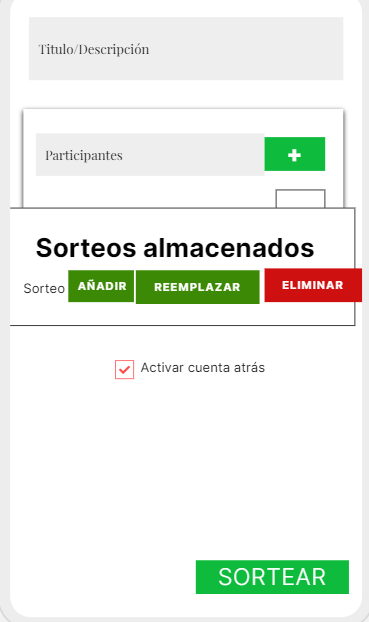


Ilustración 11: Lista de sorteos cargadas en Sorteo Simple

En la sección de sorteos, al pulsar en la tarjeta de “Números aleatorios” llevará al usuario a la pantalla del generador de números aleatorios

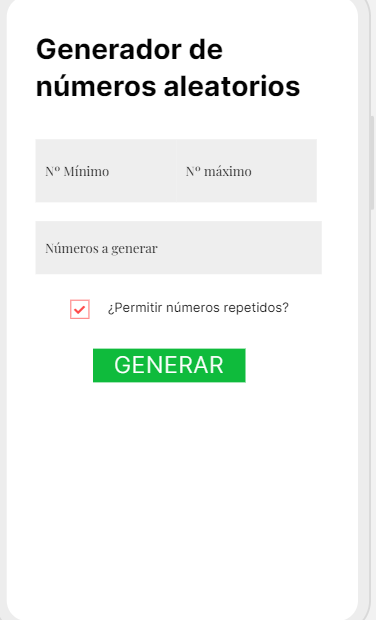


Ilustración 12: Generador de números aleatorios

En el generador de números aleatorios, si todos los campos están completados y correctamente, al pulsar el botón “Generar”, aparecerán los resultados en la pantalla.

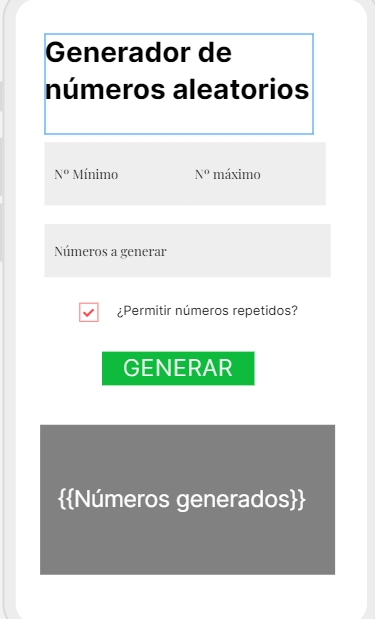


Ilustración 13: Generador de números aleatorios con resultados

En el generador de números aleatorios, en caso de no haber completado todos los campos o que alguno de ellos tenga un valor incorrecto, aparecerá un mensaje de error.

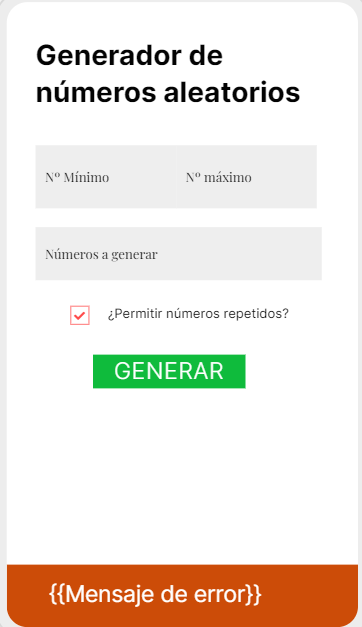


Ilustración 14: Generador de números aleatorios con error

En la sección de Sorteos, al pulsar en la tarjeta de girar una flecha, nos llevará a su respectiva pantalla.

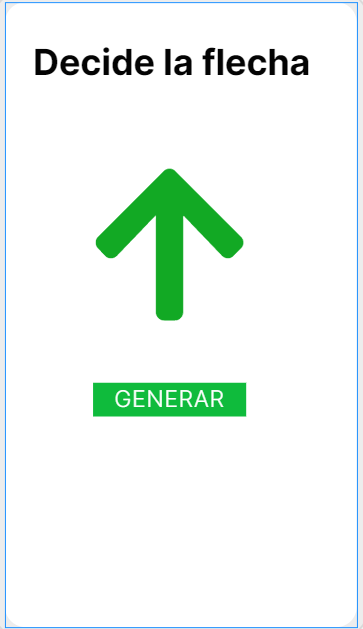


Ilustración 15: Pantalla "Girar una flecha"

En la sección de Sorteos, al pulsar en la tarjeta de “Lanzar un dado”, nos llevará a su respectiva pantalla.

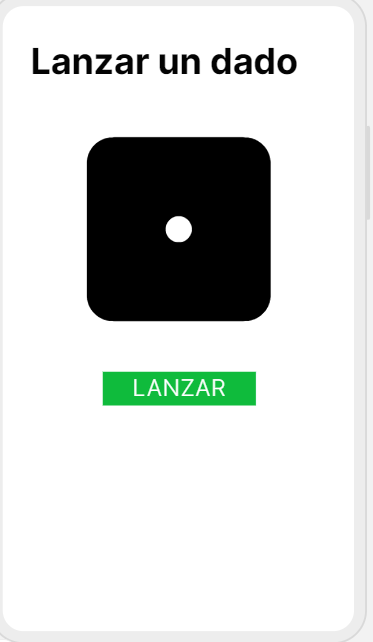


Ilustración 16: Pantalla "Lanzar un dado"

En la sección de Sorteos, al pulsar en la tarjeta de “Cara o cruz”, llevará al usuario a la respectiva pantalla,

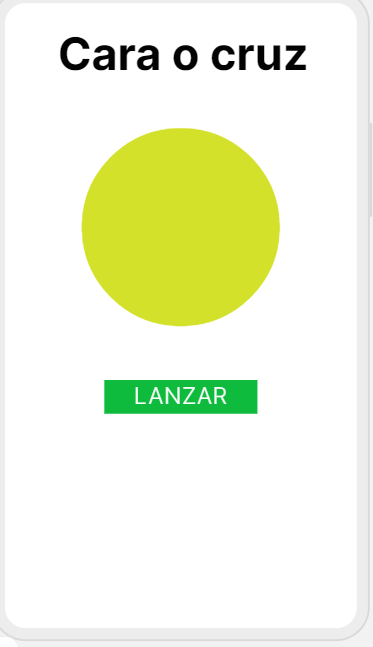


Ilustración 17: Pantalla "Cara o Cruz"

**Aplicación web:**

El header de la aplicación contiene el logo y nombre de la aplicación, un menú con submenú para el apartado de sorteos y sus subapartados.

En caso de no tener iniciada sesión, en el header aparecerán unos botones para iniciar sesión o registrarse.



Ilustración 18: Header (Sin sesión iniciada)

En caso de haber una sesión iniciada, en el header aparecerá información de usuario y un botón para cerrar la sesión.



Ilustración 19: Header (Con sesión iniciada)

Al acceder a la aplicación web, la página principal mostrará las secciones disponibles en formato de tarjeta.



Ilustración 20: Página principal de la app web

En caso de ver la página principal desde un navegador en un dispositivo móvil, la página se adaptará ya que es responsiva.

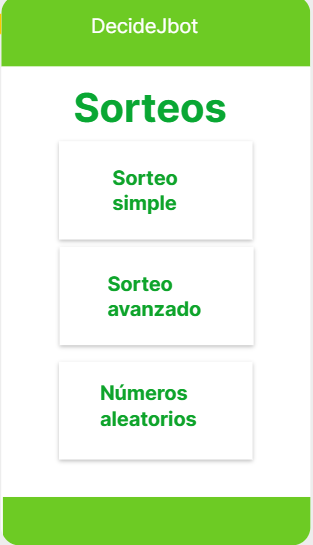


Ilustración 21: Página principal de la app web vista en móvil

En la página principal, en caso de tener sesión iniciada y que el usuario haya realizado sorteos y disponga de resultados guardados externamente, aparecerá la tarjeta “Historial de sorteos”



Ilustración 22: Pantalla principal con historial de sorteos

Si el usuario accede a la página de sorteo simple, verá el siguiente contenido:



Ilustración 23: Página "Sorteo Simple"

En la página del sorteo simple, en caso de que el usuario no haya completado los campos o alguno sea inválido, aparecerá una ventana modal con un mensaje de error.



Ilustración 24: Página "Sorteo Simple" con error

En la página de Sorteo Simple, si los campos se han completado correctamente, al pulsar el botón “Sortear” se generarán los resultados.



Ilustración 25: Resultados del Sorteo Simple

En la página de Sorteo Simple, si se activa el contador, al pulsar el botón “Sortear”, aparecerá un contador de 5 segundos que irá disminuyendo, acompañado de un texto de carga.



Ilustración 26: Resultados del Sorteo Simple

Si el usuario accede mediante el menú del header o mediante la página inicial al sorteo avanzado, lo primero que verá será lo siguiente:

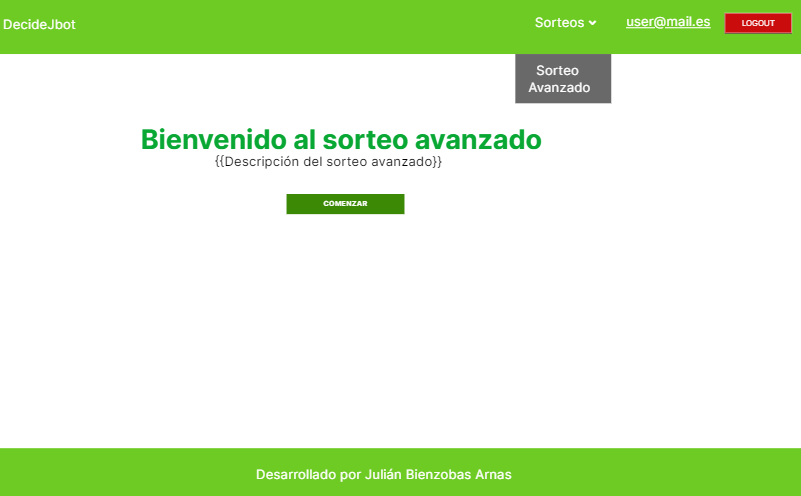


Ilustración 27: Página de bienvenida al Sorteo Avanzado

Cuando el usuario pulsa el botón “Comenzar” en la bienvenida del Sorteo Avanzado, aparece lo siguiente:



Ilustración 28: Paso 1 del Sorteo Avanzado

Cuando el usuario pulsa el botón “Participantes” en el paso 1 del Sorteo Avanzado, aparece el paso 2, correspondiente a los participantes:



Ilustración 29: Paso 2 del Sorteo Avanzado

Si el usuario completa correctamente el paso 2 del Sorteo Avanzado y pulsa el botón “Premios”, aparece el paso 3, correspondiente a los premios:



Ilustración 30: Paso 3 del Sorteo Avanzado



Ilustración 31: Paso 4 del Sorteo Avanzado

Cuando el usuario pulse el botón “Finalizar” del paso 4 del sorteo avanzado, aparecerán todos los datos del sorteo con botones para volver a pasos anteriores o confirmar y realizar el sorteo.



Ilustración 32: Detalles del Sorteo Avanzado

Cuando el usuario pulse el botón de “Sortear” en el Sorteo Avanzado aparecerá un enlace con un botón para copiar este enlace.

