Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Grado en Ingeniería Informática

Gamificación de visitas urbanas con Beacons

Gamification of urban visits with Beacons

Rafael Cala González

La Laguna, 14 de julio de 2023

D. **Jezabel Miriam Molina Gil**, con N.I.F. 78.507.682-B profesor Ayudante Doctor, adscrita al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutora.

C E R T I F I C A (N)

Que la presente memoria titulada:

“Gamificación de visitas urbanas con Beacons”

ha sido realizada bajo su dirección por D. **Rafael Cala González**,

con N.I.F. 45.689.185-Q.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 14 de julio de 2023

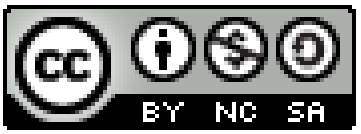
Agradecimientos

A mi tutora de proyecto, Jezabel Miriam Molina Gil, por brindarme la idea de este trabajo y proporcionar todos los recursos y la información necesaria para el proyecto.

A mi familia, pareja y amigos, por ser el barco que me ha permitido navegar con un rumbo fijo en esta larga travesía.

Y a papá. Porque aunque ya no estés, estás.
En el sol que me abraza,
en la brisa que me acaricia,
en el silencio que me escucha
y en mí.

Licencia



© Esta obra está bajo una licencia de Creative Commons Reconocimiento-NoComercial-CompartirIgual 4.0 Internacional.

Resumen

El turismo se ha convertido en uno de los sectores más importantes a la hora de generar riquezas dentro de un país o una ciudad, produciendo un impacto económico en un amplio rango de ámbitos, tales como la gastronomía, museos, monumentos, lugares de ocio, el transporte...

Introducir mecánicas típicas de juego a este sector a través de la gamificación puede resultar interesante para promover las visitas a ciertos lugares de interés cultural, aportando además un mayor dinamismo a la experiencia.

El objetivo de este proyecto es el desarrollo de una aplicación consistente en un juego cuya competitividad viene dada por visitar lugares de interés cultural, de manera que a modo de interfaz principal se dispone de un mapa donde se situarán marcadores asociados a estos sitios. En cada una de esas posiciones, se encontrarán localizadas balizas electrónicas denominadas Beacons a la espera de que el usuario se acerque para lanzar en la aplicación una pregunta relacionada con el entorno en el que está.

Esta aplicación está disponible para ser usada en dispositivos móviles Android, con una interfaz sencilla y minimalista, tratando de evocar la sensación de exploración que se busca conseguir en el juego e incorporando un sistema de puntuación con el que se obtiene la gamificación, dependiente de los lugares que se visitan.

Para llevar a cabo este trabajo se han utilizado como tecnologías principales: React, React Native, Firebase y dispositivos Beacons.

Palabras clave: Turismo, gamificación, mapa, React, React Native, Firebase, Beacons.

Abstract

Tourism has become one of the most important sectors when it comes to generating wealth within a country or a city, generating an economic impact in a wide range of areas, such as gastronomy, museums, monuments, places of leisure, transport...

Introducing typical game mechanics to this sector through gamification can be interesting to promote visits to certain places of cultural interest, also providing a greater dynamism to the experience.

The objective of this project is the development of an application consisting of a game whose competitiveness is given by visiting places of cultural interest, so that as the main interface there is a map where markers associated with these sites will be placed. In each of these positions, electronic beacons called Beacon will be located waiting for the user to approach to launch in the application a question related to the environment in which it is.

This application is available for use on Android mobile devices, with a simple and minimalist interface, trying to evoke the feeling of exploration that is sought in the game and incorporating a scoring system with which the gamification is obtained, dependent on the places visited.

To carry out this work, it had to be used as main technologies: React, React Native, Firebase and Beacons devices.

Keywords: Tourism, gamification, map, React, React Native, Firebase, Beacons.

Índice general

Capítulo 1 Introducción..... 1
1.1 Motivación.....1
1.2 Objetivos.....1
1.3 Estructura de la memoria.....2
Capítulo 2 Estado del arte..... 3
2.1 Estado del arte.....4
2.2 Aplicaciones existentes..... 4
2.3 Conclusiones sobre las aplicaciones existentes.....4
Capítulo 3 Preliminares..... 4
3.1 Glosario..... 5
3.2 Funcionamiento del juego.....6
Capítulo 4 Tecnologías utilizadas.....7
4.1 Base de datos.....7
4.2 Frontend de la aplicación web de administración.....7
4.3 Frontend para dispositivos móviles o tabletas.....7
4.4 Backend.....7
4.5 Otros.....7
Capítulo 5 Backend..... 9
5.1 Base de Datos..... 9
5.2 Roles del proyecto..... 11
5.3 Autenticación de usuarios.....11
5.4 Reglas de la Base de Datos..... 12
5.5 Alojamiento de imágenes..... 12
Capítulo 6 Frontend..... 13
6.1 Diseño.....13
6.2 Aplicación móvil..... 13
6.2.1 Algoritmo de detección de Beacons.....13
6.2.2 Módulos relevantes..... 13
6.2.3 Pantallas..... 14
6.3 Aplicación web.....22
6.3.1 Aspectos a tener en cuenta..... 22
6.3.2 Pantalla de inicio de sesión..... 22

6.3.3 Funcionalidad general de la interfaz principal..... 23
6.3.4 Pantallas de la interfaz principal con funcionalidad general..... 31
6.3.5 Pantallas de la interfaz principal con funcionalidad específica... 33
Capítulo 7 Presupuesto..... 43
Capítulo 8 Conclusiones y líneas futuras..... 44
Capítulo 9 Summary and Conclusions..... 45
Capítulo 10 Anexos..... 48

Índice de figuras

Figura 6.1: Splash Screen 13
Figura 6.2: Diapositivas de la pantalla ‘Onboarding’ 14
Figura 6.3: Inicio de sesión 15
Figura 6.4: Crear cuenta (paso 1) 15
Figura 6.5: ‘Crear cuenta’ (paso 2) 16
Figura 6.6: Mapa de juego 16
Figura 6.7: Icono posición del usuario 17
Figura 6.8: Puntuaciones en el mapa 17
Figura 6.9: Aviso de bluetooth desactivado 17
Figura 6.10: Diálogo de bluetooth desactivado 17
Figura 6.11: Aviso de notificación encontrada 17
Figura 6.12: Pregunta de la notificación emergente 18
Figura 6.13: Posibles estados del Beamarker 18
Figura 6.14: Menú de pistas de notificaciones 19
Figura 6.15: Menú de navegación inferior 19
Figura 6.16: Modos del BeaCard 19
Figura 6.17: Estadísticas de la pantalla ‘Cuenta’ 20
Figura 6.18: Listado de la Beapedia 20
Figura 6.20: Interfaz de inicio de sesión en la aplicación web 22
Figura 6.21: Componente Header 23
Figura 6.22: Menú lateral 23
Figura 6.23: Ejemplo de ModuleView 23
Figura 6.24: Ejemplo de LoadingOverlay activo 24
Figura 6.25: CustomTable de la vista ‘Eventos’ 24
Figura 6.26: CustomTable de la vista ‘Eventos’ haciendo uso del filtro 24
Figura 6.27: Diálogo de confirmación de eliminación de elemento 25
Figura 6.28: GenericDetailView de la colección Beacons 25
Figura 6.29: Input de tipo ‘Booleano’ 26
Figura 6.30: Input de tipo ‘Texto’ 26
Figura 6.31: Input de tipo ‘Área de texto’ 26
Figura 6.32: Input de tipo ‘Numérico’ 26
Figura 6.33: Input de tipo ‘Fecha’ 26

Figura 6.34: Asignación por búsqueda en el input de tipo ‘Coordenadas’ 27
Figura 6.35: Asignación manual en el input de tipo ‘Coordenadas’ 27
Figura 6.36: Resultado de asignación en el input de tipo ‘Coordenadas’ 27
Figura 6.38: Input de tipo ‘URL de imagen’ 28
Figura 6.39: Input de tipo ‘Cálculo’ 28
Figura 6.40: Input de tipo ‘Objeto’ 28
Figura 6.41: Mensajes de error en las validaciones de Lugares 29
Figura 6.42: Ejemplos de notificaciones 29
Figura 6.43: Pantalla Niveles de usuario 32
Figura 6.44: Nivel de usuario 33
Figura 6.46: Apartado de selección de colores en el nivel de usuario 33
Figura 6.47: Tooltip de ayuda en el apartado de selección de colores 34
Figura 6.48: Apartado de puntuaciones del nivel 34
Figura 6.49: Incumplimiento de las reglas de niveles de usuario 35
Figura 6.50: Nivel por añadir 35
Figura 6.51: Nivel por eliminar 36
Figura 6.52: Notificación de superación de la puntuación máxima 36
Figura 6.53: Filtros de las notificaciones 37
Figura 6.54: Detalle del usuario 38
Figura 6.55: Diálogo de asignación de notificaciones 39
Figura 6.56: Pantalla Almacenamiento 40
Figura 6.57: Diálogo de modificación de imagen 40
Figura 6.58: Diálogo de nueva imagen 41

Índice de tablas

Tabla 7.1: Presupuesto del proyecto 41

Capítulo 1 Introducción

1.1 Motivación

A día de hoy, el turismo se ha tornado una de las actividades económicas y culturales de las que más se puede nutrir un país o una ciudad. Así mismo, el turismo se puede clasificar en numerosos tipos, siendo el cultural y el de aventura los que ocupan mayor importancia en el proyecto que se pretende realizar.

Por otro lado, la competitividad es una característica natural en el ser humano que aporta una serie de ventajas en nosotros, como permitirnos progresar, despertar nuestra creatividad o mejorar nuestra eficiencia, entre otras muchas cosas. Es posible combinar las dos ideas previamente expuestas por medio del concepto denominado gamificación, que busca trasladar la mecánica de los juegos al ámbito educativo-profesional. Con estas premisas puestas sobre la mesa, resulta evidente que disponer de una aplicación donde se compite por visitar lugares de interés cultural, supone una dinamización de la experiencia turística.

Es así como nace este Proyecto de Trabajo de Fin de Grado. Se situarán balizas electrónicas llamadas Beacons en las zonas a visitar, de modo que estarán pendientes que el usuario entre dentro de su alcance, en cuyo caso se dará por visitada la zona en el lugar. Este proceso entra dentro del funcionamiento del juego, por lo que se verá con mayor profundidad más adelante.

1.2 Objetivos

Tal y como se ha comentado anteriormente, el objetivo principal es desarrollar un proyecto que fomente las visitas urbanas por medio de la gamificación. El resultado es una **aplicación móvil** que cuenta como interfaz principal con un mapa, donde se sitúan marcadores referentes a una serie de dispositivos denominados Beacons en puntos diferentes del mapa previamente nombrado.

Se ha tenido en cuenta el contexto del proyecto, incluyendo la época del año y el tipo de usuarios que utilizarán la aplicación. Por este motivo, se ha desarrollado una **aplicación web de administración** que permite la configuración de preguntas, notificaciones y otros aspectos relacionados con los dispositivos Beacons. Además, se ha habilitado la parametrización de textos descriptivos, la asignación de posiciones en el mapa para cada beacon y la gestión de puntos ganados por los usuarios, entre otros datos relevantes dentro de este trabajo.

Para lograr esto, se ha integrado una **base de datos** que permite el manejo, almacenamiento y alojamiento de toda esta información de forma eficiente y segura.

Asimismo, se ha considerado fundamental contar con un **sistema de registro de usuarios**, que permite a cada usuario identificarse dentro de la aplicación y tener su propia puntuación personalizada en función de los lugares visitados.

1.2.1 Aplicación móvil

En la aplicación móvil se da la funcionalidad principal del proyecto y está destinada a los usuarios. Dispone de una interfaz inicial, donde se puede iniciar sesión si se tiene una cuenta creada, o crearla en caso contrario. Una vez se tenga una sesión iniciada se cargará la interfaz principal: el mapa donde están situados los marcadores. La aplicación también provee al usuario de una pantalla desde la que podrá ver información sobre los lugares a visitar y otra pantalla en la que podrá consultar toda la información de su perfil.

1.2.2 Aplicación web

La aplicación web proporciona a los administradores del proyecto una herramienta con la que poder realizar operaciones de lectura, creación, borrado y modificación de los datos con los que se cuenta. De este modo, resulta mucho más sencilla y segura la gestión de los mismos, ya que se incluyen los campos a rellenar con sus validaciones correspondientes. Sumado a ello, el realizar un diseño pensado específicamente para cada colección del proyecto resulta en un mayor dinamismo y una mejor experiencia respecto a modificar la información directamente desde la base de datos.

1.3 Estructura de la memoria

La estructura de la memoria se organiza de la siguiente manera:

En el **Capítulo 1 Introducción** se introduce el proyecto, definiendo las motivaciones y objetivos que lo impulsan. Se ofrece una breve explicación sobre el propósito de las aplicaciones desarrolladas.

El **Capítulo 2 Estado del Arte** se dedica al estado del arte, donde se realiza un estudio de la situación actual de los conceptos que dan significado al proyecto. Se lleva a cabo un análisis y se extraen conclusiones sobre aplicaciones similares.

El **Capítulo 3 Preliminares**, se compone de un glosario de términos utilizados en la redacción y una explicación del funcionamiento de la aplicación móvil.

En el **Capítulo 4 Tecnologías utilizadas** se presenta un listado de las herramientas y tecnologías que se han utilizado para el desarrollo del proyecto y se describe detalladamente cada una de ellas y su relevancia en el contexto del proyecto.

El **Capítulo 5 Backend** se centra en aspectos de la arquitectura de la aplicación y en la definición de los elementos que constituyen el aplicativo Backend del proyecto, como son el registro de usuarios, la base de datos y el alojamiento de imágenes.

El **Capítulo 6 Frontend** aborda el diseño y toda la funcionalidad de la aplicación móvil y web. Para ello, se describen los elementos que constituyen todas las interfaces implementadas y los procesos que se llevan a cabo en ellas.