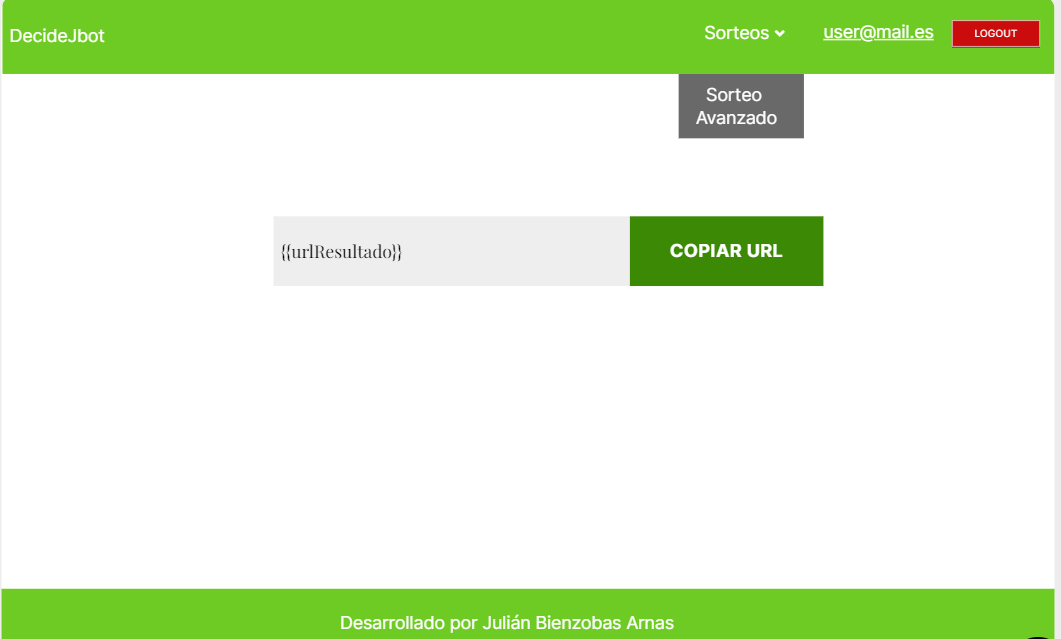


Ilustración 33: Enlace al resultado del Sorteo Avanzado

Cualquier usuario que acceda al enlace generado del resultado en caso de que la fecha programada no se haya alcanzado y los resultados no se hayan generado, verá lo siguiente:



Ilustración 34: Detalles del Sorteo Avanzado

En caso de que el usuario que acceda al enlace generado del resultado sea el usuario que creó el sorteo y no se haya alcanzado la fecha programada ni se hayan generado los resultados, se mostrará lo siguiente:

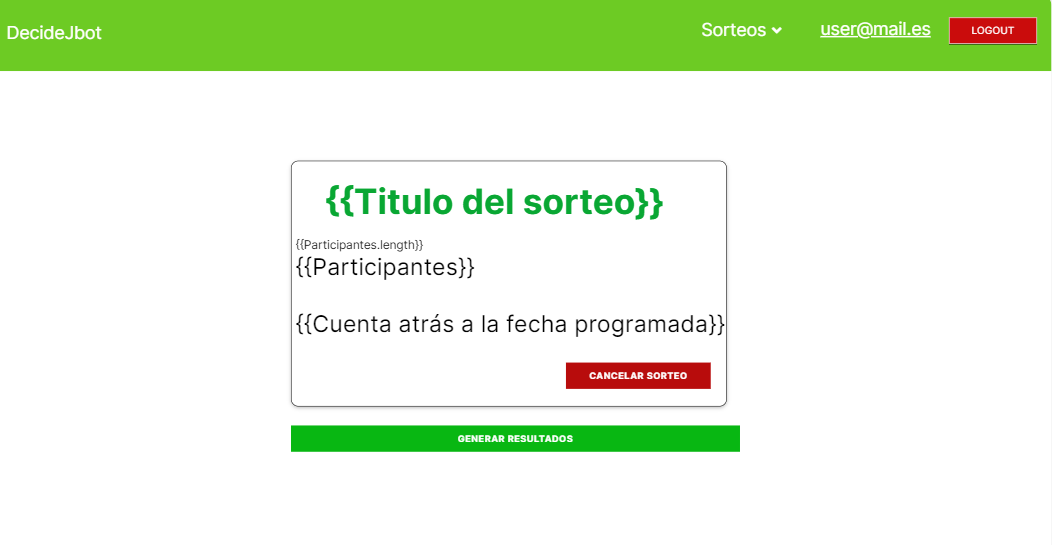


Ilustración 35: Sorteo programado del usuario logueado

En caso de acceder al resultado y que los resultados ya estén generados, en caso de no ser un resultado creado por el usuario logueado, se mostrará de la siguiente manera:



Ilustración 36: Página de un resultado

En caso de acceder a un resultado ya generado, y que pertenezca al usuario logueado, se mostrará de la siguiente manera:



Ilustración 37: Página de un resultado del usuario logueado

Si el usuario accede al “Historial De Sorteos” desde la página principal, si tiene algún resultado generado, podrá observar lo siguiente:



Ilustración 38: Página "Historial De Sorteos"

Si el usuario accede al “Generador de Números Aleatorios” verá la siguiente página:

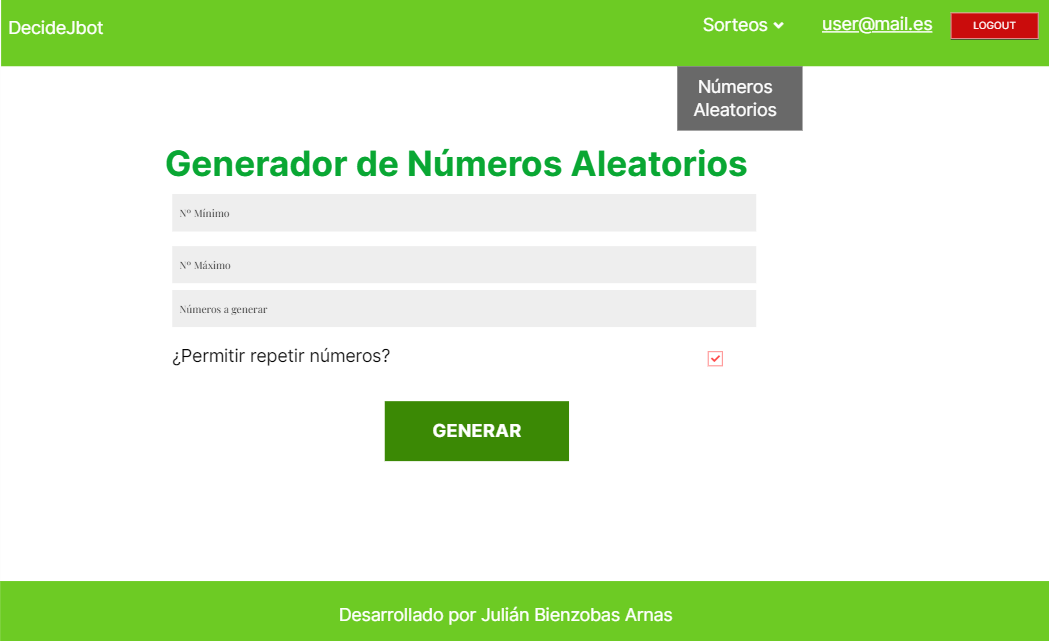


Ilustración 39: Página "Generador de Números Aleatorios"

Si el usuario completa correctamente todos los campos en el Generador de Números Aleatorios y pulsa el botón “Generar”, se mostrarán los números generados:

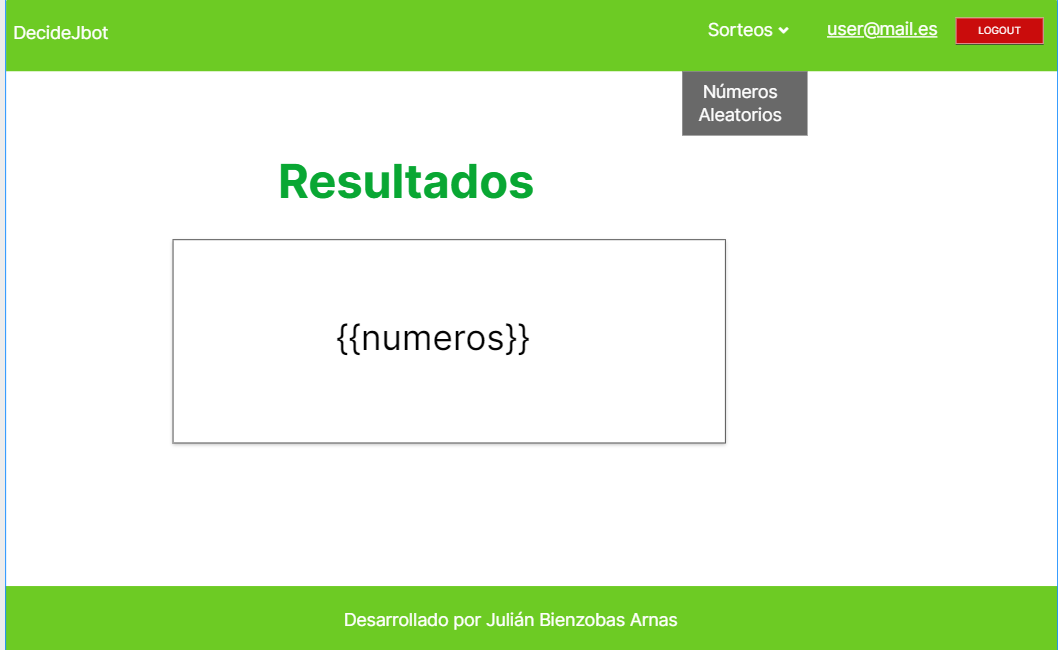


Ilustración 40: Resultados en la página "Generador de Números Aleatorios"

Si el usuario pulsa el botón de iniciar sesión, se le mostrará la página de inicio de sesión:

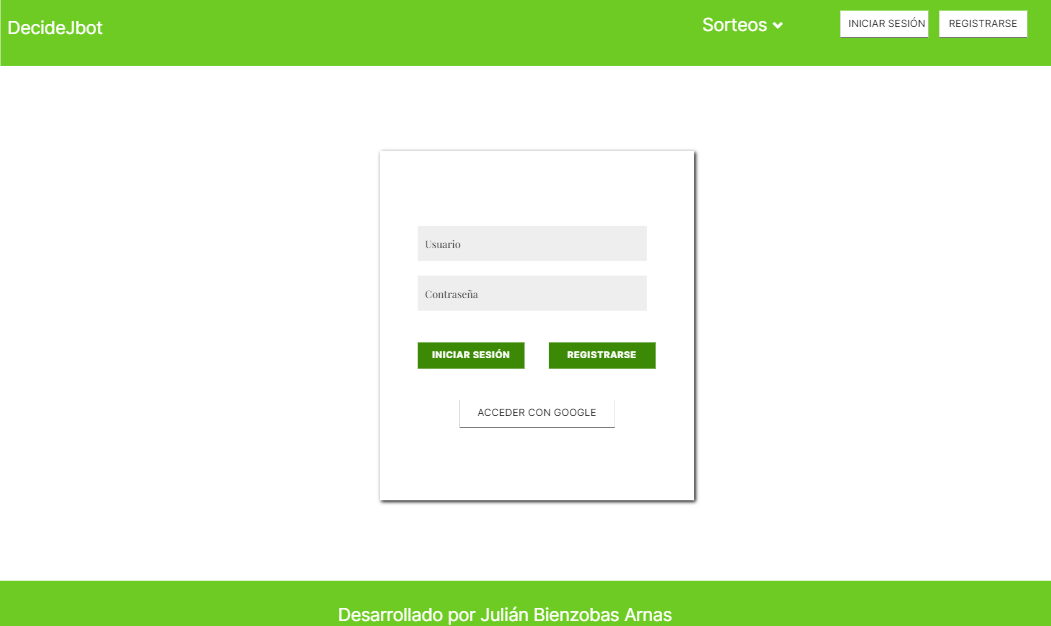


Ilustración 41: Página de inicio de sesión

Si el usuario pulsa el botón de registrarse, se le mostrará la página de registro:

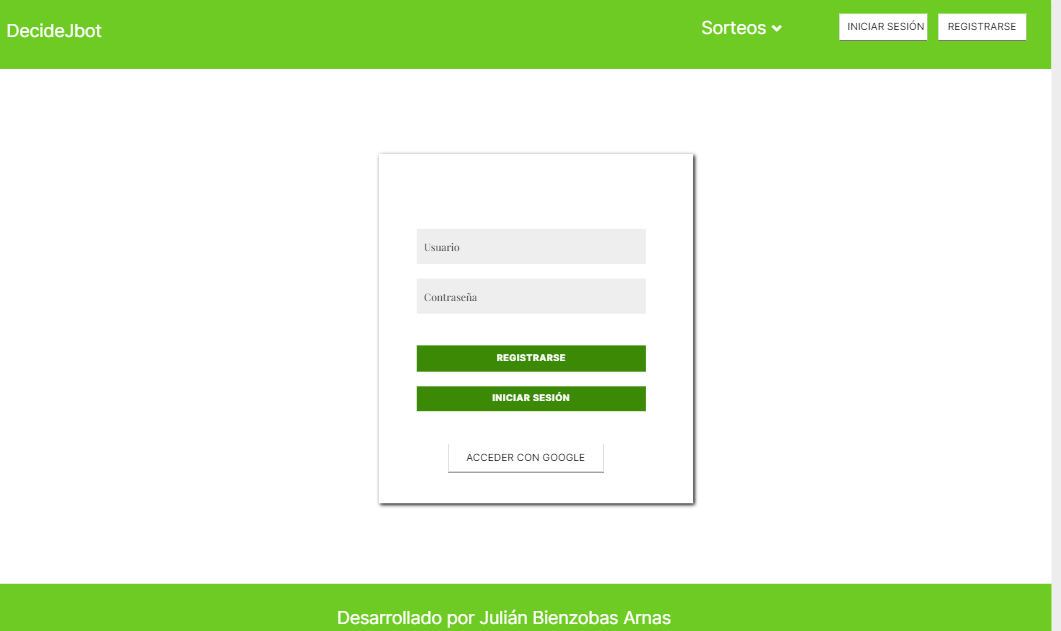


Ilustración 42: Página de registro de usuario

Una vez pensada la idea del diseño de la interfaz de usuario de las aplicaciones, y teniendo un primer diseño en mockups, se procede al diseño real de las interfaces de usuario de las aplicaciones, quedando como se muestra a continuación.

**Aplicación móvil:**



Ilustración 43: Splash Screen

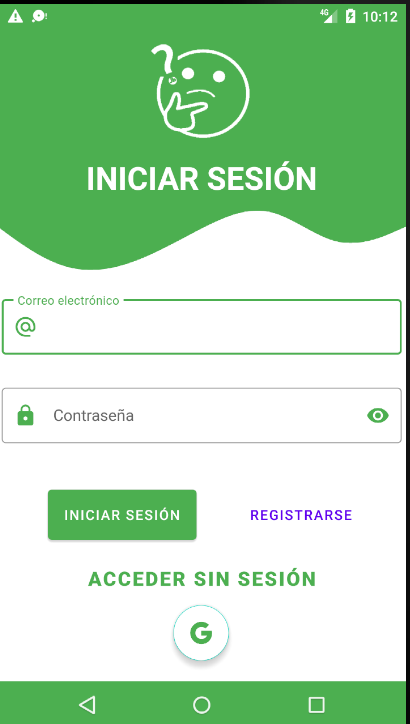


Ilustración 44: Pantalla de inicio de sesión

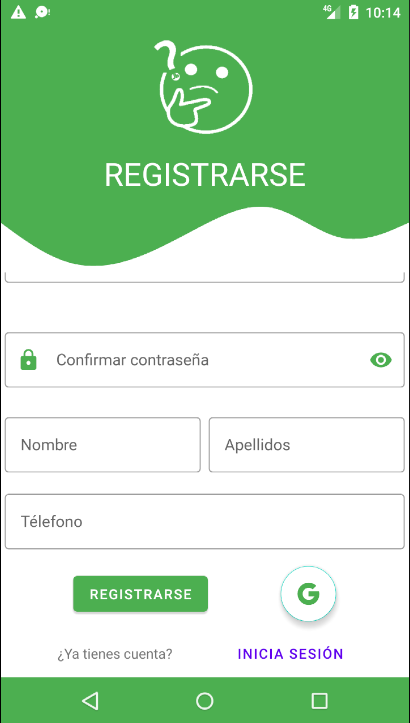


Ilustración 45: Pantalla de registro

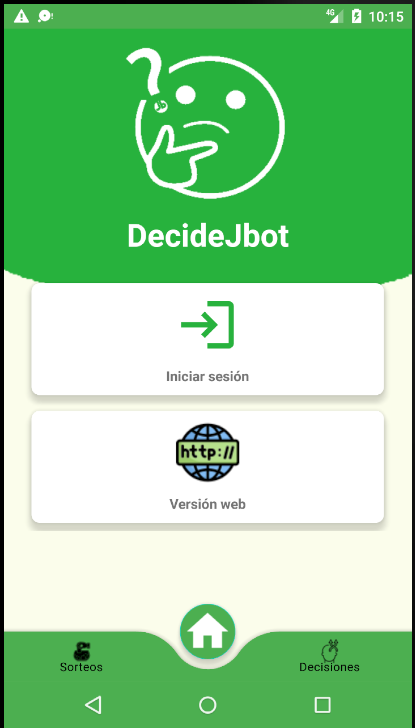


Ilustración 46: Pantalla de inicio



Ilustración 47: Pantalla de sorteos

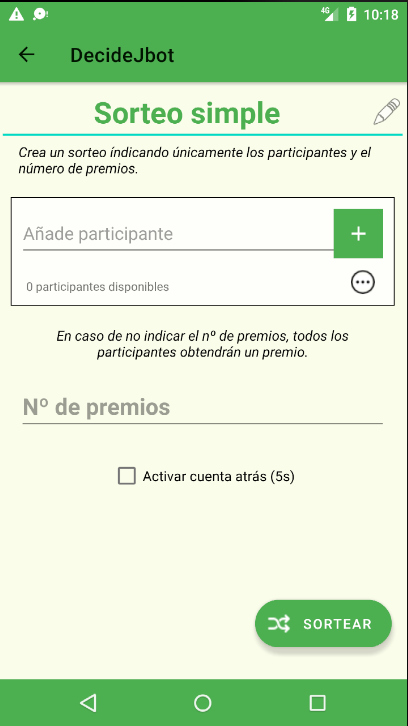


Ilustración 48: Pantalla de Sorteo Simple

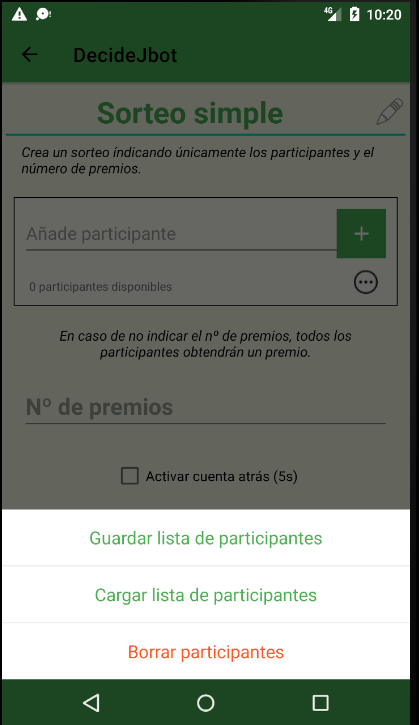


Ilustración 49: Opciones del Sorteo Simple

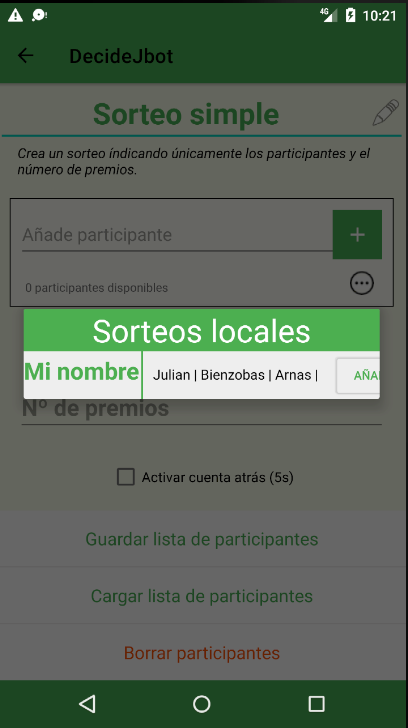


Ilustración 50: Cargar sorteo en el Sorteo Simple

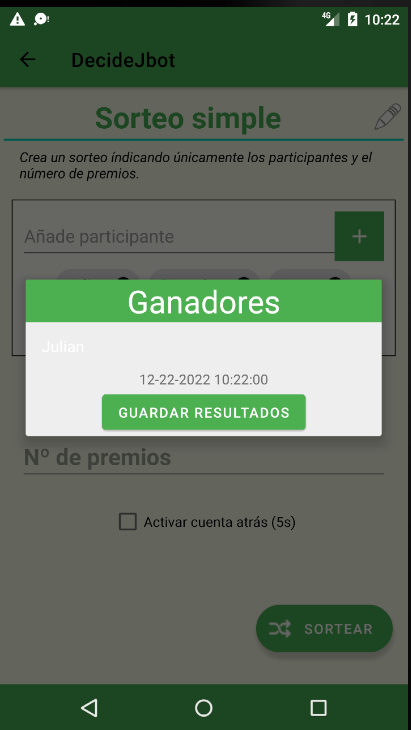


Ilustración 51: Resultados del Sorteo Simple

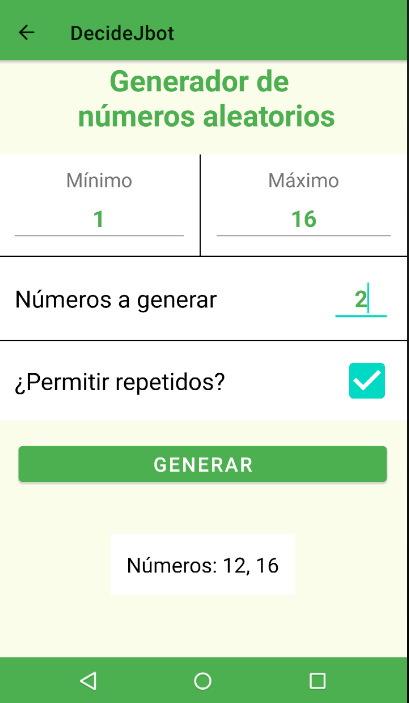


Ilustración 52: Generador de Números Aleatorios

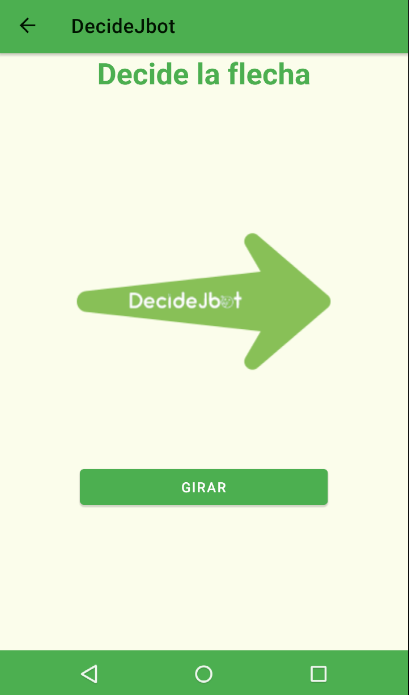


Ilustración 53: Pantalla de Girar una Flecha

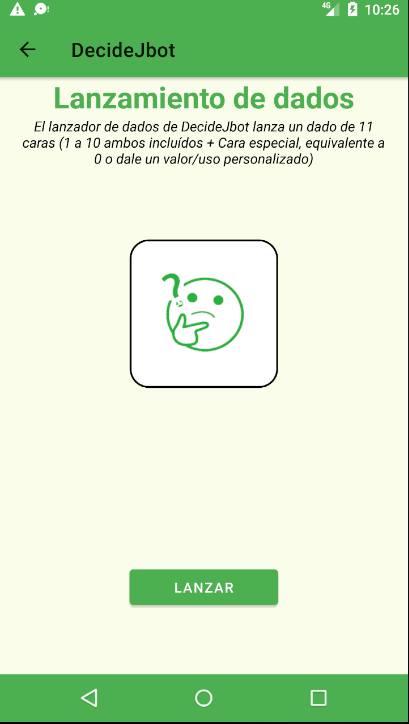


Ilustración 54: Pantalla de Lanzar un Dado



Ilustración 55: Pantalla Cara o Cruz

**Aplicación web:**

****

Ilustración 56: Página de inicio (Sin sesión iniciada)



Ilustración 57: Página de inicio (Con sesión iniciada)



Ilustración 58: Página "Sorteo Simple"



Ilustración 59: Página "Sorteo Simple" (Con datos)