El **Capítulo 7 Presupuesto** presenta una tabla con el desglose de tareas realizadas en el desarrollo completo del trabajo, aportando un coste estimado del mismo en base a un precio/hora estimado.

En el **Capítulo 8 Conclusiones y líneas futuras**, se realiza una discusión y se extraen conclusiones. Se analizan y discuten los resultados obtenidos, comparándolos con los objetivos planteados en el Capítulo 1. Se presentan las conclusiones finales sobre el éxito

del proyecto y posibles mejoras futuras a incorporar en el proyecto.

In **Chapter 9, Summary and conclusions**, a discussion is held and conclusions are drawn. The results obtained are analyzed and discussed, comparing them with the objectives set out in Chapter 1. The final conclusions about the success of the project and possible future improvements to be incorporated into the project are presented.

Finalmente, el **Capítulo 10**, titulado Anexos, incluye cualquier material adicional relevante para el proyecto, como porciones de código, documentos técnicos y capturas entre otros.

Capítulo 2 Estado del arte

2.1 Estado del arte

El uso de Beacons en la actualidad se encuentra en un constante y prometedor crecimiento, con un despliegue cada vez mayor de estos dispositivos y múltiples casos de uso como pueden ser el rastreo de dispositivos u objetos, pedidos remotos o el movimiento en interiores.

Muchas empresas e industrias están tratando de aprovechar esta tecnología para, no sólo incrementar su cuota de mercado, sino también para enriquecer la experiencia del consumo de sus clientes.[1]

En el sector que ocupa a este proyecto, el turismo, estas balizas electrónicas están suponiendo una revolución en cuanto a la comunicación con las personas, ofreciendo una visualización de información del entorno más interactiva y dinámica, teniendo en cuenta que un inmenso porcentaje de la población porta consigo un dispositivo móvil.

2.2 Aplicaciones existentes

Como se ha dicho previamente, existen muchas aplicaciones que involucran el turismo y el uso de Beacons, pero dentro de un nivel local. Es decir, su finalidad está destinada a ciertas infraestructuras como museos, exposiciones u otros eventos con temática similar, lo que le resta escalabilidad.

Existen multitud de proyectos con objetivos similares a este, como Pokemon Go[2] o Touristfy[3]. Ambos utilizan mecánicas de gamificación e incorporan un mapa en sus respectivas dinámicas, pero se diferencian en su enfoque y tecnología.

Pokemon Go es un juego móvil de realidad aumentada donde los jugadores, llamados “entrenadores”, tratan de capturar criaturas virtuales llamadas “Pokémon” que aparecen en ubicaciones del mundo real dentro del mapa. Como se puede apreciar, aunque implementa un mapa como interfaz principal y naturalmente tiene todo tipo de técnicas de gamificación, no tiene un fin cultural ni turístico si bien fomenta la exploración.

Por otro lado, Touristfy, una app que permite gamificar itinerarios o recorridos turísticos orientados a un turismo familiar, tiene un objetivo similar al de este proyecto, incorporando incluso mecánicas de gamificación, pero su enfoque no es urbano. Además, la dinámica de juego respecto al proyecto .

Se destaca en adición que ambos se basan en la geolocalización, lo que puede resultar en una menor precisión, con las limitaciones que conlleva.

2.3 Conclusiones sobre las aplicaciones existentes

Se concluye que existen aplicaciones que ofrecen experiencias parecidas a las de este proyecto, dada la similitud en algunas de las características comentadas. Sin embargo, no se hallan aplicaciones de relevancia que hagan uso de dispositivos Beacon, lo cual es un

factor diferencial, ya que ello implica que, a pesar de las limitaciones físicas que presentan estas balizas electrónicas, proporcionan una mayor precisión en la localización de los lugares y en consecuencia un uso más eficaz en interiores.

Esta aplicación se centra en el turismo urbano, donde los lugares de interés cultural son la prioridad. Además, la naturaleza del proyecto permite que se pueda implementar en ciudades de todo el mundo, sin estar limitado a eventos o infraestructuras específicas. Por ende, el enfoque inicial del proyecto está dirigido a un público más interesado en lo cultural, aunque alicientes como la escalabilidad, el uso de la gamificación y el diseño aportan a la aplicación el potencial para abarcar un público mayor en el futuro.

Capítulo 3 Preliminares

3.1 Glosario

A lo largo de la memoria del proyecto, se hará alusión a numerosos términos que, ya sea por tratarse de anglicismos o tecnicismos, podrían ser ajenos al conocimiento del lector, por lo que merecen una explicación individual más detallada sobre los mismos, que se recogerá en este glosario.

Frontend: Es la parte de una aplicación que comprende todos los elementos que permiten navegar dentro de la misma y con los que interactúa el usuario. [4]

Backend: Es la parte de una aplicación donde se maneja toda la lógica y el conjunto de acciones que hacen que esta funcione, como por ejemplo la comunicación con el servidor. [5]

Local storage: Es el espacio de almacenamiento interno a nivel de aplicación en el cual típicamente se guardan datos como configuraciones internas, preferencias del usuario, tokens de autenticación, datos temporales o cualquier otra información relevante para el funcionamiento de la aplicación.

React Hooks: Los hooks son una nueva característica introducida en la versión 16.8 de React que permite utilizar el estado y otras características de React sin escribir una clase. [6]

Tablero Kanban: Un tablero de kanban es una herramienta ágil de gestión de proyectos diseñada para ayudar a visualizar el trabajo, limitar el trabajo en curso y maximizar la eficiencia o el flujo.[7]

Material Design: Conjunto de directrices, componentes y herramientas sobre buenas prácticas para el diseño de interfaces de usuario. [8]

Firebase: Es un Backend-as-a-Service (Baas) construido por Google que proporciona a los desarrolladores una variedad de herramientas y servicios que les ayudan en el desarrollo de proyectos informáticos. [9]

Beacon: Dispositivo inalámbrico BLE (Bluetooth Low Energy) que funciona a modo de baliza electrónica, es decir, transmite repetidamente una señal constante que puede ser detectable por otros dispositivos. [10]

Beamarker: Nombre que se le ha designado a los marcadores dentro del mapa del juego del proyecto.

Input: Es la palabra designada para englobar un conjunto de campos de texto utilizados para almacenar la información de un solo dato. Estos inputs tienen una lógica adecuada a cada tipo de dato que se puede manejar en el proyecto.

NoSQL: También conocidas como No relacionales, son las bases de datos que no emplean el modelo de datos relacional que utilizan bases de datos como Oracle, MySQL,

PostgreSQL, haciendo uso de esquemas flexibles y fáciles de desarrollar. [11]

3.2 Funcionamiento del juego

La mecánica que da el funcionamiento principal del trabajo se da en la aplicación móvil. La interfaz donde transcurre el juego muestra un mapa sobre el que se sitúan una serie de marcadores denominados *Beamarkers* y la posición actual del usuario, que se actualizará conforme se vaya moviendo en la vida real.

Estos marcadores designan una ubicación a visitar en el juego y al ser pulsados sobre ellos, despliegan algo de información sobre dicho lugar, como el nombre que lo caracteriza, la distancia a la que está el usuario del lugar, así como también información sobre las notificaciones asociadas al lugar, como el nivel de dificultad y el evento que las representa o una pista relacionada con la localización física del beacon.

Cuando el usuario se encuentra lo suficientemente cerca de uno de los marcadores como para recibir la señal de uno de sus beacons, en la app emergerá una notificación que mostrará una pregunta sobre el entorno y una serie de respuestas, siendo solo una de ella la correcta. Dependiendo de si se acierta o no y del nivel de dificultad que tenga la notificación en el lugar, el usuario recibirá un número de puntos.

No obstante, si el usuario se aleja de nuevo de la posición del beacon sin haber respondido en la notificación, ésta volverá a ocultarse.

La cantidad de puntos que tenga el usuario servirá para clasificarlo en un nivel de jugador dentro de la propia aplicación. Estos pueden ser: bronce, plata, oro y diamante. Asimismo, el componente perteneciente a la puntuación se renderiza con unos colores acordes al nivel actual.

Cuando la pregunta es respondida, el usuario registra la notificación de dicho lugar junto a este último, si no hubiera sido visitado con anterioridad. El usuario podrá consultar información sobre cada uno de los lugares registrados en la aplicación y las notificaciones, distinguiéndose las visitadas de las que no ha visitado aún.

Capítulo 4 Tecnologías utilizadas

Para exponer las tecnologías que se han utilizado a lo largo del proyecto, es conveniente hacer una distinción a modo de apartados, los cuales son:

- Base de datos
- Frontend de la aplicación web de administración
- Frontend para dispositivos móviles o tabletas
- Backend
- Otros

4.1 Base de datos

Cloud Firestore: Es un servicio que provee Firebase cuya funcionalidad es servir como almacenamiento de datos en la nube. Se trata de una base de datos NoSQL. [12]

4.2 Frontend de la aplicación web de administración

React: Librería de JavaScript que permite la construcción de interfaces de usuario. [13]

Material-UI: Paquete de componentes que siguen los principios de Material Design.

4.3 Frontend para dispositivos móviles o tabletas

React Native: Librería que, al igual que React, permite la construcción de componentes o interfaces de usuario, con la diferencia de que en este caso, se ejecuta sobre plataformas móviles nativas. [14]

React Native Paper: Colección de componentes personalizables para React Native que sigue las directrices de Material Design.

4.4 Backend

Firebase Authentication: Herramienta de Firebase que ofrece servicios de Backend para la autenticación de usuarios en el proyecto.[15]

Firebase Cloud Storage: Servicio de Firebase consistente en el almacenamiento de imágenes, audio, video y otros tipos de contenido generado por el usuario. [16]

4.5 Otros

GitHub: Plataforma con la que se ha llevado a cabo el control de versiones del proyecto en la nube. [A1] [A2]

Notion: Software que se ha utilizado para la gestión del proyecto, haciendo uso de tableros Kanban que han permitido llevar un flujo organizado de trabajo.

Visual Studio: Es el entorno de desarrollo integrado que se ha utilizado para el desarrollo

del código perteneciente a las aplicaciones web y móvil.

Android Studio: Es el entorno de desarrollo integrado que se ha utilizado para ejecutar y compilar la aplicación móvil.

npm: Gestor de paquetes de JavaScript que se ha usado para agregar y administrar eficazmente los módulos usados por el proyecto.

Capítulo 5 Backend

5.1 Base de Datos

Se ha optado por utilizar Cloud Firestore como la herramienta que provee la Base de Datos. Esencialmente, Cloud Firestore usa colecciones, y cada una de ellas alberga documentos que tienen una serie de campos representativos.

Las colecciones que recoge el proyecto junto a los campos de sus documentos son las siguientes.

Usuarios (users): Almacena todos los campos representativos del perfil de cada usuario de la aplicación.

- **Email:** Correo electrónico del usuario.
- **Name:** Nombre del usuario.
- **Apellido:** Apellido del usuario.
- **Username:** Nombre identificativo para el usuario.
- **Rol:** Rol del usuario, que puede ser “player” o “admin”.
- **Puntuación:** Número de puntos que tiene el usuario.
- **Notificaciones visitadas:** Lista de ids de los documentos de la colección Notificaciones que ha visitado el usuario.
- **Lugares visitados:** Lista de ids de los documentos de la colección Lugares que ha visitado el usuario.
- **Aciertos:** Lista de ids de los documentos de la colección Notificaciones que ha acertado el usuario al visitarla.

Usernames (usernames): Los documentos de esta colección no tienen campos. Sus UIDs contienen el nombre de usuario recogido en la colección “Usuarios”. Cuando se crea o se modifica un nombre de usuario, se comprueba si dicha UID ya existe en esta colección. Dicha redundancia evita realizar consultas en las que se compruebe usuario por usuario si el nombre de usuario ya se encuentra en uso, lo cual resulta más ineficiente.

Beacons (beacons): Almacena la información necesaria de cada Beacon del proyecto.

- **Identificador**: Nombre que se le da al beacon a fin de reconocerlo con mayor facilidad.
- **UUID**: Cadena de 32 caracteres.
- **Major**: Valor entero entre 1 y 65535.
- **Minor**: Valor entero entre 1 y 65535.
- **Notificación asociada**: Referencia de un documento de la colección Notificaciones que designa la notificación que está vinculada al beacon.

Lugares (locations): Recoge toda la información de los lugares registrados en el proyecto para su posterior visualización en el juego.

- **Nombre del luga**r: Nombre que identifica al lugar.

- **Imagen**: Imagen identificativa del lugar.
- **Localización**: Coordenadas geográficas del lugar.
- **Notificaciones**: Lista de notificaciones asociadas.
- **Descripción**: Breve descripción histórica y contextual del lugar.

Eventos (events): Contiene la información sobre las campañas que se pueden dar en el juego, teniendo definidos un momento de inicio y opcionalmente un momento de finalización. Solo estarán disponibles en la aplicación las notificaciones cuyos eventos estén en curso.

- **Título**: Nombre del evento.
- **Descripción** (*opcional)*: Texto descriptivo de la campaña o evento.
- **Fecha inicio**: Fecha en la que da comienzo la campaña.
- **Fecha fin** (*opcional)*: Fecha en la que finaliza la campaña.

Notificaciones (notifications): Contiene la información de las notificaciones que pueden ser encontradas por el usuarios en los lugares de la aplicación. Una notificación se debe asociar, como se adelantaba, con un lugar, con un evento, con un nivel de dificultad y opcionalmente con un beacon.

- **Hint** (*opcional):* Pista sobre cómo encontrar la ubicación física de la notificación.
- **Notificación:**
    - **Pregunta:** Pregunta a responder de la notificación.
    - **Respuestas:** Respuestas posibles, a su vez tiene 2 campos.
        - **Respuesta**
        - **Es correcta**
- **Evento**: Referencia a un documento de la colección de Eventos.
- **Nivel:** Referencia a un documento de la colección de Niveles de dificultad.
- **Location:** Referencia a un documento de la colección Lugares.
- **Beacon:** Referencia a un documento de la colección Beacons.