Ilustración 60: Opciones para el Sorteo Simple



Ilustración 61: Opciones de importación en el Sorteo Simple



Ilustración 62: Opciones de exportación en el sorteo simple

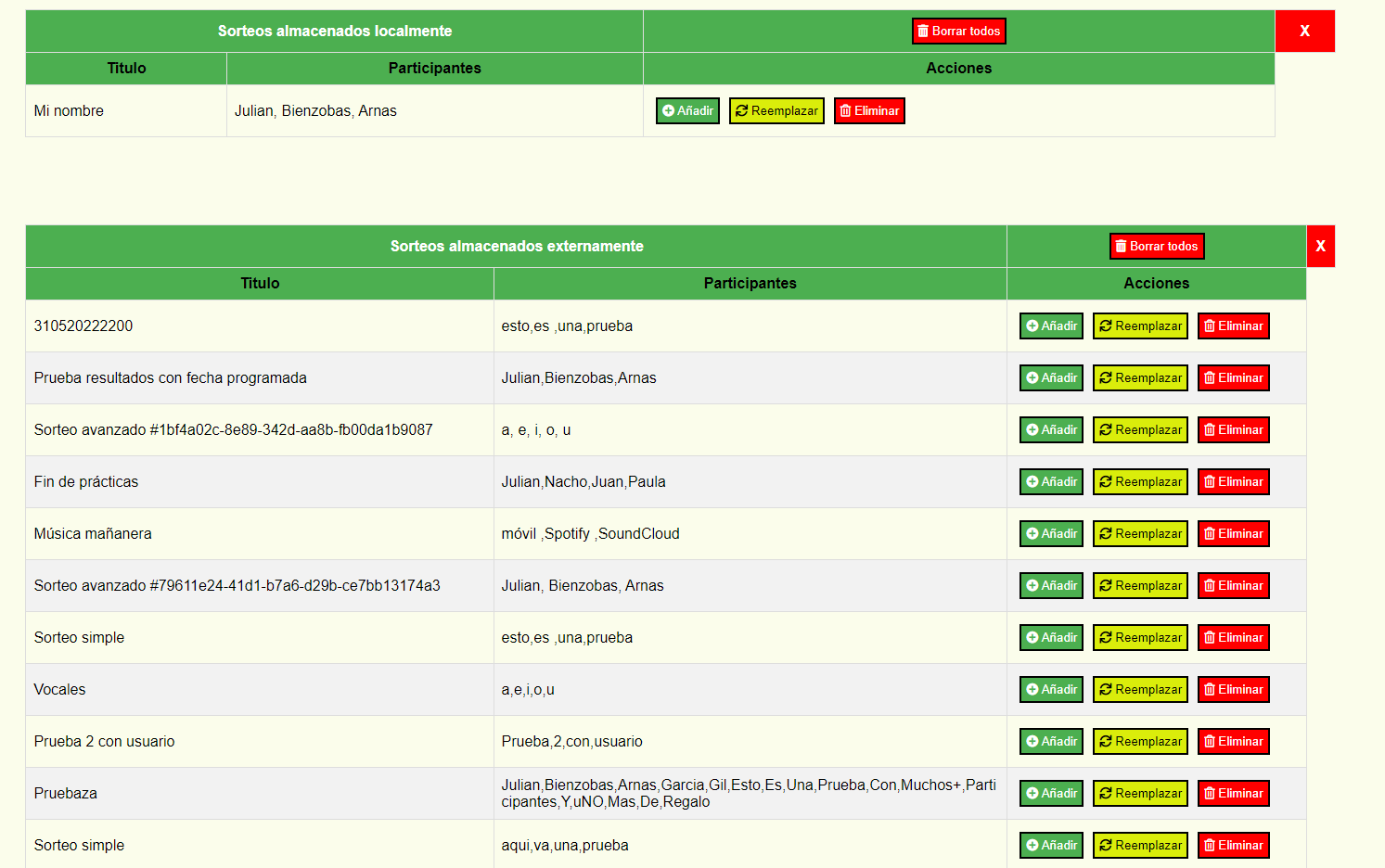


Ilustración 63: Importación de participantes al Sorteo Simple

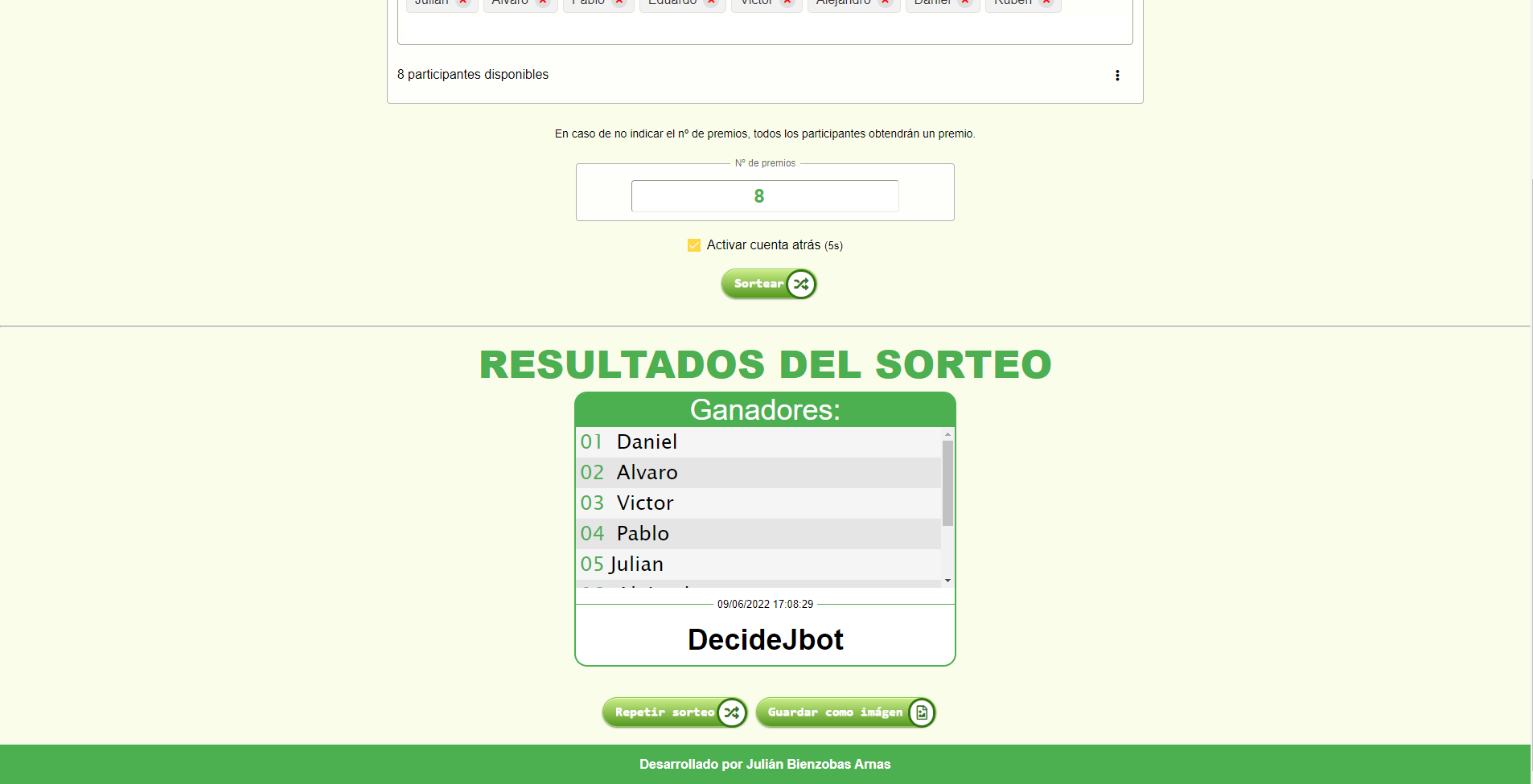


Ilustración 64: Resultados en Sorteo Simple



Ilustración 65: Inicio al Sorteo Avanzado



Ilustración 66: Paso 1 del Sorteo Avanzado



Ilustración 67: Paso 2 del Sorteo Avanzado



Ilustración 68: Paso 3 del Sorteo Avanzado

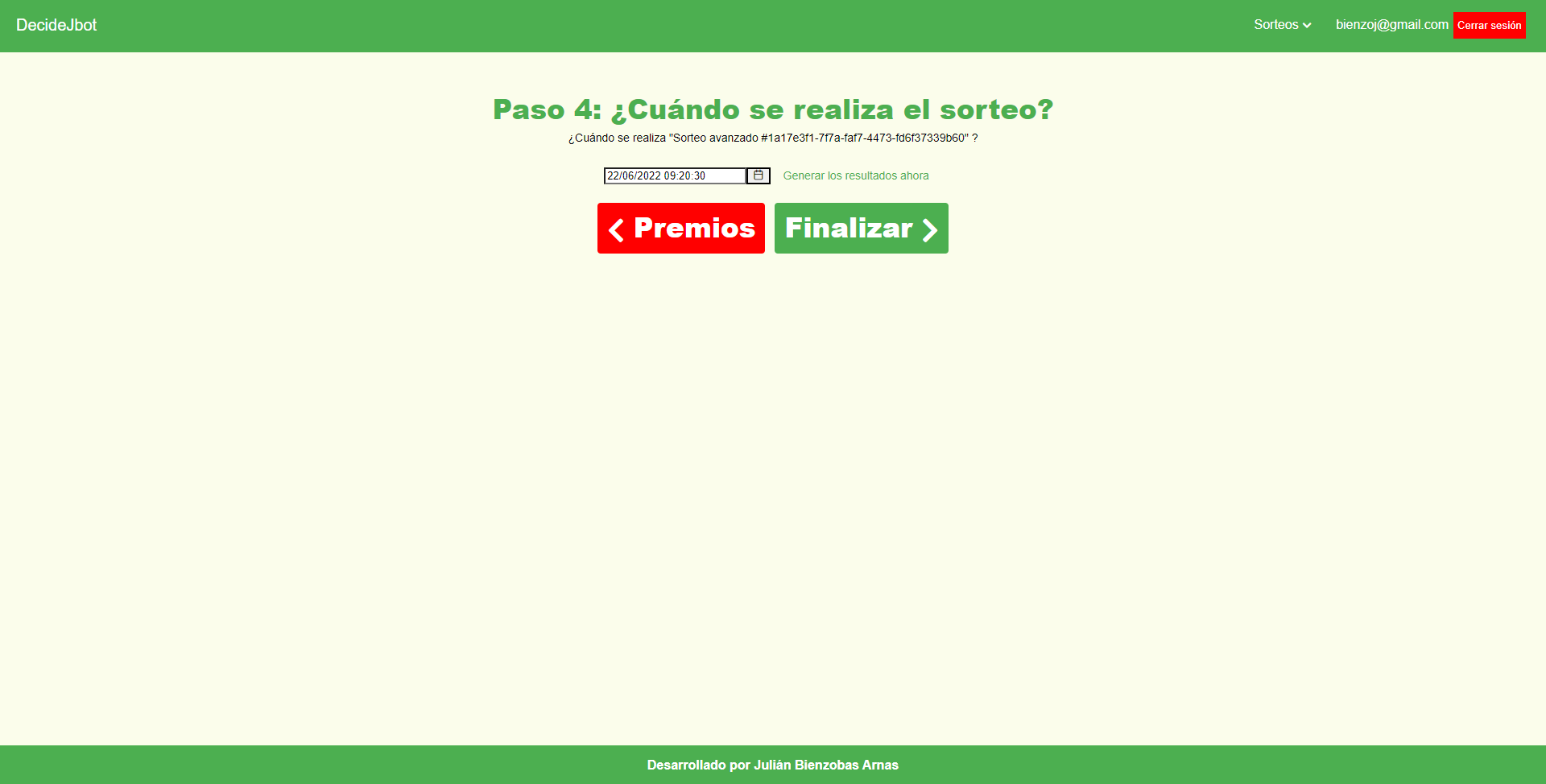


Ilustración 69: Paso 4 del Sorteo Avanzado



Ilustración 70: Detalles del Sorteo Avanzado



Ilustración 71: URL a un resultado



Ilustración 72: Resultado programado del usuario logueado



Ilustración 73: Resultado generado (Del usuario logueado)