Niveles de dificultad (difficultLevels): Presenta documentos con los diferentes niveles de dificultad que tienen las preguntas de Notificaciones, de modo que se indican también los puntos que se ganan tanto al acertar como al fallar la pregunta.

- **Nivel:** Nombre del nivel de dificultad.
- **Correcta:** Puntos que obtiene el usuario si acierta la pregunta de la notificación.
- **Equivocada:** Puntos que obtiene el usuario si falla la pregunta de la notificación

Niveles de usuario (userLevels): Presenta documentos con los diferentes niveles que puede adquirir el usuario conforme va ganando puntos respondiendo las preguntas. Cada documento tiene además parámetros de configuración para su presentación en la aplicación.

- **Nivel:** Nombre del nivel de usuario
- **Máximo:** Valor máximo del nivel
- **Mínimo:** Valor mínimo del nivel
- **Es nivel máximo:** Indica si el nivel es máximo que puede alcanzar el usuario. Cuando está activo, no se tiene en cuenta el Máximo
- **Color:** Color que caracteriza el nivel

- **Color de texto:** Color que caracteriza el texto del nivel

Configuración (configuration): Las configuraciones contienen documentos con parámetros generales para el funcionamiento de la aplicación. Se permite tener varios documentos mientras sólo haya un documento activo. Esto se debe puede resultar conveniente tener configuraciones para ciertos momentos o festividades del año.

- **Lugar por defecto:** URL de imagen que se muestra por defecto en el caso de que la imagen de un lugar ya no exista, no esté disponible o definida.
- **Usuario por defecto:** URL de imagen que se muestra por defecto en el caso de que la imagen de un usuario ya no exista, no esté disponible o definida.
- **Máxima puntuación:** Valor que representa la puntuación máxima alcanzable por los usuarios.
- **Activo:** Indica si la configuración está activa.
- **Última modificación:** Indica la fecha de la última vez que se modificó el documento.

5.2 Roles del proyecto

Se distinguen dos tipos de usuario en el proyecto:

Administradores: Encargados de añadir, modificar o eliminar documentos de las diferentes colecciones del proyecto. Solo este tipo de usuario tiene acceso a la página web de administración del proyecto.

Usuarios: Son aquellos que solo tienen acceso a la aplicación móvil, disponiendo de todas las funcionalidades que ofrece la misma. Como se verá más adelante, estos solo tienen permiso de modificación para su propio usuario creado, y solo lectura para las otras colecciones de la Base de Datos.

5.3 Autenticación de usuarios

El registro de usuarios se ha logrado haciendo uso de la funcionalidad de autenticación con la que cuenta Firebase, liberando al proyecto de las tareas correspondientes en el Backend. Actualmente, solo se puede llevar a cabo dicho proceso mediante correo electrónico y contraseña.

Se utilizan tres funciones en el proyecto, en cada cual ocurre lo siguiente:

Registro

Se crea una cuenta de usuario, junto con el documento correspondiente al perfil de usuario asociado a la misma en la colección *Usuarios.* Al mismo tiempo, se comprueba si el nombre de usuario introducido es válido y no es repetido, en cuyo caso se escribe en un documento creado en la colección *Usernames*. La UID de dicho perfil será la misma que la UID del usuario que ha creado la cuenta.

Iniciar sesión

Cuando se inicia sesión, se obtienen los datos de la cuenta de usuario, donde uno de

esos campos es la UID. De esa UID se obtiene la información del perfil, mostrándose en la aplicación.

Cerrar sesión

Tras el cierre de sesión, se sale de la interfaz de la aplicación y se carga la pantalla de inicio de sesión, ocultando la mayoría de las operaciones posibles en la Base de Datos.

5.4 Reglas de la Base de Datos

Las reglas definidas en la Base de Datos resultan sencillas pero lo suficientemente robustas como para lograr una correcta seguridad en el proyecto [A3]:

- Todas las colecciones permiten la lectura siempre y cuando **se esté autenticado**.
- Los **usuarios** tienen permitida la lectura de otros usuarios, pero solo pueden realizar **modificaciones sobre sí mismos**.
- Aparte del punto anterior, solo están permitidas las modificaciones en el resto de colecciones siempre y cuando el usuario que vaya a realizar la operación sea "**ADMIN**".

5.5 Alojamiento de imágenes

Para alojar las imágenes de la aplicación se hace uso de Cloud Storage. Se cuenta con los siguientes directorios:

Usuarios (users/)

Se almacenan las fotos de perfil actuales de cada usuario que se hayan asignado una.

Defecto (default/)

Directorio usado para las imágenes por defecto de las configuraciones generales.

Lugares (places/)

En este directorio se alojan las imágenes de los lugares registrados en la aplicación.

Capítulo 6 Frontend

6.1 Diseño

El diseño de este Trabajo de Fin de Grado sigue una estética sencilla y minimalista, tratando de evocar en el usuario el interés por la exploración que el proyecto pretende alcanzar, pero sin llegar a perder la inclusión de información de carácter cultural relevante que le pueda resultar interesante.

Si bien el diseño se sustenta sobre las pautas y principios de Material Design, se ha optado por una mayor personalización en las interfaces y elementos para darles una identidad propia que les aporte personalidad.

Para el caso de la aplicación web, se ha usado el framework de componentes de React llamado Material UI, mientras que en la aplicación móvil se ha hecho uso de Paper. Sin embargo, la personalización desde este lado es mayor, dando lugar a componentes que distan bastante de lo que suele ofrecer el estilo Material.

Cabe destacar que la aplicación móvil también ha hecho uso de otros módulos que merecen especial atención dada la importancia que tienen para el correcto funcionamiento y se detallarán más adelante.

6.2 Aplicación móvil

6.2.1 Algoritmo de detección de Beacons

Se tiene un evento que está permanentemente a la escucha de Beacons. Cuando se encuentra uno, se procesa la información que almacena. Si se trata de un beacon del proyecto, se pasa a comprobar que este no haya sido visitado anteriormente por el usuario y que su notificación esté en curso. Esto último se puede saber a través del evento al que está asociada la propia notificación, comprobando que el momento actual esté situado entre la fecha de inicio y la fecha de finalización indicado en el evento. Si se dan las condiciones, se hace visible la notificación con la pregunta que debe ser respondida. En caso contrario, se ignora el beacon.

Se cuenta también con un evento a la escucha de los Beacons que dejan de encontrarse en el rango detectable por el dispositivo móvil. De este modo, si el usuario que se había acercado lo suficiente a un beacon vinculado a un lugar no visitado del proyecto y había recibido la notificación se vuelve a alejar, este evento volverá a hacer invisible dicha notificación, por lo que tendría que volver a acercarse al beacon para tener acceso a ella. [A4]

6.2.2 Módulos relevantes

react-native-kontaktio: Permite la configuración y comunicación con los Beacon en dispositivos Android y iOS.

react-native-maps: Módulo que permite el uso de numerosas funcionalidades de Google Maps en React Native. Se ha usado para mostrar el mapa del juego y para disponer en él

los distintos Beamarkers registrados en la aplicación.

react-navigation: Módulo que facilita en gran medida el enrutamiento y la navegación por las diferentes interfaces que componen la aplicación.

react-native-image-crop-picker: Módulo que permite la lectura de imágenes del dispositivo móvil y que se usa para asignar una foto de perfil al usuario.

@react-native-async-storage/async-storage: Módulo que proporciona una lógica sencilla y asíncrona para almacenar datos persistentes a nivel de dispositivo. Se usa para determinar si es o no la primera vez que se carga la aplicación, mostrándose una pantalla u otra dependiendo de ello.

react-native-toast-message: Permite mostrar mensajes emergentes en la aplicación, informando de errores ocurridos, de acciones realizadas con éxito, o cómo avisos en otros casos.

react-native-bluetooth-state-manager: Este módulo facilita la gestión del estado del Bluetooth en el dispositivo. Es muy importante para la aplicación, dado que es necesario que esté activo para la detección de los beacons.

6.2.3 Pantallas

Pantalla Splash

La pantalla de bienvenida, también llamada Splash Screen, se muestra siempre que se abre la aplicación sin estar previamente ejecutándose en segundo plano. Tiene como objetivo mostrar una experiencia inicial atractiva para el usuario desde el primer momento mientras se carga la aplicación.

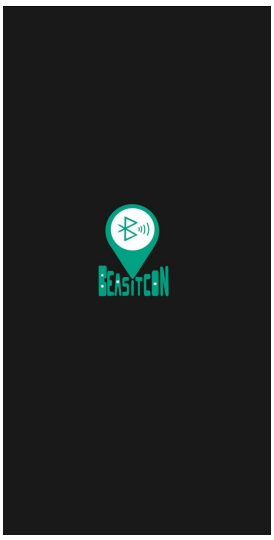


Figura 6.1: Splash Screen

Una vez cargada la aplicación, se redirigirá al usuario a una pantalla o a otra dependiendo de ciertos factores:

- **Onboarding,** si es la primera vez que se abre la aplicación tras su instalación.
- **Iniciar sesión,** si no se da el caso anterior y tampoco hay una sesión iniciada.

- **Interfaz principal - Mapa,** si no se da el primer caso pero el usuario tiene una sesión iniciada.

Onboarding

Cuando se abre la aplicación por primera vez, es decir, justamente tras instalarla, se muestra el Onboarding: una pantalla que proporciona una introducción de la aplicación, ayudando a los nuevos usuarios a familiarizarse con las características de esta. Se muestra una secuencia de cuatro diapositivas dentro de la pantalla, tal y como se aprecia:

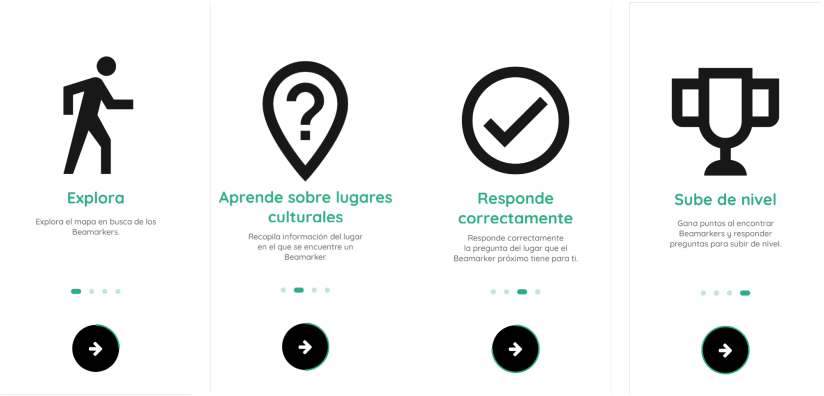


Figura 6.2: Diapositivas de la pantalla ‘Onboarding’

Para desplazarse por ellas, se permite deslizar la pantalla con el dedo hacia la izquierda o derecha, o bien pulsar en la flecha inferior. Ello irá cargando un borde alrededor de esta que indica el progreso, permitiendo al usuario inferir cuantas diapositivas le quedan por ver. Si se pulsa sobre la flecha de la última diapositiva, se redirigirá a la pantalla de ‘Iniciar sesión’, dando por finalizadas las explicaciones proporcionadas en esta.

Una vez se presenta la funcionalidad y el cometido de la aplicación a través del ‘Onboarding’, se indica por medio del localstorage que ya se ha cargado por primera vez la aplicación. Esto provocará que las próximas veces en las que se inicie la aplicación no se muestre de nuevo esta pantalla.

Inicio de sesión

Las dos pantallas que se detallan a continuación son accesibles siempre que no haya una sesión iniciada. En primera instancia, se muestra la pantalla de inicio de sesión, donde se puede observar:

- **Logo** de la app.
- Campo de texto relativo a **correo electrónico**.
- Campo de texto relativo a la **contraseña**.
- **Botón de inicio de sesión**. Al pulsarse, se lleva a cabo la validación de los campos y la autenticación del usuario en caso de que todo haya ido bien. En otro caso, emergerá un mensaje informando del error.
- Enlace “**¿No tienes cuenta? Regístrate**” que conduce a la pantalla de *Crear cuenta*.

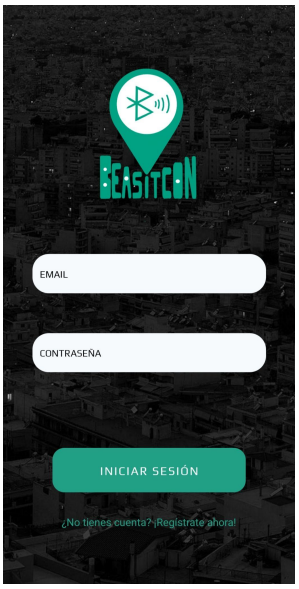


Figura 6.3: Inicio de sesión

Crear cuenta

La pantalla ‘Crear cuenta’ se presenta de una forma diferente a la de ‘Iniciar sesión’, pues se muestran los campos mínimos y obligatorios para la creación de un usuario en dos pasos. Se opta por esta disposición para evitar una sobrecarga de inputs en la interfaz, lo que podría resultar abrumador para el usuario.

En el primer paso se muestran los siguientes campos:

- **Correo electrónico:** Debe ser un correo electrónico válido.
- **Contraseña:** Debe tener más de 6 caracteres.
- **Confirmar contraseña:** Debe coincidir con el valor definido en ‘Contraseña’.

Figura 6.4: Crear cuenta (paso 1)

Al pulsar en ‘Siguiente’, se validan estos campos. Si existen errores, se muestra cada error bajo su respectivo input.

Si todo ha ido correctamente, se avanza al siguiente paso, que muestra:

- **Nombre:** No debe estar vacío.
- **Nombre de usuario:** Debe ser un nombre de usuario válido y no debe estar ya

registrado en el sistema.

Figura 6.5: 'Crear cuenta' (paso 2)

De forma similar al paso anterior, al pulsar en 'Enviar' se validan los dos campos. Si existen fallos, se indican como en el paso 1.

En caso de éxito, se crea la cuenta del usuario y se carga la interfaz principal. La nueva cuenta naturalmente se establece con la puntuación a 0, se asigna un avatar por defecto y no se proporciona un apellido. Estas dos últimas cuestiones comentadas pueden ser modificadas posteriormente por el usuario según sus preferencias.

Mapa

Es la interfaz principal de la aplicación una vez se inicia la sesión. Antes de mostrar la pantalla, es necesario realizar una consulta al servidor para obtener los datos, por lo que se dispone un indicador de carga para informar al usuario de ello.

Una vez se ha cargado completamente la información, se muestra la interfaz. En ella, se puede observar un mapa por el cual el usuario puede desplazarse libremente y en el que aparecen los Beamarkers asociados a las notificaciones:

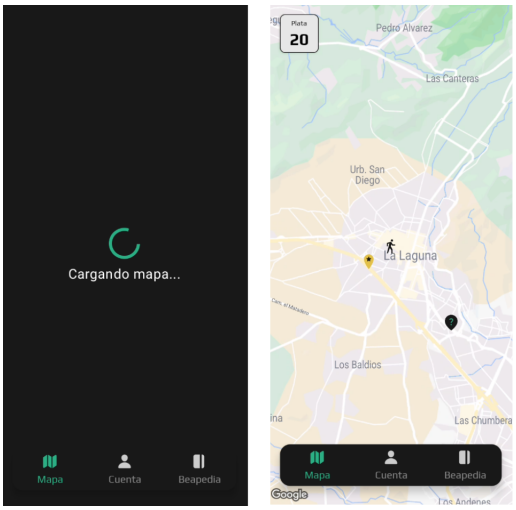


Figura 6.6: Mapa de juego

En un primer vistazo, se pueden observar numerosos elementos dentro del mapa. Se mostrarán unos u otros dependiendo de ciertos factores que se verán a continuación:

- **Posición del usuario**

  Se establece la posición del usuario como un icono de tipo ‘Persona’ que permite distinguirlo del resto de marcadores:

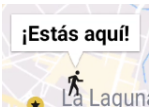


Figura 6.7: Icono posición del usuario

- **Puntuación del usuario**

  En la esquina superior izquierda, se muestra la puntuación y el nivel actual del usuario en un pequeño contenedor.El color del contenedor y el texto dependen del propio nivel en el que se encuentre, como se aprecia en el ejemplo siguiente:

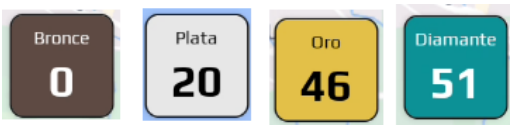


Figura 6.8: Puntuaciones en el mapa

- **Aviso de bluetooth desactivado**

  Dada la importancia del Bluetooth en la detección de los beacons, se habilita este botón para el caso en el que el dispositivo lo tenga deshabilitado, y se sitúa justo debajo de la puntuación del usuario.

Figura 6.9: Aviso de bluetooth desactivado

Al pulsar en él, se abre un diálogo informando de la relevancia de tener esta funcionalidad activa y se permite activarlo desde el propio diálogo.

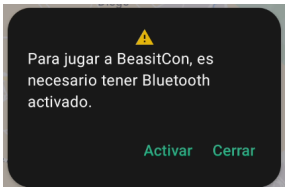


Figura 6.10: Diálogo de bluetooth desactivado

- **Notificación de beacon próximo**

  Con el Bluetooth activado, el dispositivo se pone a la escucha de beacons que pertenezcan al proyecto, no estén visitados ya y estén en curso, acorde al apartado de Algoritmo de detección de los Beacons. Cuando se cumplen las condiciones, aparece un botón en el mismo lugar que el de ‘Aviso de bluetooth desactivado’:

Figura 6.11: Aviso de notificación encontrada

Al pulsar en él, emergerá la notificación que contiene la pregunta:

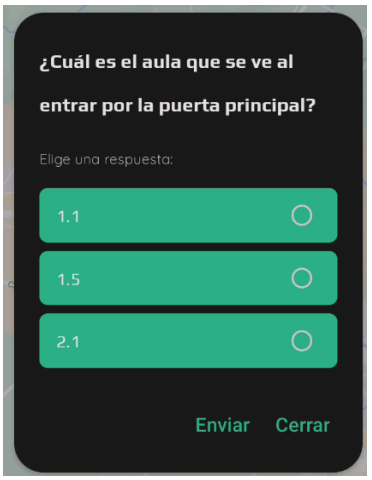


Figura 6.12: Pregunta de la notificación emergente

Si el usuario trata de responder sin haber seleccionado una respuesta, se le alertará de ello y no se le permitirá avanzar en el proceso de respuesta.

Si el usuario responde, se comprueba si la respuesta es correcta. En este punto se pueden dar dos casos.

a. Si el usuario acierta, se le indica con un mensaje de éxito, junto a la cantidad de puntos que obtiene.

b. Si el usuario se equivoca al responder, se le indica con un mensaje de error, junto a la cantidad de puntos que obtiene si es el caso. Ello depende del nivel de dificultad asociado a la notificación que se está respondiendo.

En cualquier caso, tras terminar el proceso, se actualiza el mapa y la puntuación del usuario, acorde a lo definido en sus propias secciones.

- **Beamarkers**

Se coloca un Beamarker en el mapa por cada lugar registrado en el sistema. Cada lugar tiene asociados de 1 a muchas notificaciones, que son las que el usuario visita realmente al acudir a él.

Al pulsar en uno de ellos, se muestra una pequeña sección con información relevante, como la distancia a la que está el usuario del lugar y la cantidad de notificaciones visitadas. Esto último también tiene un impacto en el icono del Beamarker que se muestra en el mapa:

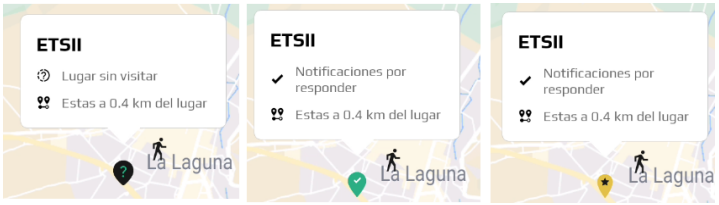


Figura 6.13: Posibles estados del Beamarker

Además, se muestra también en la esquina superior derecha información sobre las pistas, pudiendo consultar información de cada una:

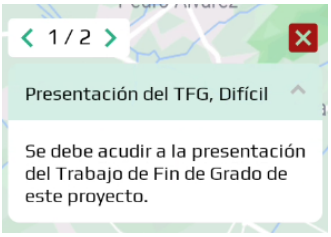


Figura 6.14: Menú de pistas de notificaciones

- **Menú de navegación**

  La navegación a través de las diferentes pantallas de la aplicación se consigue mediante un menú inferior.

Figura 6.15: Menú de navegación inferior

Cuenta

En 'Cuenta', el usuario podrá consultar toda la información relacionada con su perfil. Una vez se carga la pantalla, se muestra:

- Una **cabecera** con el nombre de usuario, un icono a la izquierda que permite regresar al mapa y un icono a la derecha que al ser pulsado despliega un menú desde el cual se puede cerrar sesión.
- La **Beacard** del usuario, en la cual se muestra:
  - **Nivel actual** sobre su color representativo.
  - **Datos del usuario: Avatar, nombre y apellido.**
  - **Ranking del usuario.**

  Esta proporciona un modo consulta y un modo edición, pudiéndose iterar entre ellos a través los iconos que aparecen por encima del componente.

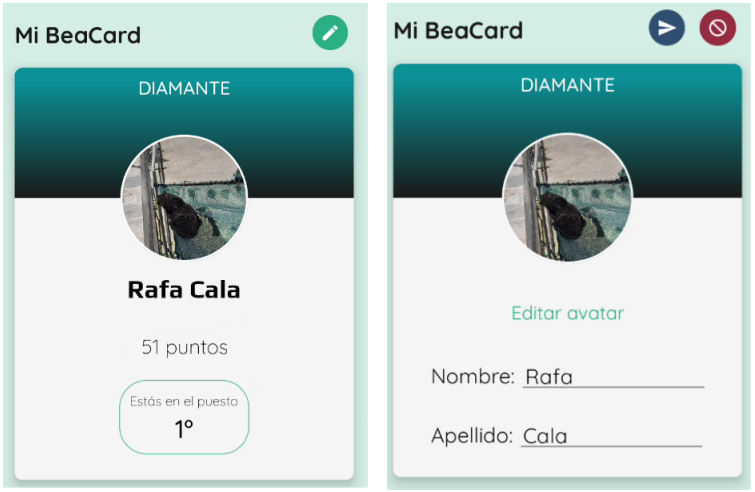


Figura 6.16: Modos del BeaCard

- **Estadísticas del usuario.**
- **Eventos que ha visitado.** Al pulsar en él se abre un diálogo en el que se puede ver el detalle del evento.

Figura 6.17: Estadísticas de la pantalla 'Cuenta'

Beapedia

La Beapedia es la pantalla que permite visualizar la información de los lugares registrados en el sistema junto a todo su detalle. Se divide en dos partes:

1. Un **listado** que muestra los lugares y los eventos en curso, y que permite acceso al detalle de cada uno pulsando en su respectivo ítem.

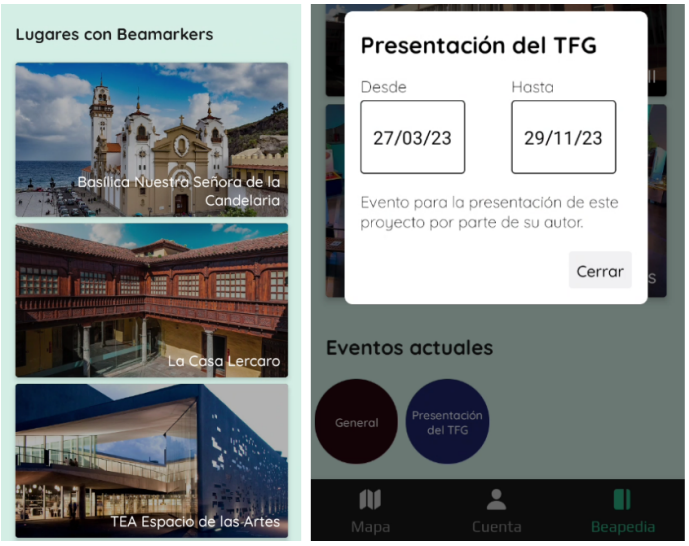


Figura 6.18: Listado de la Beapedia

2. La vista de **Detalle**, que ofrece información del lugar en mayor profundidad, como

una descripción del lugar, las notificaciones visitadas y las que quedan por visitar.

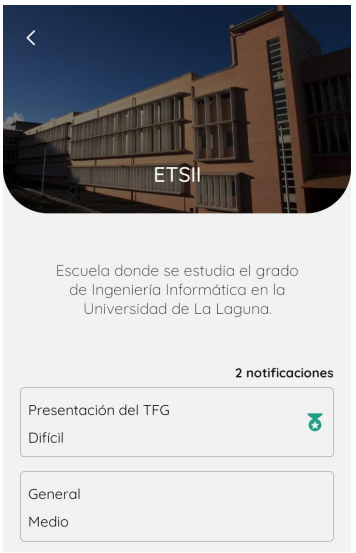


Figura 6.19: Detalle de la Beapedia

6.3 Aplicación web

6.3.1 Aspectos a tener en cuenta

Despliegue de la aplicación

Se ha optado por llevar a cabo su despliegue en la plataforma **Vercel**. De este modo, los administradores tendrán acceso a la aplicación en todo momento con simplemente entrar al dominio proporcionado por dicha plataforma. [A5]

Responsividad

Teniendo en cuenta la accesibilidad total que se le pretende dar a la aplicación web, se ha prestado especial atención por contar con un diseño responsivo que aporte una buena experiencia de usuario, independientemente del dispositivo con el que esté trabajando. Como prueba de ello, se adjunta una imagen desde la herramienta Responsively con diferentes resoluciones de una de las pantallas del proyecto. [A6]

6.3.2 Pantalla de inicio de sesión

Se tiene como punto de partida una pantalla de inicio de sesión que contiene un pequeño formulario con los campos:

- **Correo electrónico**
- **Contraseña**

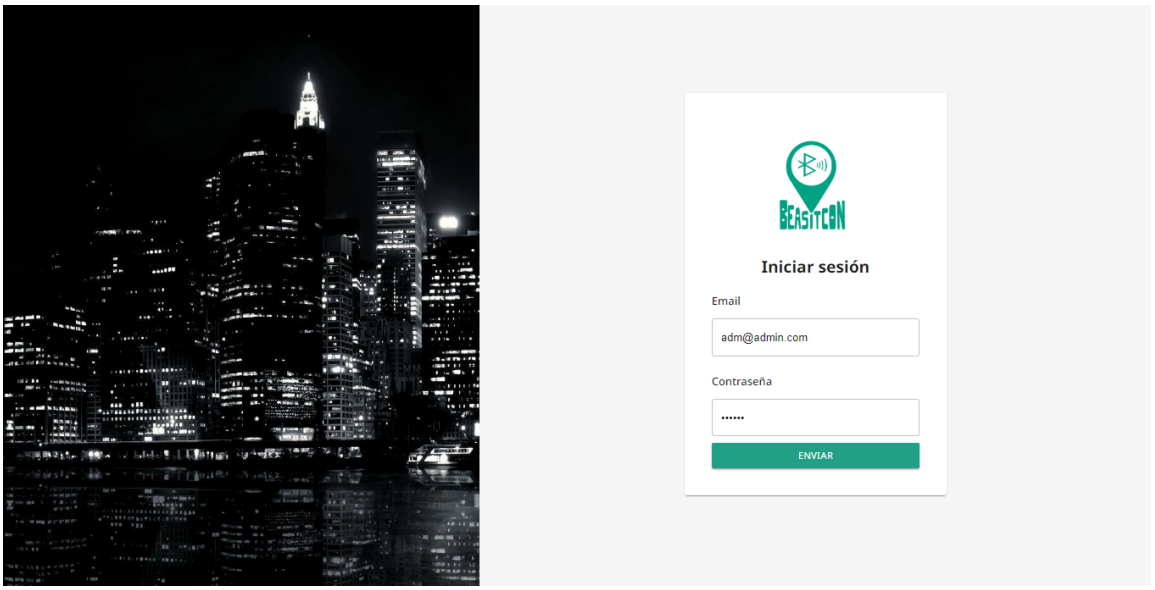


Figura 6.20: Interfaz de inicio de sesión en la aplicación web

Al pulsar en el botón ‘Enviar’, se comprueba que el correo electrónico y la contraseña sean válidos, y pertenezcan a un usuario registrado, que además debe contar con el rol de administrador.