Ilustración 74: Historial de Sorteos



Ilustración 75: Resultado de otro usuario



Ilustración 76: Generador de Números Aleatorios



Ilustración 77: Página de Inicio de Sesión

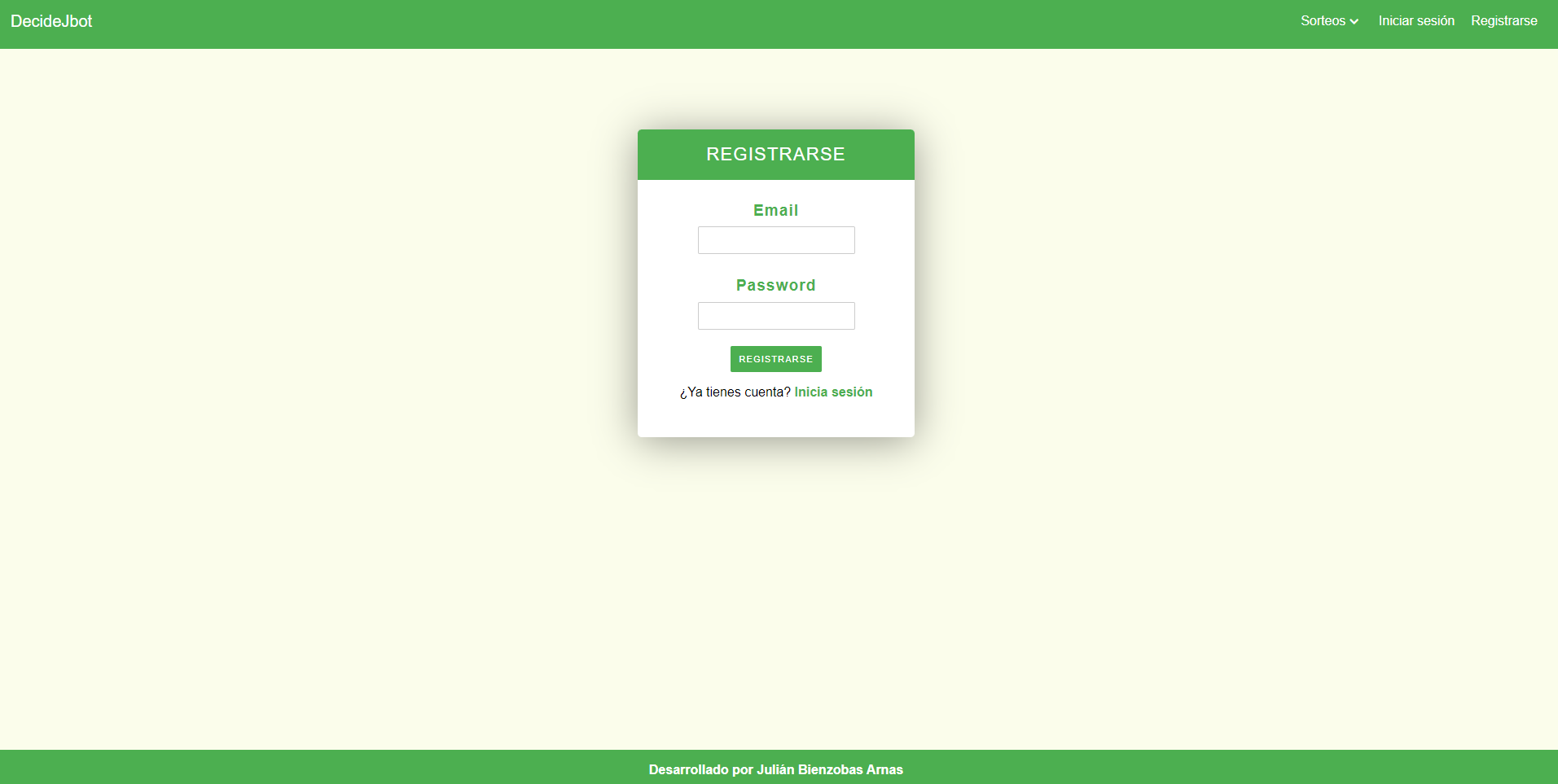


Ilustración 78: Página de Registro

## **Análisis y diseño del sistema funcional**

En este apartado, se describe el funcionamiento que deberá realizar cada componente de las aplicaciones, para que el Developer lo programe y haga la aplicación funcional.

**App móvil:**

Al pasar 5 segundos de la pantalla de carga (Splash Screen), en caso de que no haya una sesión iniciada, aparecerá la pantalla de inicio de sesión:

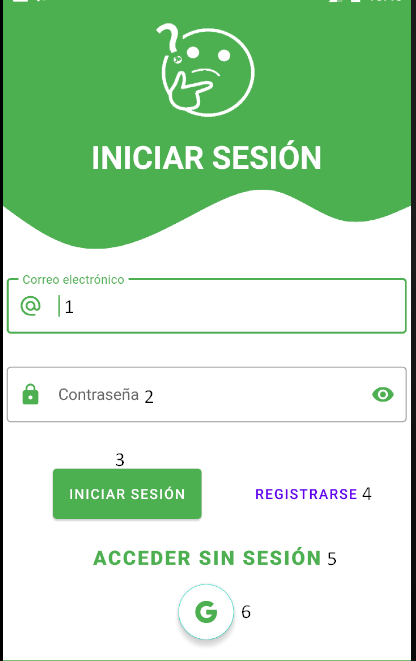


Ilustración 79: Pantalla de Inicio de Sesión

1. El usuario debe introducir su correo electrónico.
2. El usuario debe introducir su contraseña.
3. Al pulsarlo, si los campos 1 y 2 son correctos, se redirige a la pantalla principal. En caso contrario, aparecen mensajes de error.
4. Al pulsarlo, se redirige a la pantalla de registro.
5. Al pulsarlo, se redirige a la pantalla principal, pero accediendo sin sesión de usuario.
6. El usuario seleccionará una cuenta vinculada a Google con la que utilizar la aplicación.

En la pantalla de registro:



Los campos obligatorios son el 1, 2, 3,4 y 5, siendo el 6 el campo optativo.

1. Al pulsar el botón, en caso de que los datos estén correctamente el usuario se registrará y será redirigido a la pantalla inicial. En caso contrario, aparecerán los mensajes de error necesarios.
2. El usuario accederá a la aplicación con su cuenta de Google.
3. Se redirige al usuario a la pantalla de inicio de sesión.

En la pantalla de inicio:

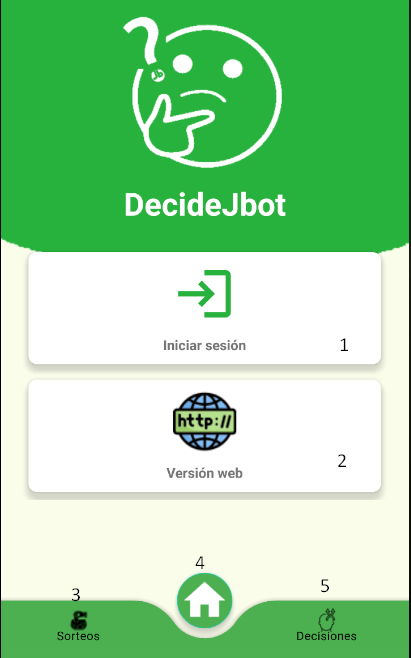


Ilustración 80: Pantalla de inicio

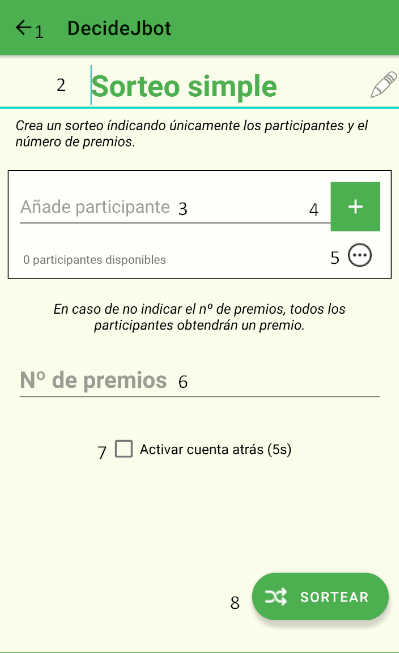
1. En caso de tener iniciada sesión, al pulsarlo se cerrará la sesión. En caso de no tener iniciada sesión, al pulsarlo redirigirá al usuario a la pantalla de inicio de sesión.
2. En caso de tener conexión, redirigirá al usuario a la aplicación web.
3. Muestra la sección de “Sorteos”.
4. Muestra la página principal.
5. Muestra la sección de “Decisiones”.

En la sección de Sorteos:

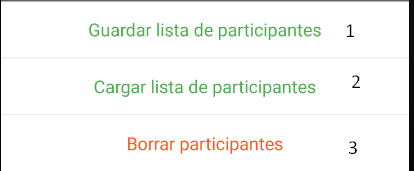


Al pulsar cada tarjeta, redirigirá a su respectiva pantalla.

Sorteo simple:



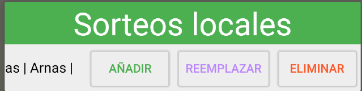
1. Redirige al usuario a la pantalla de inicio.
2. Introducir el título/Descripción del sorteo.
3. Introducir el valor del participante a añadir.
4. Al pulsarlo, en caso de no estar en la lista de participantes del sorteo, se añadirá el participante a esta.
5. Se abre un menú de opciones para el sorteo:



* 1. Aparece un menú para decidir donde guardar el sorteo:



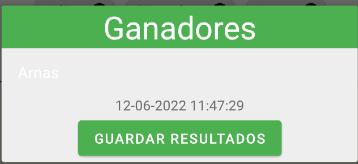
* 1. Se cargan y se muestran los sorteos almacenados:



* Añadir: Añade a la lista de participantes del sorteo actual los participantes del sorteo seleccionado.
* Reemplazar: Reemplaza los participantes del sorteo actual por los del sorteo seleccionado.
* Eliminar: Elimina el sorteo seleccionado del almacenamiento.

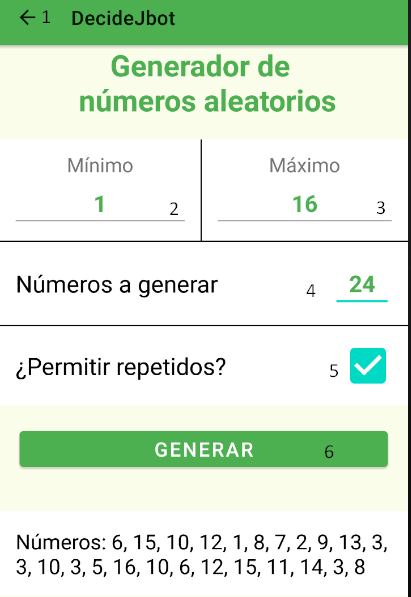
5.3 Vacía la lista de participantes del sorteo actual.

1. Se indica el número de premios que se desean repartir en el sorteo, o se deja vacío para que todos los participantes reciban premios.
2. Si se activa, al realizar el sorteo aparecerá un diálogo de carga.
3. Si todos los campos están correctos, se realiza el sorteo, mostrando un cuadro de diálogo con los resultados del sorteo.



En caso de pulsar el botón “Guardar resultados”, el resultado se almacenará externamente y la aplicación lanzará una notificación en la que, si el usuario la pulsa, será redirigido a la aplicación web, mostrándose el resultado.

En el generador de números aleatorios:



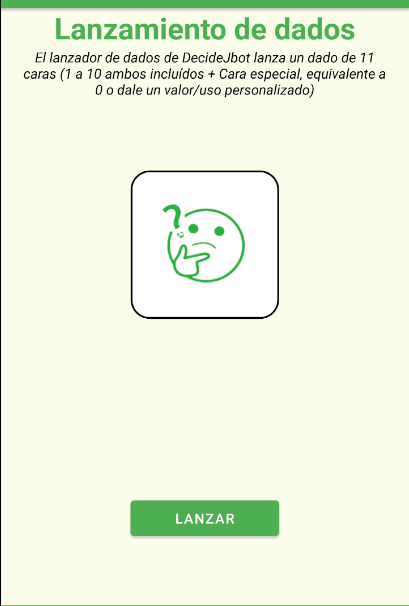
1. Si se pulsa, se redirige al usuario a la página principal.
2. Se introducirá un número que deberá ser menor al del campo 3.
3. Se introducirá un número que deberá ser mayor al del campo 4.
4. Se indicará cuantos números quieren generarse, teniendo en cuenta el campo 2, 3 y 5.
5. En caso de seleccionarse, se permitirá generar números repetidos.
6. Si todos los campos están correctos y teniendo en cuenta los valores de estos, se generarán un listado de números aleatorios.

La sección de girar una flecha:



Al pulsar en la flecha o en el botón, la flecha girará durante unos segundos, deteniéndose en una posición aleatoria.

La sección de “Lanzar un dado”:



Al pulsar sobre el dado o sobre el botón se asignará un valor aleatorio de 0 a 10, imprimiendo el respectivo valor en un texto y en una imagen sobre el dado.

La sección de “Cara o Cruz”:



Al pulsar sobre la moneda o sobre el botón, se asignará un valor booleano aleatorio, asignando estos valores a cara o cruz e imprimiéndolos en un texto y en una imagen sobre la moneda.

**App web:**

* Header:



Ilustración 81: Header

1. Al pulsar el logo, se redirigirá al usuario a la pantalla principal.
2. Despliega el submenú de sorteos, mostrando los subapartados de este.
3. (Aparece en caso de no tener la sesión iniciada). Redirige al usuario al Login.
4. (Aparece en caso de no tener la sesión iniciada). Redirige al usuario a la página de registro.

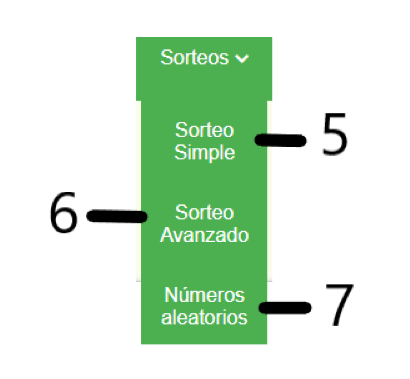


Ilustración 82: Menú "Sorteos" en el Header

1. Redirige al usuario a la página de “Sorteo Simple”
2. Redirige al usuario a la página de “Sorteo Avanzado”
3. Redirige al usuario a la página de “Números Aleatorios”



1. (Solo se muestra si hay sesión iniciada). Cierra la sesión de usuario.

* Página principal:



Ilustración 83: Index

1, Redirige al usuario a la página de “Sorteo Simple”

2. Redirige al usuario a la página de “Sorteo Avanzado”

3. Redirige al usuario a la página “Generador de Números Aleatorios”

4. (Solo aparece si hay sesión iniciada y si el usuario ha realizado sorteos avanzados y tiene resultados guardados externamente). Redirige al usuario a la página “Historial de sorteos”.

* Sorteo simple (SS) :



Ilustración 84: Página "Sorteo Simple"

1. Al pulsar el icono de editar (Lápiz) o dar doble clic en el título del sorteo, este se convertirá en un input que nos permitirá editar su valor.
2. Input para introducir el valor del nuevo participante. Al pulsar la tecla “Enter” el valor del campo se añade a la lista de participantes y mostrándose junto a los demás participantes a la izquierda del input. En caso de que esté vacío, no añadirá nada, y si seleccionamos la tecla “Backspace”, en caso de estar vacío el campo y haber participantes en la lista, se eliminará el ultimo de la lista. Al añadir un participante el contador se actualiza, indicando el tamaño de la lista de participantes.
3. Al pulsarlo, se mostrará un menú de opciones para realizar acciones sobre el sorteo.



Ilustración 85: Menú de opciones del Sorteo Simple

3.1 (Aparece si hay un participante o más). Copia la lista de participantes al portapapeles.

3.2 (Aparece siempre)

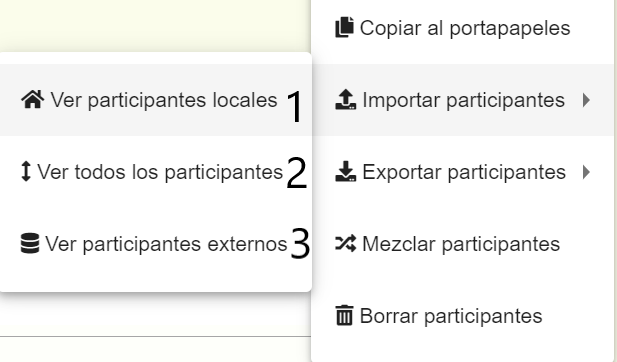


Ilustración 86: Menú importar participantes al Sorteo Simple

3.2.1. En caso de tener sorteos guardados localmente, se mostrarán en una tabla. (El almacenamiento local se realiza con LocalStorage).

3.2.2 En caso de haberlos, muestra en tablas los sorteos almacenados localmente y externamente. (Para mostrar los externos requiere inicio de sesión).

3.3.3 En caso de tener sesión iniciada y disponer de sorteos almacenados externamente, los muestra en una tabla.



Ilustración 87: Sorteos almacenados

1. Elimina todos los sorteos mostrados en la tabla (Solicita confirmación mediante un modal).
2. Cierra y deja de mostrar la tabla.
3. Añade los participantes de la fila a la lista de participantes del sorteo actual.
4. Reemplaza los participantes del sorteo actual por los participantes de la celda.
5. Elimina el sorteo seleccionado, solicitando confirmación mediante un modal.

3.3 (Aparece si hay por lo menos un participante)

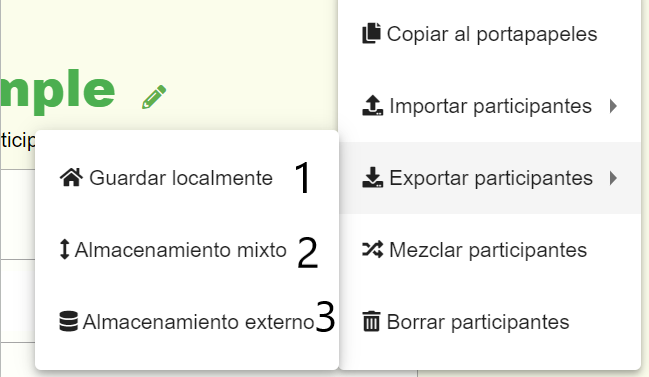


Ilustración 88: Opción de exportar sorteos simples

3.3.1. Guarda el sorteo (Título y participantes) localmente con LocalStorage.

3.3.2 Almacena el sorteo localmente y, en caso de tener sesión iniciada también externamente.

3.3.3 (Requiere inicio de sesión). Guarda el sorteo en la base de datos externa de Firestore de Firebase.

3.4 (Aparece si hay un participante o más). Desordena la lista de participantes aleatoriamente.

3.5 (Aparece si hay por lo menos un participante). Vacía la lista de participantes.

1. Se introducirá el número de ganadores que va a tener el sorteo. No debe ser mayor al número de participantes o aparecerá un diálogo de error. En caso de estar vacíos, se detectará el número de participantes y será el número de premios a repartir.
2. En caso de estar activado, aparecerá una cuenta atrás de 5 segundos antes de generarse los resultados. En caso contrario, se generarán directamente.
3. Al pulsarlo, en caso de que haya participantes se generarán los resultados.

Al generarse los resultados, se muestra lo siguiente:



Ilustración 89: Resultados de un sorteo simple

1. Genera nuevos resultados del sorteo.
2. Descarga y guarda el bloque de ganadores como una imagen.

* Sorteo avanzado (SA):

Al acceder a la página de sorteo avanzado primero se muestra una descripción de la sección.



Ilustración 90: Bienvenida al Sorteo Avanzado

Al pulsar el botón “Comenzar”, nos muestra en la misma página lo siguiente:



Ilustración 91: Paso 1 del Sorteo Avanzado

1. Input asociado al título del sorteo. En caso de quedarse vacío, se le dará un valor automático.
2. Al pulsarlo, se muestra el siguiente paso:



Ilustración 92: Paso 2 del Sorteo Avanzado

La función de este paso es idéntica a la función de añadir participantes en el Sorteo simple.

En caso de pulsar el botón “Título” se mostrará el paso anterior y, en caso de pulsar el botón “Premios”, se muestra el siguiente paso:



Ilustración 93: Paso 3 del Sorteo Avanzado

La función de este paso es idéntica a la función de indicar el nº de premios en el Sorteo Simple.

En caso de pulsar el botón “Participantes”, se muestra el paso anterior y, en caso de pulsar el botón “Fecha”, se muestra el siguiente paso:

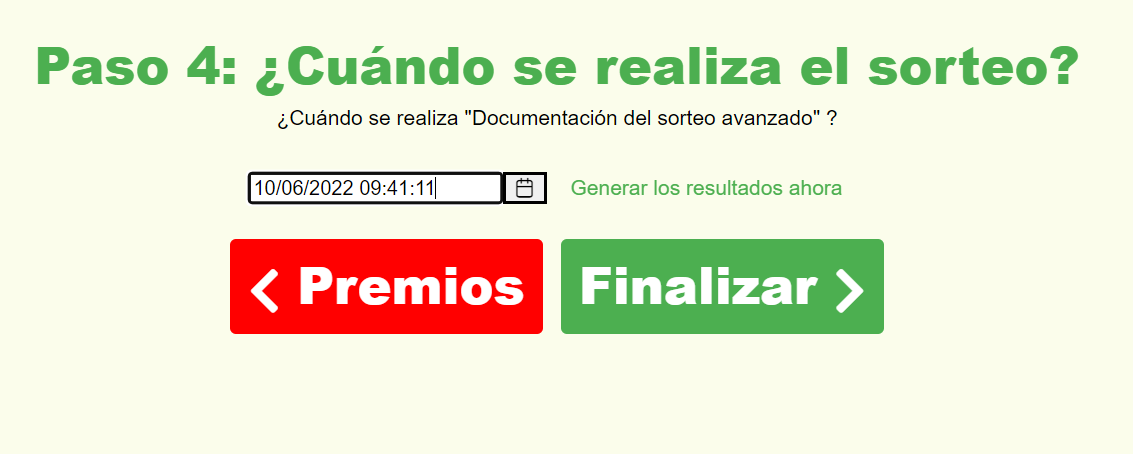
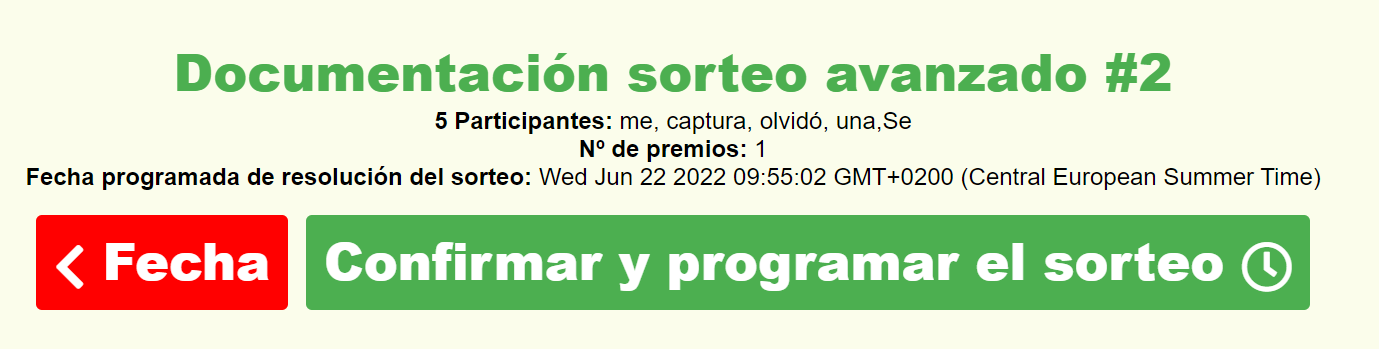


Ilustración 94: Paso 4 del Sorteo Avanzado

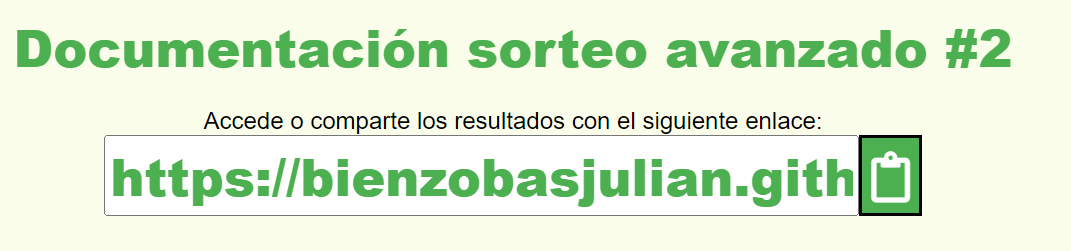
Se introduce la fecha en la que se quieren generar los resultados del sorteo.

En caso de pulsar “Generar los resultados ahora” el sorteo estará programado para la fecha actual y se mostrará el siguiente paso.

En caso de pulsar el botón “Premios” se mostrará el paso anterior y, en caso de pulsar el botón “Finalizar” se establecerá la fecha programada indicada al sorteo y se mostrará el siguiente paso:

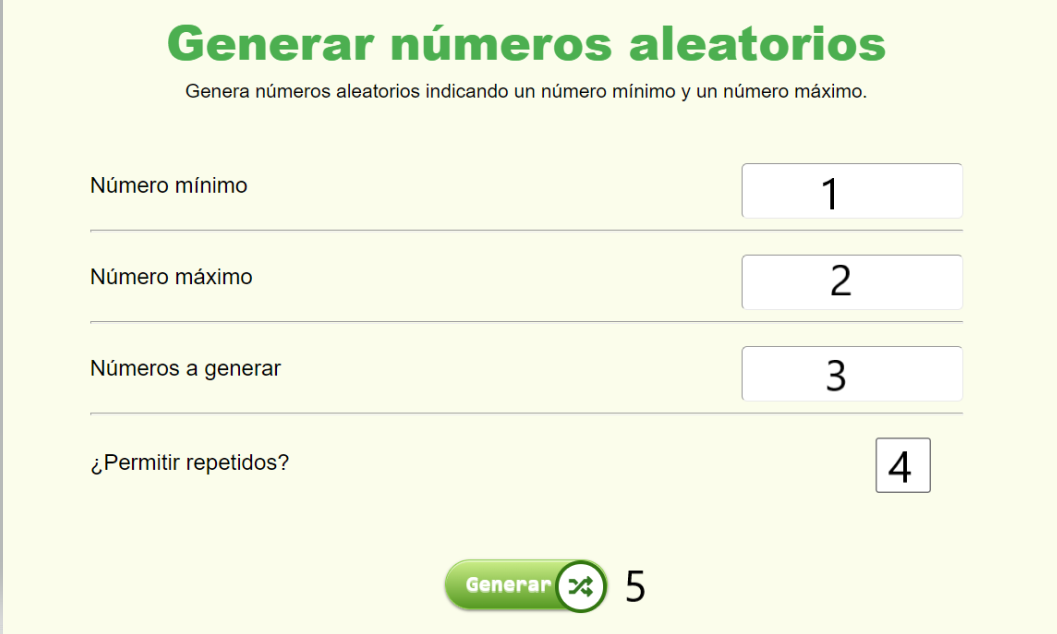


En caso de pulsar el botón “Fecha”, se mostrará el paso anterior y, en caso de pulsar el botón “Confirmar” se mostrará lo siguiente:



Se muestra el enlace en el que se generarán o se mostrarán los resultados, permitiéndole al usuario copiarlo para acceder a él o compartirlo.