Si el usuario que está iniciando sesión no es administrador u ocurre algún otro tipo de error, no se pasará la validación y se mostrará una notificación indicando la razón.

Si la validación se da correctamente. Se redirigirá al usuario a la interfaz principal del proyecto. Si bien inicialmente la pantalla mostrada es la de ‘Configuración’, se cuenta con un menú lateral que da acceso a otras ocho pantallas. Se comentará el propósito de cada una de ellas más adelante.

6.3.3 Funcionalidad general de la interfaz principal

A excepción de **Almacenamiento, Niveles de usuario y Notificaciones**, que han requerido una lógica más concreta, el resto de pantallas presentan una funcionalidad general que se sustenta sobre un flujo y una serie de componentes comunes en los que vale la pena profundizar para una mejor comprensión de estas. A continuación, se presentarán dichos componentes al mismo tiempo que se describe el flujo nombrado.

Header

La interfaz principal tiene en todo momento una cabecera sencilla con los siguientes elementos:

- A la izquierda, un icono ‘Menu’ que al pulsarse **despliega el menú lateral** de navegación.
- En el centro, el **logo del proyecto**.
- A la derecha, un icono que al pulsarse **cierra la sesión** del usuario.

Se le aplica un poco de espaciado y sombreado al componente para dar la sensación de que flota en la pantalla.

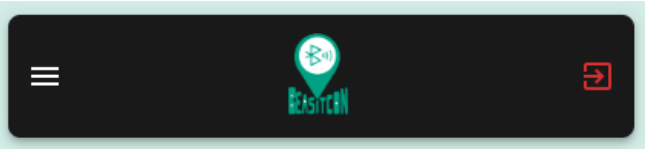


Figura 6.21: Componente Header

Menú lateral

El menú lateral permite la navegación por los distintos módulos del proyecto sobre los que se pueden realizar acciones. De igual forma que el **Header**, se muestra de forma "flotante". Se destaca la vista que está activa con el color característico de la aplicación.

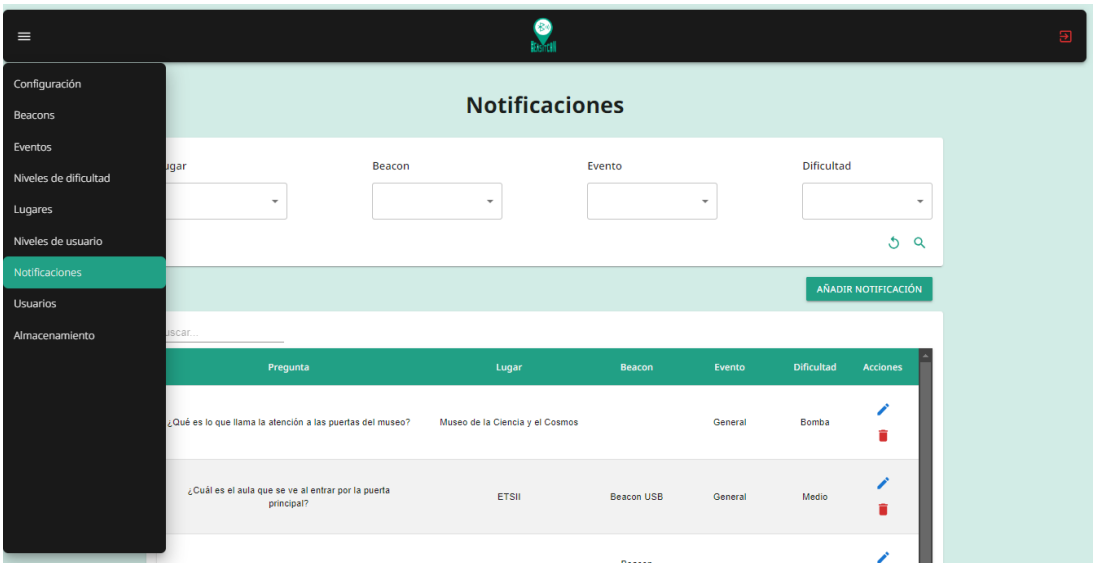


Figura 6.22: Menú lateral

ModuleView

En esencia, al acceder a cada pantalla desde el menú, se renderiza un layout con el **título** de la colección con la que se quiere operar y un **listado** inicial de sus documentos.

Figura 6.23: Ejemplo de ModuleView

LoadingOverlay

Mientras se obtiene la información del listado, se muestra un **indicador de carga** para hacer notar al usuario que aún se está llevando a cabo el proceso.

Este indicador también es desplegado cuando se realiza cualquier otra operación contra la base de datos.

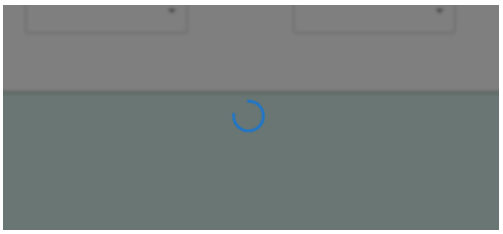


Figura 6.24: Ejemplo de LoadingOverlay activo

Se puede notar que mientras está activo el **LoadingOverlay**, se aplica una capa de oscurecimiento y difuminado al resto de la pantalla.

CustomTable

Este listado se muestra en forma de **tabla**. Internamente, el **CustomTable** recibe del **ModuleView**: el módulo, los datos, las columnas relevantes a mostrarse en la tabla y las funciones colaterales que se ejecutarán al realizar cualquiera las operaciones de creación, modificación o eliminación de las que dispone a través de iconos que se muestran, como se puede apreciar:

Figura 6.25: CustomTable de la vista ‘Eventos’

La tabla cuenta además con un campo de búsqueda, de modo que se filtran los elementos que tengan información que coincida con lo definido en él:

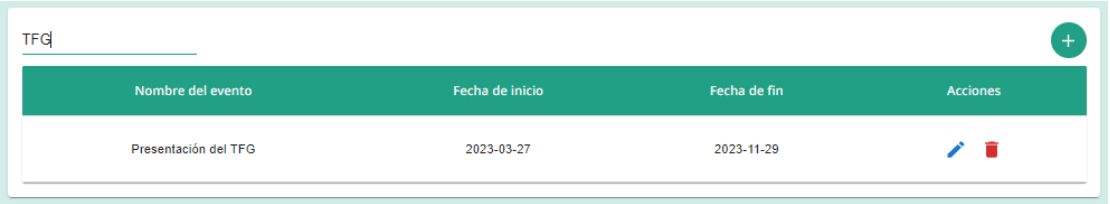


Figura 6.26: CustomTable de la vista ‘Eventos’ haciendo uso del filtro

Confirmación de eliminación

Cuando se pulsa en el icono de eliminación antes de realizar directamente la operación, se abre un diálogo de confirmación de la misma, a fin de certificar la intención del usuario y evitar eliminaciones accidentales. Al confirmar la eliminación, desaparece la fila de la

tabla.

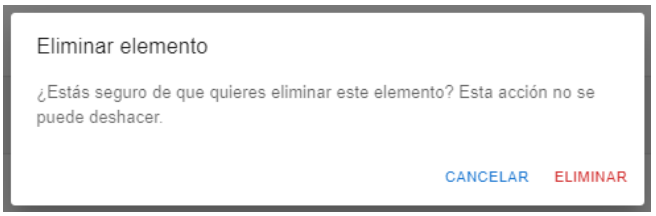


Figura 6.27: Diálogo de confirmación de eliminación de elemento

GenericDetailView

Tanto si se pulsa en el botón '+' para crear un nuevo elemento como si se pulsa en el icono de editar para modificar uno existente, se redirige la página a una pantalla de detalle, que cuenta con:

- El título de **Detalle**, acompañando a un icono '<-' que permite la vuelta a la vista del listado.
- Un formulario **GenericForm** con los campos a modificar de la colección sobre la que esté operando.
- Botones de 'Guardar' y 'Cancelar' para realizar las correspondientes acciones.

La tabla le pasa al **GenericDetailView** todos los datos necesarios para mostrarse como corresponde, como son: el nombre de la colección, los campos a modificar así como sus tipos, y los datos que contiene el elemento que se va a consultar, en el caso de no tratarse de un elemento a crear.

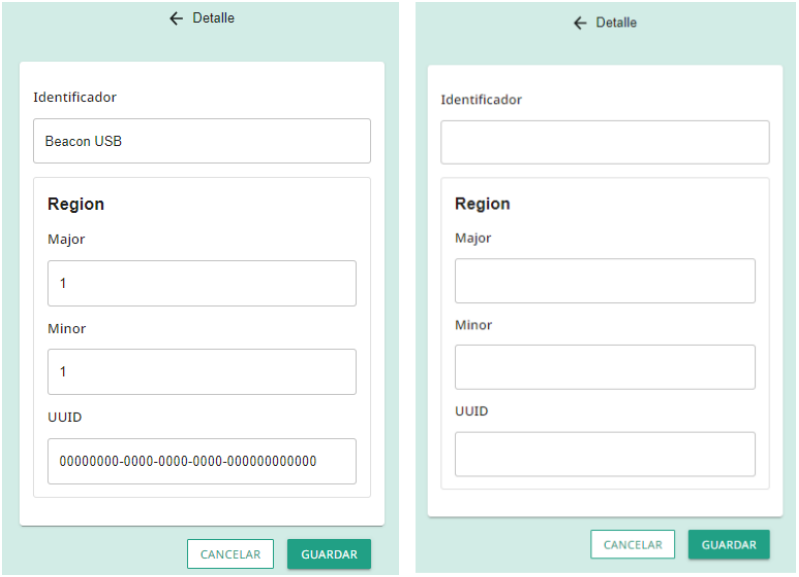


Figura 6.28: GenericDetailView de la colección Beacons

GenericForm

El **GenericForm** es el formulario que se construye en la vista de Detalle a partir de la información de los campos, su etiqueta correspondiente, y sus tipos, enviada por la tabla. Este formulario se rellenará con datos si se trata de la edición de un elemento existente, o vacío en el caso de que se vaya a crear uno.

Como se venía adelantando, el formulario mapeará los campos y renderizará un input, apropiado para cada uno de ellos. También se situará por encima una etiqueta que ayude a identificar cada campo. La lista de inputs queda de la siguiente forma:

- **Booleano:** Este input se utiliza para aquellos campos de las colecciones que sean verdadero o falso.

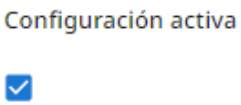


Figura 6.29: Input de tipo ‘Booleano’

- **Texto:** Este input renderiza un campo de texto típico.

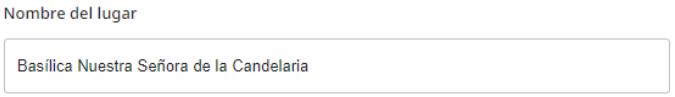


Figura 6.30: Input de tipo ‘Texto’

- **Área de texto:** Es un input similar al tipo ‘Texto’ con la diferencia de que a medida que va incrementando la cantidad de texto en él, va creciendo su altura para que se pueda seguir visualizando. Es un input idóneo para descripciones, que normalmente contienen una mayor cantidad de información.

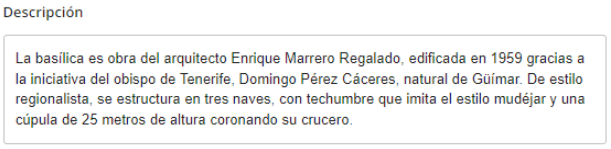


Figura 6.31: Input de tipo ‘Área de texto’

- **Numérico:** Es un input usado para representar valores numéricos. Está preparado para solo permitir caracteres de este tipo.

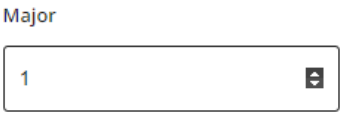


Figura 6.32: Input de tipo ‘Numérico’

- **Fecha:** Es un input apropiado para los campos que contienen fechas, pues solo permite ese formato.

Figura 6.33: Input de tipo ‘Fecha’

- **Coordenadas:** Es un input personalizado para las coordenadas de los Lugares. Permite definir coordenadas de dos formas:

Asignación por búsqueda: Se escribe el lugar del que se quiere obtener las coordenadas. A medida que se va escribiendo, se ejecuta una búsqueda que arroja posibles resultados.

Cuando se selecciona una de las opciones, se toman las coordenadas del lugar y se escriben automáticamente en la latitud y la longitud del input, así como el nombre del **último lugar buscado**.

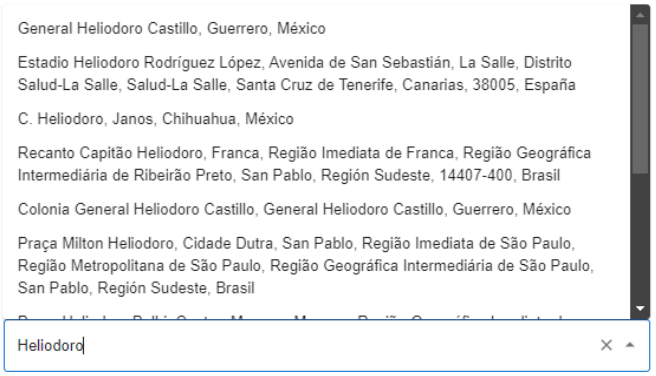


Figura 6.34: Asignación por búsqueda en el input de tipo ‘Coordenadas’

Asignación manual: Si se pulsa en ‘Asignar manualmente’, se abrirá un diálogo en el que se podrán pegar directamente las coordenadas en un input. Este resulta especialmente útil si se han copiado directamente desde Google Maps. Al pulsar en ‘Asignar’, se valida que sean coordenadas correctas y se introducen en los campos del texto latitud y longitud, mientras que en **último lugar buscado** aparecerán las coordenadas conjuntamente.

Figura 6.35: Asignación manual en el input de tipo ‘Coordenadas’

Una vez terminada la asignación de cualquiera de las dos formas, el input queda de la siguiente manera:

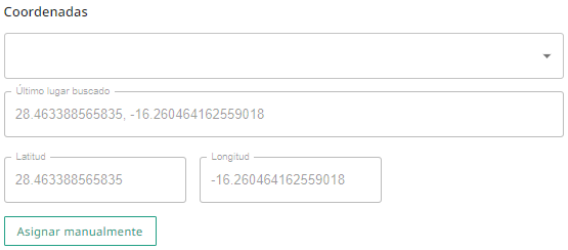


Figura 6.36: Resultado de asignación en el input de tipo ‘Coordenadas’

- **URL de imagen:** Es un input útil para aquellos campos de los documentos que contengan URL de imágenes. El botón a la derecha del input permite previsualizar

la imagen, y advertir al usuario de que no existe la imagen en caso de error.

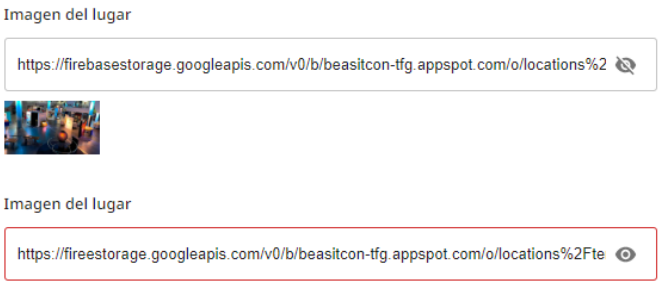


Figura 6.38: Input de tipo 'URL de imagen'

- **Cálculo:** Es un input que recibe una función de cálculo, y que es ejecutada cuando se pulsa el botón que contiene a su derecha. Se usa para el campo **Máxima puntuación alcanzable** de la colección **Configuration**, que es la suma de las máximas puntuaciones que se pueden obtener con cada notificación. Como se trata de un valor que se obtiene a partir de un cálculo, el input es **no** editable.

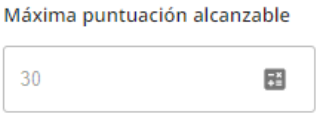


Figura 6.39: Input de tipo 'Cálculo'

- **Objeto:** Es un input que contiene varios inputs. Cuando se recibe un campo de este tipo, se genera un contenedor sobre el que se generarán a su vez inputs para los campos dentro del objeto. Sucede por ejemplo con el campo **region** de la colección **Beacons**, que es un objeto que contiene dentro el uuid, major y minor.

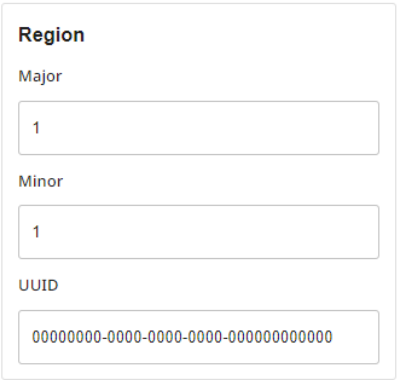


Figura 6.40: Input de tipo 'Objeto'

Validaciones

Al guardar las modificaciones realizadas en los inputs correspondientes a los campos de las colecciones, se realiza una validación previa de los mismos. A través del nombre del módulo que recibe el **GenericDetailView**, obtiene un objeto con la validación que se debe ejecutar en cada campo. El algoritmo de validación comprende tres posibles casos de validación: [A7]

- **Requerido:** Comprueba que el campo no esté vacío.
- **Expresión regular:** Comprueba que el campo cumpla con cierta expresión regular.
- **Personalizado:** Comprueba que el campo cumpla con una función personalizada.

Para cada uno de los tres casos, se define el mensaje de error que se debe mostrar en caso de darse sin éxito la validación.

Cuando se dan errores en la validación, estos son mostrados debajo de los inputs que lo contienen con un leve parpadeo para que llame la atención del usuario. Además, se cancela el proceso de guardado. Esto último se indica en una notificación de error emergente.

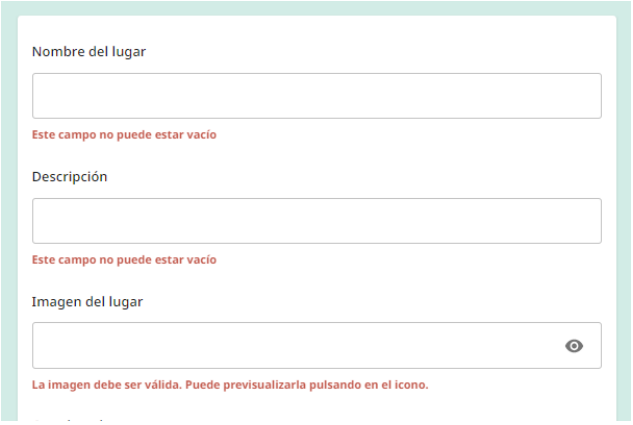


Figura 6.41: Ejemplo de mensajes de error en las validaciones de Lugares

Guardado

En caso de haberse dado correctamente las validaciones, se continúa con la fase de guardado. Durante el proceso, se toma cada uno de los campos y se tratan los datos acorde a su tipo para que sean manejables para **Firestore.**

Si todo se da correctamente, se guarda el elemento en la colección. En caso de tratarse de un nuevo elemento, se redirige de nuevo al listado del módulo.

Notificaciones

En cada una de las operaciones que se realizan durante la interacción con las pantallas, aparece una notificación para indicar al usuario el éxito, el fallo o la información de relevancia como respuesta a la acción que ha hecho.

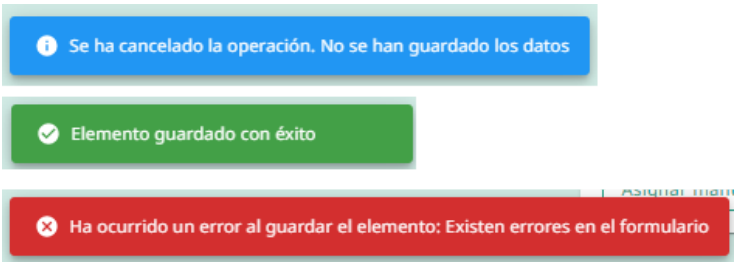


Figura 6.42: Ejemplos de notificaciones

Enrutamiento

Se tiene una navegación de rutas controlada y homogénea a lo largo de todo el proceso comentado, resultando en una definición lógica del enrutamiento de la aplicación web:

- Si no hay un usuario autenticado, se redirige automáticamente a la pantalla de **Inicio de sesión**.
- Todas las rutas, a excepción de la de inicio de sesión, están **protegidas**. Es decir, no son accesibles a menos que haya un usuario autenticado.
- Todas las rutas protegidas están envueltas en un **Layout** que contiene el **Header** y la información del módulo al que se está accediendo.
- Cuando se inicia sesión, se navega por defecto a la pantalla de **Configuración**.
- Las rutas para las vistas de **Listado** contienen únicamente el nombre de la colección. Por ejemplo, la vista de **Listado** de la colección **Beacons** es "*/beacons*".
- La vista de **Detalle** tendrá una ruta u otra dependiendo de si se trata de una creación o una modificación. Si es un nuevo elemento, la estructura será "*coleccion/new/details*", mientras que si es una modificación la ruta será "*coleccion/:id/details*", donde :id es el id del documento a actualizar.

6.3.4 Pantallas de la interfaz principal con funcionalidad general

Las siguientes pantallas siguen al completo la funcionalidad general especificada en el apartado anterior. Se comentarán brevemente las columnas utilizadas en sus tablas y los campos que se pueden modificar en el detalle.

Configuración

Esta pantalla muestra las configuraciones disponibles. Cada configuración contiene parámetros generales que son usados por la aplicación. La razón por la que se permiten tener varias configuraciones es porque puede resultar interesante tener varias sobre las que alternar. Por ejemplo, se podría disponer de una configuración más apropiada para la época navideña, donde el avatar por defecto puede con un gorro navideño y la imagen de lugar por defecto una imagen con regalos.

Columnas de la tabla:

- Configuración activa
- Fecha última modificación

Campos del formulario:

- Imagen de lugar por defecto: Debe ser una imagen válida.
- Avatar por defecto: Debe ser una imagen válida.
- Máxima puntuación alcanzable: No puede estar vacío.
- Configuración activa

Solo puede haber una activa, por lo que cuando se crea o se modifica una configuración y se deja la configuración activa, se desactiva la que estuviese activa actualmente.

Es importante destacar que los datos de la configuración que esté activa se almacena a nivel global dentro de la aplicación, ya que está información es de utilidad en ciertas partes de la aplicación web, como se verá más adelante.

Beacons

Esta pantalla muestra los beacons con los que se cuentan en el proyecto. Cuando en la aplicación móvil aparece un beacon, se comprueba mediante estos campos que el beacon pertenece al sistema.

Columnas de la tabla:

- Identificador
- Major
- Minor
- UUID

Campos del formulario:

- Identificador: No puede estar vacío.
- Major: No puede estar vacío.
- Minor: No puede estar vacío.
- UUID: No puede estar vacío.

Al guardarse, los campos **major, minor y uuid** se almacenan dentro del objeto **region** que lo caracteriza. No se permite la eliminación de un beacon que esté asociado a una notificación.

Lugares

Esta pantalla muestra los lugares registrados en el proyecto. Los Beamarkers que aparecen en el mapa de la aplicación se sitúan en el mapa en base a las coordenadas. Estos lugares corresponden también a los lugares que se pueden consultar en la Beapedia.

Columnas de la tabla:

- Nombre del lugar

Campos del formulario:

- Nombre del lugar: No puede estar vacío.
- Descripción: No puede estar vacío.
- Imagen del lugar: Debe ser una imagen válida.
- Coordenadas: Deben tener una longitud y una longitud comprendidas entre los valores [30, -30]

No se permite la eliminación de un lugar que esté asociado a una notificación.

Eventos

Esta pantalla muestra los eventos o las también llamadas campañas. Las notificaciones tienen siempre un evento asociado, con el que se permite determinar si la notificación está en curso. Los eventos además disponen de una descripción que le da sentido. Se tiene un evento ‘General’ para las notificaciones que están siempre disponibles en la aplicación.

Columnas de la tabla:

- Nombre del evento
- Fecha de inicio

- Fecha de fin

Campos del formulario:

- Nombre del evento: No puede estar vacío.
- Descripción
- Fecha de inicio: Debe ser una fecha válida.
- Fecha de fin: Puede ser vacío o una fecha válida.

No se permite la eliminación de un lugar que esté asociado a una notificación.

Niveles de dificultad

Esta pantalla muestra los niveles de dificultad que hay para las preguntas de la notificación. Según el nivel y si se acierta o no, se obtendrá una cantidad de puntos diferente de puntos que se encuentra parametrizada y configurable.

Columnas de la tabla:

- Nivel
- Puntos por acierto
- Puntos por fallo

Campos del formulario:

- Nivel: No puede estar vacío.
- Puntos por acierto: Debe ser un número entero no negativo.
- Puntos por fallo: Debe ser un número entero.

Nótese que en 'Puntos por fallo' se permiten números enteros negativos. Esto puede resultar interesante si se quisieran incorporar preguntas penalizables. No se permite la eliminación de un lugar que esté asociado a una notificación.

6.3.5 Pantallas de la interfaz principal con funcionalidad específica

Se exponen las pantallas de los módulos cuya lógica no puede ser tratada con la funcionalidad general comentada y, en consecuencia, requieren una implementación y diseño específicos.

Niveles de usuario

Esta pantalla muestra la configuración de niveles que puede obtener el usuario a medida que obtiene puntos. Este nivel influye en cómo se muestran ciertos elementos de la aplicación móvil, lo que añade una mayor personalización en su interfaz.

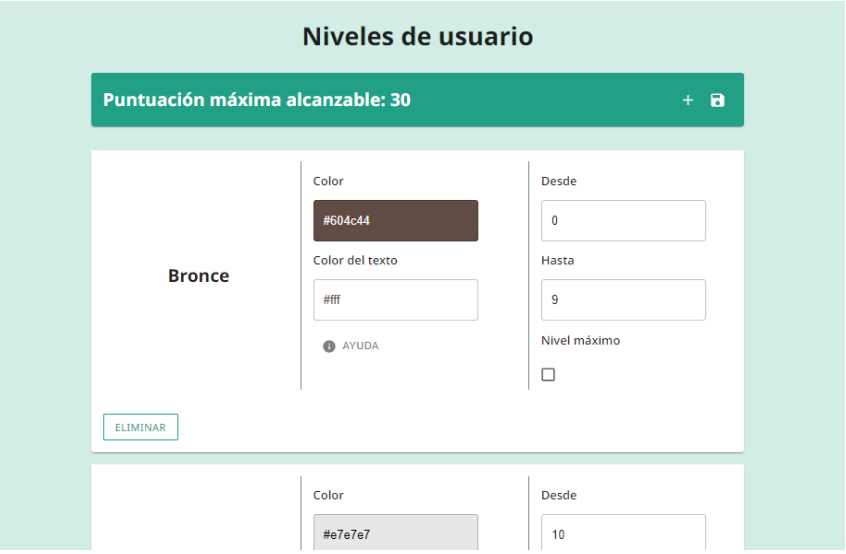


Figura 6.43: Pantalla Niveles de usuario

A diferencia de las pantallas expuestas, esta solo tienen una única vista de listado, en la que se llevan a cabo todas las acciones. Se muestra una cabecera dentro de la vista que muestra la máxima puntuación alcanzable, dadas las notificaciones registradas para que el administrador tenga a su disposición dicha información de cara a la modificación de niveles. A la derecha de la cabecera están los botones de 'Añadir' y 'Guardar'.