* Generador de Números Aleatorios (GNA):



1. Se introducirá un número (Debe ser menor al del campo 2).
2. Se introducirá un número (Debe ser mayor al del campo 1).
3. Se introducen cuantos números se van a generar. Para este campo hay que tener en cuenta los campos 1,2 y 4.
4. Activar/desactivar el permitir generar números repetidos.
5. Tendrá en cuenta los campos aleatorios y en relación a ellos generará un listado de números que se mostrarán de la siguiente manera:



* Historial de sorteos (HS):



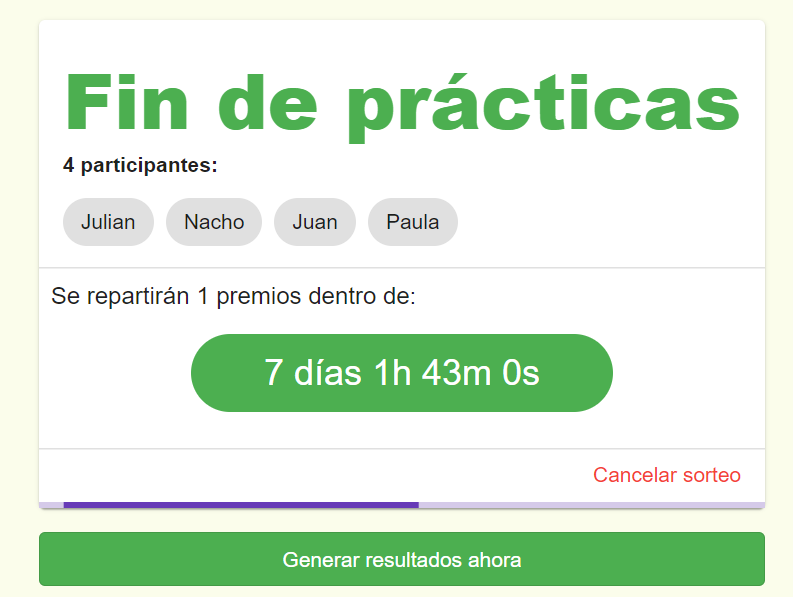
Solo se podrá acceder con sesión iniciada y en caso de que el usuario tenga resultados almacenados externamente.

Se obtendrán y mostrarán todos los resultados almacenados del usuario logueado. Al seleccionar uno de ellos, se enviará al usuario a la página del resultado correspondiente.

* Resultado:

El usuario podrá acceder a un resultado desde el historial de sorteos o mediante su correspondiente enlace.

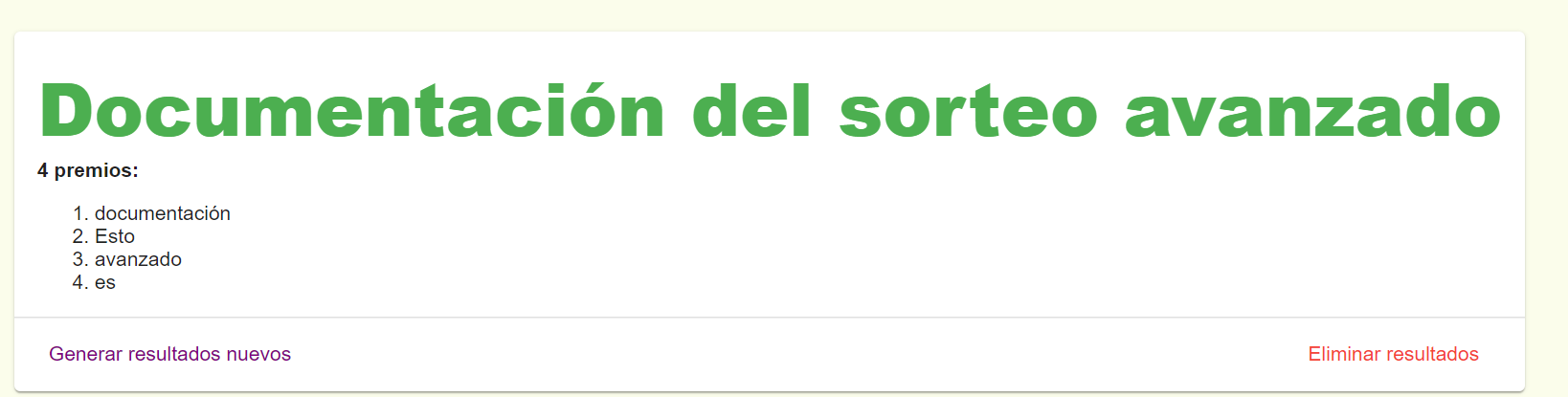
En caso de que el sorteo esté programado para una fecha aun no alcanzada, y pertenezca al usuario logueado:



En caso de seleccionar “Cancelar Sorteo”, aparecerá un modal de confirmación, y en caso afirmativo, el resultado se eliminará de la base de datos externa y se redirigirá al usuario a la página de inicio, ya que el enlace quedará inservible.

Si se selecciona “Generar resultados ahora”, se generarán, mostrarán y guardarán los resultados del sorteo.

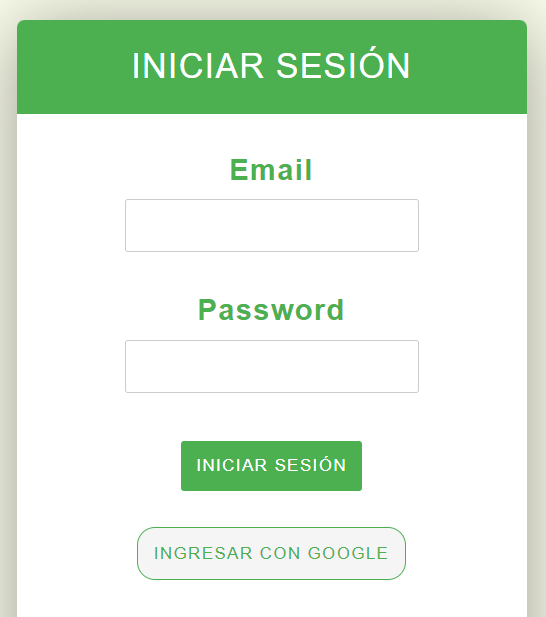
En caso de acceder a un resultado perteneciente a usuario logueado y que ya se han generado sus resultados:



En caso de seleccionar “Generar nuevos resultados”, como el nombre del botón índica, se generarán unos nuevos resultados sobre el sorteo asociado al resultado, mostrándose los neuvos resultados y actualizando el resultado en la base de datos externa.

En caso de seleccionar “Eliminar resultados”, se solicitará confirmación mediante un modal y, en caso afirmativo, el resultado se eliminará de la base de datos externa.

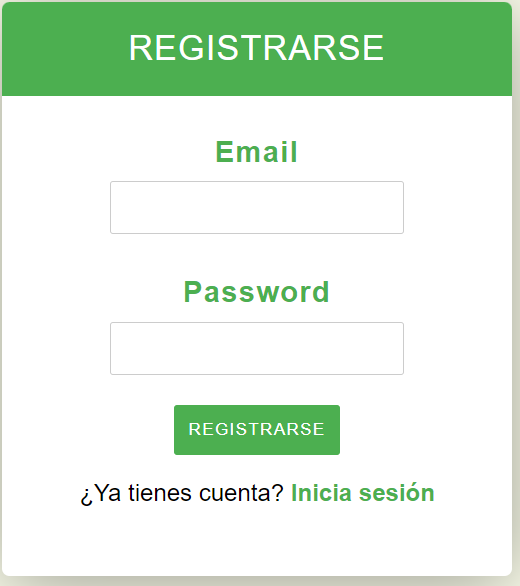
* Inicio de sesión:



El inicio de sesión se realizará con las funciones de autenticación de Firebase. Se introducirán los datos de inicio de sesión (Email y password) y en caso de estar registrados, se iniciará la sesión. En caso contrario, se mostrará un mensaje de error.

En caso de seleccionar “Ingresar con Google”, el usuario utilizará la aplicación web asociada a su cuenta de Google.

* Registro:



El usuario introducirá los campos de registro y si se realiza satisfactoriamente se creará el usuario y se iniciará sesión. En caso contrario, se mostrará un mensaje de error.

En caso de seleccionar “Inicia sesión”, se redirigirá al usuario a la pantalla de inicio se sesión.

## **Diseño de la arquitectura de la aplicación**

### **Tecnologías y herramientas usadas**

**Entornos de desarrollo:**

Los entornos de desarrollo que he utilizado para la realización del proyecto han sido Android Studio para la aplicación móvil y Visual Studio Code para la aplicación Web.

**Lenguajes:**

Los lenguajes que se han utilizado en el proyecto son Java para la aplicación móvil y, para la aplicación Web lenguajes como tal se utiliza Typescript, además de utilizar CSS y HTML, que no llegan a ser considerados lenguajes de programación.

**Frameworks:**

En la aplicación web se ha utilizado el Framework de Angular, que ha ayudado a agilizar el desarrollo de esta aplicación y proyecto.

**Librerías:**

En la aplicación móvil se ha implementado la Material Design para trabajar con algunos componentes y poder mejorar el diseño y funciones de la aplicación.

Otra implementación de la aplicación móvil ha sido SugarORM, para trabajar con un ORM en el almacenamiento local de esta aplicación.

En la aplicación Web se ha trabajado también con la versión de Material para Angular, así como con algún componente de la librería PrimeNG, además de implementar SweetAlert2, que añade modales con sus respectivas funciones.

**Bases de datos:**

El proyecto consta de almacenamiento local y externo en ambas aplicaciones, siendo el almacenamiento externo un almacenamiento de datos compartidos para ambas aplicaciones.

En la aplicación móvil el almacenamiento local se realiza con SQLite con la ayuda de SugarORM.

El almacenamiento externo utiliza Firestore, una base de datos NoSQL de Firebase.

En el caso del almacenamiento local de la aplicación Web, se utilizará LocalStorage, una función propia de Typescript y Javascript que almacena datos en el navegador.

El proyecto consta de dos bases de datos, la local en cada aplicación y la compartida para que el usuario pueda decidir dónde y como almacenar sus datos.

¿No quieres iniciar sesión o no tienes sesión? Puedes almacenar datos localmente.

¿Tienes cuenta y quieres tener los datos en cualquier lado? Puedes almacenar los datos externamente.

**Otras herramientas utilizadas:**

Aunque ya he mencionado en la parte de base de datos que se utiliza Firebase, esta aplicación no se utiliza solo para la base de datos, ya que Firebase ofrece más servicios, como puede ser el servicio de Autenticación, que utilizamos en la aplicación para controlar los usuarios de una manera más cómoda y sencilla, con funciones ya implementadas por Firebase y muchas ventajas implementadas como puede ser decidir la manera de iniciar sesión (Email y contraseña, Google, una red social, móvil…), envíos de correos de verificación y cambio de contraseña, envíos de mensajes móviles de verificación…

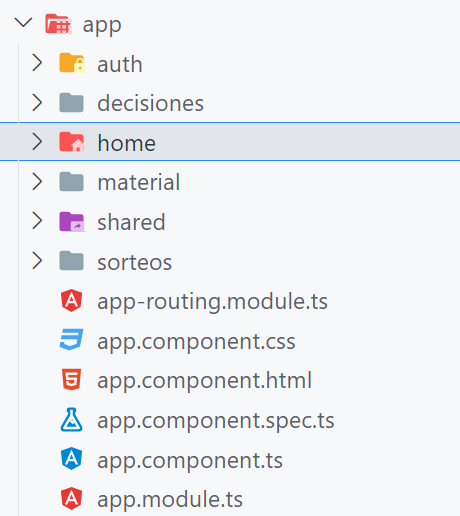
Además, Firebase también implementa en el proyecto “Google Analytics” que nos da estadísticas del uso de la aplicación por parte de los usuarios, indicando por ejemplo la media de usuarios al día en las aplicaciones, la fecha de inicio de sesión de estos, los ingresos (Si los hubiera), etc.