Debajo de la cabecera se sitúa el listado de niveles, ordenado por puntuación.

- **Nivel de Usuario**

  Cada nivel se muestra dentro de una tarjeta donde se exponen los siguientes campos:

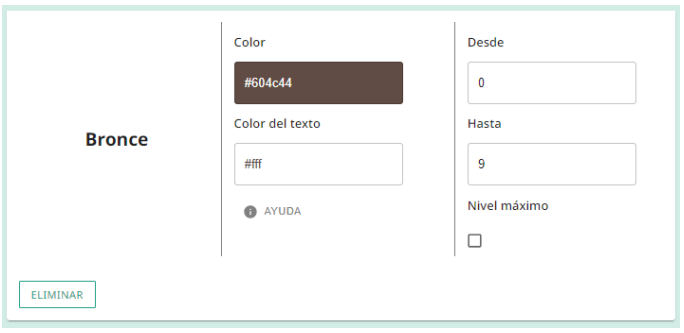


Figura 6.44: Nivel de usuario

- **Nivel:** Nombre identificador del nivel. Aparentemente es un texto, pero si se pulsa en él es posible su edición.
- **Color**: Corresponde al color de fondo que representa al nivel. Al pulsar sobre el input, se despliega una pequeña ventana para establecer un color. Al seleccionar un color, este se guarda como color de fondo del input, y el valor hexadecimal del color escogido se mostrará con el color del texto definido abajo. De esta forma, permite al usuario ver con facilidad como combinan ambos colores.
- **Color del texto**: Corresponde al color del texto que representa al nivel. Funciona de la misma forma expuesta en el punto anterior.

Figura 6.46: Apartado de selección de colores en el nivel de usuario

- **Ayuda**: Este, si bien no es un campo del nivel, es un tooltip que provee una explicación sobre cómo funcionan estos inputs de selección de colores. Esta información se muestra al situar el cursor en él.

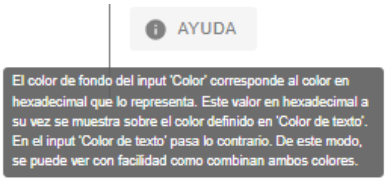


Figura 6.47: Tooltip de ayuda en el apartado de selección de colores

- **Desde**: Es un campo de texto numérico que representa la puntuación mínima del nivel.
- **Hasta**: Es un campo de texto numérico que representa la puntuación máxima del nivel.
- **Nivel máximo**: Es un checkbox que se usa como indicador de que el nivel es el máximo que se puede alcanzar. Cuando este está activo, se inhabilita el campo de texto del Hasta y se elimina su valor.

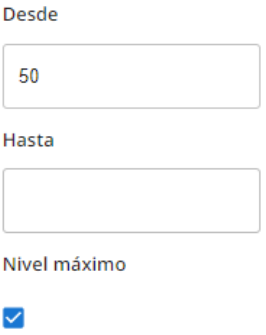


Figura 6.48: Apartado de puntuaciones del nivel

- **Reglas del sistema de niveles**

  Existen unas reglas en el sistema de niveles que deben seguirse para mantener un comportamiento controlado y coherente dentro de la aplicación móvil. Esto implica una dependencia entre los niveles, por lo que al realizarse un guardado, se valida todo el sistema en su conjunto y si la validación es correcta, se guardan los niveles modificados y creados, y se eliminan aquellos que hayan sido marcados para ello. Las reglas se excluyen de los niveles a eliminar, y son las siguientes: [A8]

- El valor **Desde** del primer nivel **siempre** debe ser 0
- El último nivel no debe tener **Hasta** y sí debe tener el **Nivel máximo** activo.
- El valor **Hasta** de un nivel debe ser siempre el valor **Desde - 1** del siguiente nivel, a excepción del último nivel, que no debe tener **Hasta.**

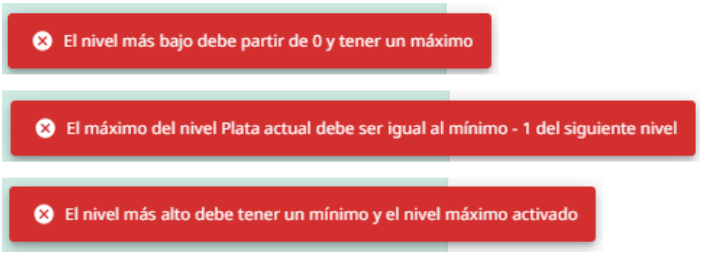


Figura 6.49: Incumplimiento de las reglas de niveles de usuario

- **Añadido de niveles**

  El añadido de niveles se da de la siguiente manera. Pulsando en el icono '+' de la cabecera, se añade un nivel con valores por defecto con un color de fondo azulado. Una vez se termina de modificar los valores que se requieren para el nivel, se pulsa en 'Añadir a la configuración'. Esto internamente cambiará la naturaleza del nivel de 'Nuevo en edición' a 'A añadir', colocándolo en el sistema de niveles finales y siguiendo el orden del listado comentado. Esto implica también que desaparezca el botón 'Añadir a la configuración' en el nivel nuevo, pero se mantiene el color azulado para indicar al usuario que se trata de un nuevo nivel.

Figura 6.50: Nivel por añadir

- **Eliminación de niveles**

  A la hora de eliminar existen dos casos: la eliminación de un nivel que ya está persistido con anterioridad y la de un nivel que está por añadirse.

  En el primer caso, al pulsar en 'Eliminar', se sombrea el nivel e internamente se le coloca una marca que indica que es un nivel que se va a eliminar una vez se realice el guardado general. Este estado provoca que el botón 'Eliminar' se sustituya por 'Cancelar eliminación', lo que devuelve el nivel a su estado original si este botón es pulsado.

Figura 6.51: Nivel por eliminar

Cuando se da el segundo caso, si se pulsa en ‘Eliminar’, se le solicita al usuario una confirmación de la operación, y si este acepta, desaparece el nivel de usuario que estaba por añadirse al sistema.

- **Guardado de niveles**

  A la hora de realizar el guardado por medio del botón ‘Guardar’ situado en la cabecera se realizan los siguientes pasos:

  1. Se extraen del sistema de niveles aquellos que se han marcado de la eliminación, ya que no tiene sentido tenerlos en cuenta para la validación de las reglas.
  2. Se hace una validación general del sistema de niveles en base a las reglas descritas anteriormente.
  3. Si la validación es exitosa, se realiza una última comprobación consistente en evaluar si el sistema de niveles excede la puntuación máxima alcanzable. Si este es el caso, se informa al usuario de ello, pero se continúa con el proceso, ya que se entiende que está condición no rompe el funcionamiento de la aplicación.

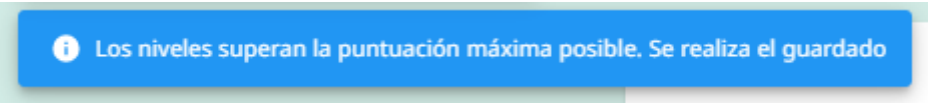


Figura 6.52: Notificación de superación de la puntuación máxima

  4. En este punto se tiene una configuración de niveles segura y lógica, por lo que se puede proceder a las operaciones de base de datos. Es decir, se aplican las actualizaciones sobre los documentos modificados, se crean los nuevos niveles y se eliminan los que estaban marcados para ello. Este proceso al conjunto debe ser atómico, por lo que si ocurre algún error se cancelan todas las operaciones. Si todo ha ido bien, se actualiza el sistema de niveles.

Notificaciones

Dado que esta colección es susceptible de albergar una cantidad mucho mayor de documentos que las expuestas, se opta por prescindir de la carga inicial de las mismas y se incluyen filtros para realizar la consulta. Estos filtros corresponden a las asociaciones

de las notificaciones con las otras colecciones y son en realidad selectores de los elementos de sus respectivas colecciones.

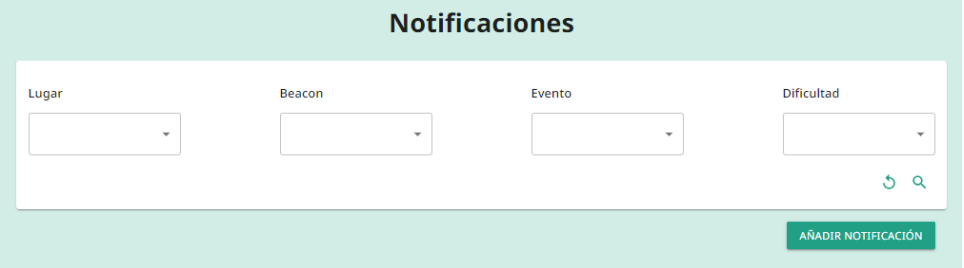


Figura 6.53: Filtros de las notificaciones

Se tiene un icono para reiniciar los filtros y otro para lanzar la búsqueda. Debajo del contenedor de los parámetros de búsqueda está también un botón ‘Añadir notificación’ que lleva a una vista de **Detalle** sin datos.

Cuando se realiza una búsqueda, se muestra la lista de notificaciones resultantes en la tabla expuesta en la funcionalidad general:

Columnas de la tabla:

- Pregunta
- Lugar
- Beacon
- Evento
- Dificultad

La vista de **Detalle** en este caso no lleva al **GenericDetailView**, sino que el formulario de esta colección tiene un diseño más adecuado a su lógica.

Campos del formulario:

- Lugar: Debe haber uno seleccionado
- Beacon
- Evento: Debe haber uno seleccionado
- Dificultad: Debe haber uno seleccionado
- Pregunta: No debe estar vacío.
- Pista
- Respuestas: Deben haber al menos dos respuestas y solo una debe ser correcta.

Dada la limitación física de los beacon, se permite que las notificaciones coexistan en la aplicación sin que tengan un Beacon asociado y se deja en manos de los administradores realizar esta gestión. Del mismo modo, la asociación entre beacons y notificaciones es 1 a 1. Es decir, un beacon solo puede estar asociado a una notificación. Es por ello que la asignación del beacon tiene una validación concreta a la hora de crear o modificar notificaciones.

Si al guardar se trata de asignar un beacon que ya está asociado a otra notificación, se avisa al usuario si aún así quiere realizar la asignación. Si la respuesta es no, se lleva a cabo el guardado del resto de modificaciones hechas en la notificación. Por otro lado si el usuario responde que sí, se procede a la desasignación del beacon en la otra notificación

junto al resto de modificaciones.

Usuarios

La utilidad de la pantalla para los Usuarios radica en las correcciones que se pueden aplicar sobre las visitas realizadas por el usuario y su puntuación y al mismo tiempo resulta de suma utilidad para las pruebas durante el desarrollo en el proyecto, ya que ofrecen una forma dinámica y sencilla de aplicar cambios.

Se muestra una tabla con los usuarios al igual que las pantallas que siguen la funcionalidad general.

Columnas de la tabla:

- Nombre de usuario
- Puntuación

Sin embargo, la vista de **Detalle** para esta colección se ha actualizado acorde a sus necesidades.

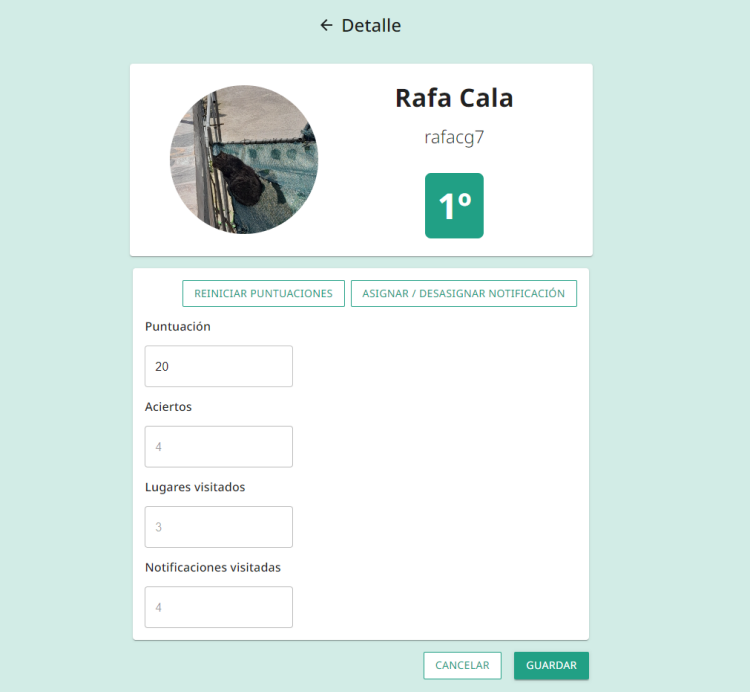


Figura 6.54: Detalle del usuario

En primer lugar, se observa la tarjeta del usuario, en la que se muestra los datos:

- **Avatar**
- **Nombre** y **apellido**
- **Nombre de usuario**,
- **Ranking** calculado en base a las puntuaciones de todos los usuarios.

Estos campos no son editables para el administrador por el simple hecho de que a excepción del ranking, que siempre es un valor calculado, el resto de campos son modificables por el usuario en la aplicación móvil.

La segunda parte contiene los datos sobre la puntuación obtenida y las visitas que ha realizado el usuario. La puntuación puede modificarse de forma manual directamente en su campo, pero para realizar modificaciones de los otros campos se debe hacer uso de los botones disponibles encima del contenedor.

- **Reiniciar puntuaciones**

  Este botón pone a 0 todos los campos, devolviendo las puntuaciones del usuario a su estado inicial.