### **Arquitectura de componentes de la aplicación**

La aplicación móvil utiliza una arquitectura de modelo-vista-controlador, disponiendo de los siguientes componentes:

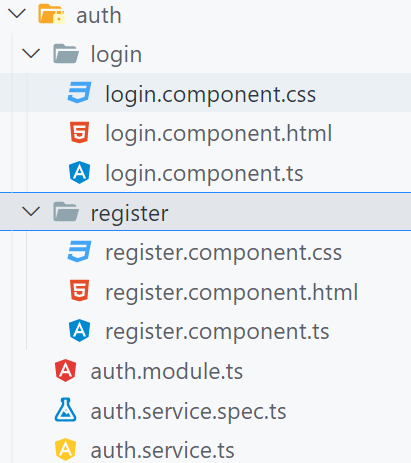
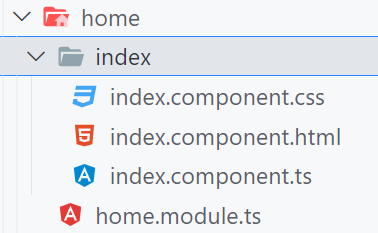
* Clase/Modelo: Son los objetos que forman la aplicación, además que algunos de estos serán las tablas de las bases de datos.
* Vistas/Actividades: Lo que el usuario ve, disponiendo de algo de funcionalidad y llamando en él Layout, que es lo que realmente se ve. En el caso de mi proyecto, las actividades se comunican con los controladores.
* Controladores: Están comunicados con las vistas y los servicios, enviando datos entre estos.
* Servicios: Gestiona datos, creándolos, obteniéndolos y trabajando con ellos. Se comunica con los controladores y los repositorios.
* Repositorios: Se comunican con las bases de datos, ya sea localmente o externamente, realizando los métodos de inserción, obtención, actualización o eliminación. Además, se comunican con los servicios devolviéndoles algunos datos.
* Factory: Convierten datos. Por ejemplo, convierten listas en strings o viceversa, o convierten un objeto en otro.

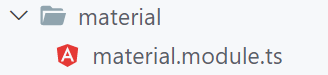
En el caso de la aplicación Web y de un proyecto realizado con Angular, la arquitectura utiliza Módulos, Servicios, Componentes, Rutas e Interfaces.

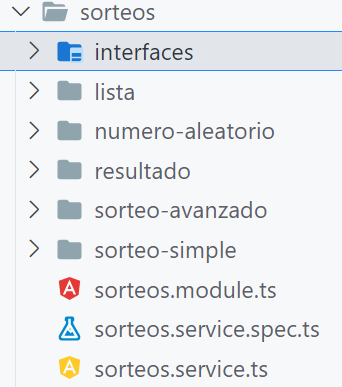
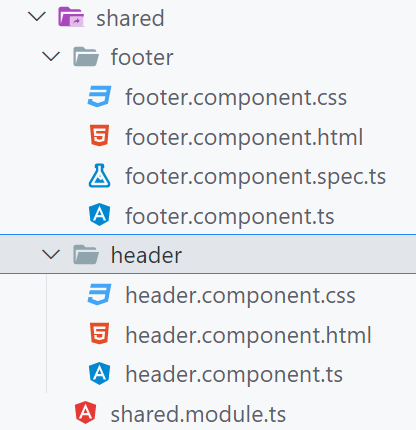
En el caso de mi proyecto, dispongo de la siguiente arquitectura para la aplicación Web: 

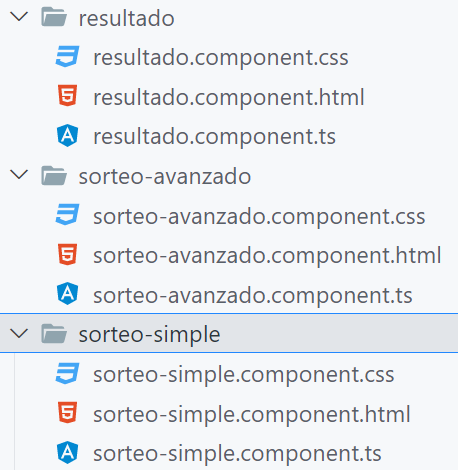
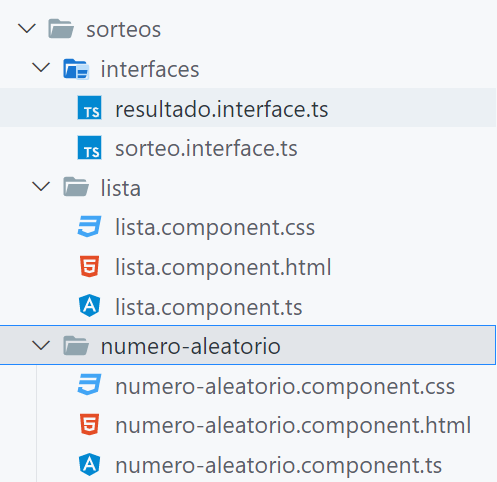
* Módulo principal (En él se importan todos los módulos).
* Módulo “Auth”:
  + Componente “Login”
  + Componente “Registro”
* Módulo “Home”:
  + Componente “Index”
* Módulo “Material” (En él se importan todos los módulos de Angular material que se van a utilizar)
* Módulo “Shared” (Componentes compartidos, que se van a utilizar en varios módulos):
  + Componente “Header”
  + Componente “Footer”
* Módulo “Sorteos”:
  + Componente “SorteoSimple”
  + Componente “SorteoAvanzado”
  + Componente “Listas”
  + Componente “NumeroAleatorio”
  + Componente “Resultado”
  + Interfaces (Objetos)

Por otra parte, cada componente está compuesto por su html, su estilo y su funcionalidad (Typescript)









# **Implementación e implantación del sistema**

## **Implementación**

## **Pruebas**

Para mi proyecto, las pruebas se han ido realizando conforme se iba desarrollando el diseño y la funcionalidad.

Por cada pantalla o página diseñada, se comprobaba que el diseño se veía bien, y, en el caso de la aplicación web, si la página dispone de responsividad, se comprobaba que en todos los tamaños de pantalla se cumplía la función de la responsividad y se veía como se tenía que ver.

Respecto a la funcionalidad, por cada funcionalidad hecha se realizaban las pruebas necesarias.

En mi caso, como la mayoría de las pantallas trabajan con datos introducidos en inputs las pruebas se realizaban introduciendo en estos inputs datos correctos e incorrectos, para comprobar que con los correctos todo funcionaba correctamente y que, con datos incorrectos, los errores también se notificaban correctamente.

Para las funciones que trabajan con bases de datos locales o externas las pruebas se realizaban de la misma manera, trabajando con datos correctos e incorrectos y comprobando que los datos correctos se almacenaban en SQLite, LocalStorage o Firebase.

Las actualizaciones y las eliminaciónes de los datos también se comprobaban en las respectivas bases de datos.

Si alguna prueba no daba el resultado esperado, ya sea satisfactorio o resultado erróneo o esperado, podía cerrarse la aplicación en caso de la aplicación móvil, o saltar errores como era en el caso de la aplicación Web. En estos casos, las pruebas ayudaban a solucionar esos fallos o errores innecesarios, así como sirven las pruebas para solucionar fallos o bugs de las aplicaciones.

# **Documento de cierre**

## **Documento de despliegue y configuración**

* **App móvil:**

Para el despliegue de la aplicación móvil solo se requiere de la instalación de la misma. Podría darse el caso de que la aplicación estuviera publicada en la Play Store, descargarla e instalarla, pero no es el caso.

Así, que, para el caso del proyecto, el despliegue de la aplicación se realiza generando el fichero apk de la misma desde el IDE Android Studio, una vez obtenido el fichero apk deberá ser transferido al dispositivo móvil en el que se desea desplegar la aplicación e instalarse.

Otro caso de despliegue podría ser ejecutar la aplicación desde Android Studio en el dispositivo móvil que se desea instalar, teniendo instalado mediante cable el dispositivo al equipo en el que se desea instalar.

Por último, la aplicación se puede desplegar en emuladores de dispositivos móviles, pero en este caso se desplegaría de la primera manera explicada, que sería descargando la aplicación e instalándola, a no ser que sea el emulador del propio Android Studio y se ejecute la aplicación en él, ya que en ese caso se instalará automáticamente.

* **App Web:**

Para desplegar la aplicación web, en caso de querer desplegarla localmente, deberemos tener instalado en el equipo NodeJs, NPM, Typescript y AngularCLI.

Si todo está instalado correctamente, lo siguiente es añadir al proyecto los módulos, ya que pueden no estar instalados. Para ello, se ejecutará uno de los siguientes comandos (Es el mismo comando, pero resumido, así que hace lo mismo):

npm i

npm install

Una vez instalados los módulos, el proyecto deberá disponer de la carpeta node\_modules. Con estos pasos realizados e instalados, para desplegar la aplicación web se ejecuta el siguiente comando:

ng serve -o

La aplicación web se podrá visualizar en “localhost:4200” (Por defecto, a no ser que se haya cambiado el puerto).

De esta manera, se habrá desplegado la aplicación y podrá visualizarse y utilizarse de manera local.

Si disponemos la aplicación en un equipo nuestro, se podrá, por ejemplo abrir el puerto 4200 para que la aplicación pueda verse fuera de local, además de añadir un dominio a la dirección.

Otra forma de desplegar la aplicación es subirla en un host, como puede ser Netlify, 000WebHost, o como es mi caso, con GitHub Pages.

En mi caso, se puede acceder a la aplicación web mediante el siguiente enlace:

<https://bienzobasjulian.github.io/DecideJbot-WEB-Angular-/#/>

## **Manual de usuario**

## **Resultados obtenidos y conclusiones**

Con el tiempo dedicado al proyecto se ha obtenido una primera versión de DecideJbot funcional.

Esta primera versión realiza la función principal planeada, que es la realización de sorteos y la posibilidad de programar estos y poder tener un registro de estos.

Las conclusiones que he podido sacar respecto al proyecto se basan en la dedicación al mismo, ya que el proyecto no se realiza solo y hay que dedicarle tiempo, y además, poner empeño en este, ya que sin esto el proyecto no llegaría a verse bien ni funcionar bien.

En mi caso, le he puesto bastante empeño y ganas en conseguir esta primera versión funcional, y cómo he comentado en varios puntos, este proyecto surgió como un reto personal y puedo decir, que una parte del reto se ha conseguido, ya que no tengo solo una, si no que tengo dos aplicaciones que cumplen lo que quería, además, he llegado a aprender cosas nuevas relativas a temas de programación, como conocer un framework nuevo que ha sido Angular y he puesto bastante empeño en entenderlo y realizar la aplicación Web con este, quedando satisfecho con la primera versión de las aplicaciones.

Por otra parte, gracias a las prácticas, he podido aprender a trabajar en un proyecto y eso me ha servido para la realización de este.

En resumen, y como conclusión final, considero que el tiempo dedicado al proyecto, ya sea el tiempo de desarrollo y programación como el tiempo de formación e investigación ha sido útil y necesario para la obtención de una primera versión funcional de “DecideJbot”, quedando yo satisfecho con el resultado de estas aplicaciones y además, habiendo adquirido nuevos conocimientos.

## **Seguimiento y desviación respecto a la planificación**

La idea general de “DecideJbot” era una idea más amplia. La aplicación seguía enfocada a la aleatoriedad y el azar, pero la idea principal era implementar una sección más personal, dedicada a decisiones de mi interés, y que a otros usuarios también les podrían servir esa sección, ya que la sección de “Decisiones” iba enfocada a decisiones respectivas al ocio, como podían ser Series, Películas, Videojuegos, Música…

Este apartado, habría servido para tomar decisiones en esos temas específicos, y tener un registro de las decisiones tomadas en estos y la actividad realizada en estos temas.

Este apartado habría utilizado APIs relacionadas a esos temas, para trabajar con datos ya creados y no tener que registrar todos esos datos manualmente, ya que son extensos.

Debido a desvíos en la planificación y al tiempo de entrega, el apartado de decisiones y algunas otras funciones planeadas quedan previstas como futuras tareas, para futuras versiones de la aplicación.

# **Bibliografía**

# **Anexos**