- **Asignar / Desasignar notificación**

  Cuando se pulsa en el botón, se abre el siguiente diálogo:

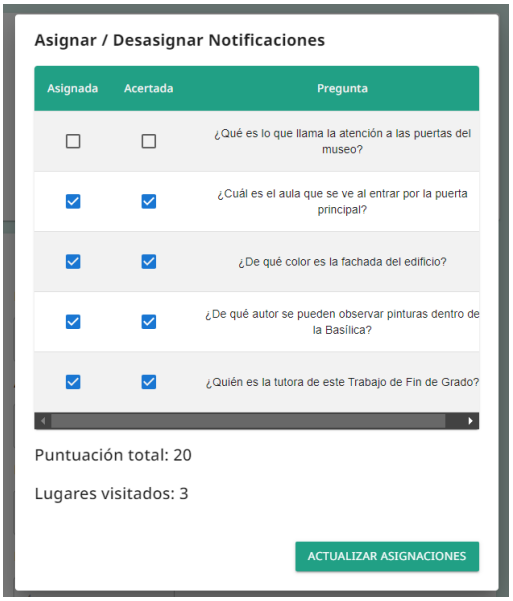


Figura 6.55: Diálogo de asignación de notificaciones

  Se muestra una tabla con las notificaciones y se pueden asignar como visitadas y como acertadas simplemente activando los checkbox que se ven a la izquierda.

  Los cambios en las asignaciones se ven directamente reflejados en los textos 'Puntuación total' y 'Lugares visitados'. Pulsando en 'Actualizar asignaciones', se establecen los nuevos datos en los campos relativos a las visitas del usuario acorde a lo indicado en el diálogo.

Finalmente, es importante destacar que todas las operaciones realizadas no serán persistidas hasta que no se pulse el botón 'Guardar'.

Almacenamiento

Esta pantalla no guarda relación con las colecciones, sino con los directorios disponibles en el servicio de almacenamiento Cloud Storage que se está utilizando en el proyecto. En ella se dispone una pestaña por cada directorio, dando la impresión al usuario de que está navegando por carpetas. Cada pestaña muestra todas las imágenes alojadas en su directorio.

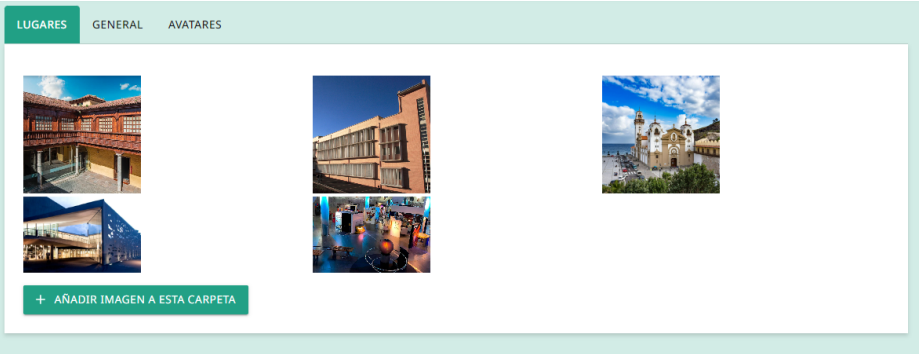


Figura 6.56: Pantalla Almacenamiento

La pestaña de Avatares es de solo consulta, pero Lugares y General cuenta con las siguientes funcionalidades.

- **Modificar una imagen**

  Es posible sobreescribir una imagen con otra. Para ello, se debe pulsar en la imagen que se quiere modificar. Esto desplegará una pequeña ventana:

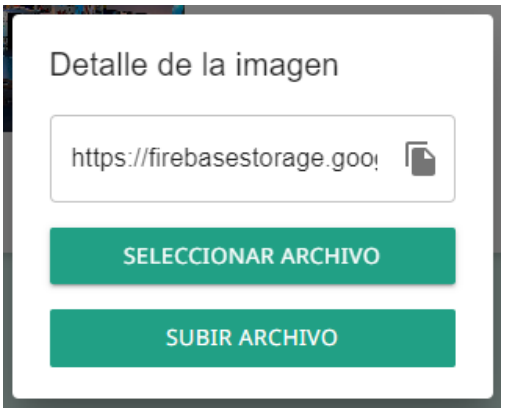


Figura 6.57: Diálogo de modificación de imagen

  En primer lugar se selecciona desde ‘Seleccionar archivo’ la imagen que se quiere subir, después de esto, se habilitará el botón ‘Subir archivo’, que al ser pulsado enviará la imagen al Storage. Finalizado el proceso de subida, se deja a disposición en el campo de texto la URL de la imagen con posibilidad de copiar el valor.

- **Subir nueva imagen**

  El proceso y el diálogo son similares al proceso de modificación.

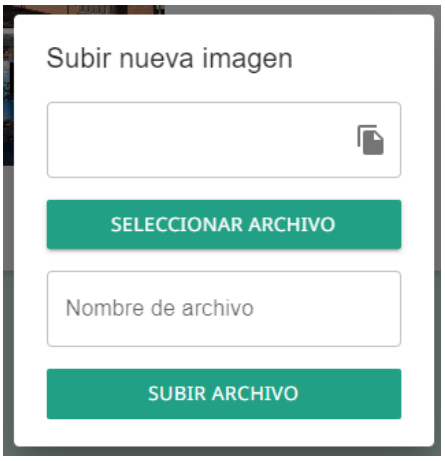


Figura 6.58: Diálogo de nueva imagen

La diferencia radica en que a la hora de subir el archivo, se debe indicar el nombre con el que se quiere guardar la nueva imagen en el fichero.

Capítulo 7 Presupuesto

A continuación se procede a mostrar una tabla donde se refleja la distribución de tareas y las horas invertidas en cada una. El precio de la hora se calcula suponiendo un salario medio de un programador junior en España de 18000 € [17], que son 1500 € mensuales. Teniendo en cuenta una cuota de Seguridad Social de 500 €, que un mes tiene aproximadamente 20 días laborables y la jornada es de 8 horas, resulta una cantidad de 12,5 €/h.

Descripción	Cantidad	Precio (€)	Total (€)
Planificación del proyecto	12	12,5	150
Documentación sobre las tecnologías	30	12,5	375
Desarrollo de la aplicación web	75	12,5	937,5
Desarrollo de la aplicación móvil	225	12,5	2.812,5
Ordenador portátil	1	659,00	659,00
Diseño de la base de datos	20	12,5	250
Redacción de la memoria	40	12,5	500
Google Firebase*	1	0	0
		TOTAL:	5.684

*se ha usado la versión gratuita, habría que consultar los precios de uso si se superan las cuotas mínimas

Tabla 7.1: Presupuesto del proyecto

Capítulo 8 Conclusiones y líneas futuras

Se han cumplido todos los objetivos que inicialmente se habían propuesto, llegando a incluir el sistema de niveles y las campañas o eventos, que a priori se suponían líneas futuras. Como resultado, se tiene una página web de administración robusta, accesible y dinámica, capaz de configurar toda la información necesaria para el funcionamiento de la aplicación móvil, por medio de una interfaz sencilla e intuitiva para un administrador conocedor del proyecto.

Asimismo, la jugabilidad de la app respecto al uso de los Beacons responde tal y como se pretendía, y puntos como el sistema de puntuación, el uso de avatares y nombres de usuario, la colección de los lugares visitados a modo de cartas y que se tenga un mapa como interfaz principal llena de diferentes elementos. En definitiva, se ha logrado añadir numerosas mecánicas de gamificación.

Se mantienen unas pautas de diseño común en ambas aplicaciones, tales como la selección de colores, que en todo momento se ha basado en aquellos que sean análogos o complementarios al color del logotipo del proyecto, siendo este el principal. Al mismo tiempo, se ha hecho hincapié en un diseño con mayor personalidad en la aplicación móvil para causar una mayor atracción al usuario. Todo esto converge en una estética homogénea que aporta profesionalidad al proyecto.

A nivel personal, el desarrollo de este Trabajo de Fin de Grado me ha permitido reforzar en buena medida mis conocimientos sobre las tecnologías del proyecto como son React, React Native y Firebase, tanto a nivel de aplicación web como móvil, llevándome a realizar un continuo análisis en la experiencia de usuario sobre todo lo que se iba diseñando o programando.

Más allá de las tecnologías, la idea y los conceptos sobre los que gira el proyecto me han servido también como inspiración para otros nuevos trabajos de índole similar, donde fomentar el turismo se torna como el principal objetivo de los mismos.

En cuanto a las líneas futuras, el proyecto puede mejorar drásticamente de muchas formas:

Rendimiento

- **Cacheo de los datos**: Incorporar el almacenamiento temporal de los datos de la aplicación en la memoria del dispositivo proporcionará muchos beneficios en el rendimiento de la aplicación, reduciendo en buena medida el consumo de los datos, permitiendo el funcionamiento offline y mejora en la experiencia del usuario respecto a los tiempos de carga de la información.
- **Paginación en el lado de la web**: Si bien la cantidad de datos que existe actualmente en el sistema no es significativamente grande, en un momento dado puede resultar conveniente dividir en bloques los resultados de los listados de las pantallas de la web, lo que permitirá mantener la correcta experiencia de usuario que se obtiene actualmente, así como el ahorro de recursos que supone, entre

otras ventajas.

Alcance

- Añadir **autenticación por medio de otras aplicaciones** como Facebook o Google e incluir la verificación de email, lo que puede motivar a un mayor número de usuarios a hacer uso de la aplicación.

- **Interés de otras organizaciones**: Una buena experiencia dentro de la aplicación puede despertar un interés por las organizaciones dedicadas al mundo de las exposiciones a hacer uso de ella. Un ejemplo sencillo sería incorporar notificaciones al proyecto a demanda de una organización en específico, de modo que el usuario pueda recibir recompensas externas al visitarlas por parte de ella.

Gamificación

- **Nueva oportunidad si no se acierta una pregunta**: Permitir en ciertas notificaciones que se pueda volver a responder una pregunta pasado cierto tiempo en caso de no haberse respondido correctamente aportaría rejugabilidad a la aplicación, incrementando su tiempo de vida.

- **Sistema de amistades**: Incluir un sistema en el que los usuarios tengan un círculo de amistades en el que puedan consultarse los logros de cada uno tendría un impacto positivo en la competitividad que se busca dentro del juego.

- **Mayor personalización del perfil de usuario**: Actualmente se cuenta con foto de perfil y nombre de usuario, pero se puede extender a una mayor personalización donde el usuario tenga a su disposición diferentes estilos de mapa, o reciba insignias, logros o apodos tras lograr alguna hazaña dentro del juego.

Capítulo 9 Summary and Conclusions

All the initially proposed objectives have been achieved, including the implementation of the level system and campaigns or events, which were considered future lines of development. As a result, we have a robust, accessible, and dynamic web administration page capable of configuring all the necessary information for the mobile application's operation through a simple and intuitive interface designed for project-aware administrators.

Furthermore, the gameplay of the app in relation to the use of Beacons aligns exactly with our intentions. Elements such as the scoring system, the use of avatars and usernames, the collection of visited places as cards, and the incorporation of a map as the main interface have been successfully integrated. In short, numerous gamification mechanics have been added.

A common design pattern is maintained in both applications, such as the selection of colors based on those that are analogous or complementary to the project's logo color, which is the primary color. At the same time, emphasis has been placed on a design with more personality in the mobile application to enhance user attraction. All of this converges into a cohesive aesthetic that brings professionalism to the project.

On a personal level, the development of this Final Degree Project has significantly reinforced my knowledge of the project's technologies, such as React, React Native, and Firebase, both for web and mobile applications. It has led me to continuously analyze the user experience in relation to all the design and programming efforts.

Beyond the technologies, the project's idea and concepts have also served as inspiration for other similar works, where promoting tourism becomes the main objective.

Regarding future directions, the project can be greatly improved in several ways:

Performance

- **Data caching**: Incorporating temporary storage of application data in the device's memory will provide many performance benefits, significantly reducing data consumption, enabling offline functionality, and improving the user experience in terms of information loading times.
- **Web-side pagination**: While the current amount of data in the system is not significantly large, there may come a time when it is convenient to divide the results of the listings on the web screens into blocks. This will maintain the correct user experience that is currently obtained, as well as save resources, among other advantages.

Scope

- Adding **authentication** through other applications **such as Facebook or Google** and **including email verification** can motivate a greater number of users to use the application.

- **Interest from other organizations**: A positive user experience within the application can generate interest from organizations dedicated to the exhibition world to make use of it. For example, incorporating custom notifications upon demand from a specific organization, where users can receive rewards from that organization for visiting its exhibitions.

Gamification

- **Retry option for questions**: Allowing users to retry answering a question after a certain period of time in certain notifications would bring replayability to the application.
- **Friend system**: Implementing a system where users can have a circle of friends to consult each other's achievements would have a positive impact on the competitiveness that is sought within the game.
- **Enhanced user profile customization**: Currently, the app includes a profile picture and username, but it can be expanded to offer more customization options such as different map styles or the reception of badges, achievements, or nicknames after accomplishing specific milestones within the game.

Capítulo 10 Anexos

A1. Repositorio del código fuente de la aplicación web

https://github.com/rcalaglez/beasitcon-webapp

A2. Repositorio del código fuente de la aplicación móvil

https://github.com/rcalaglez/beasitcon-app

A3. Reglas de la base de datos

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {

    match /{document=**} {
      allow read: if request.auth != null;
    }

    match /users/{userId} {
      allow read: if request.auth != null;
      allow write: if request.auth.uid == userId;
    }

    match /usernames/{usernameId} {
      allow read: if request.auth != null;
      allow write: if isAdminUser();
      allow delete: if isAdminUser();
    }

    match /beacons/{beaconId} {
      allow read: if request.auth != null;
      allow write: if isAdminUser();
      allow delete: if isAdminUser();
    }

    match /configuration/{configId} {
      allow read: if request.auth != null;
      allow write: if isAdminUser();
      allow delete: if isAdminUser();
    }

    match /difficultLevels/{levelId} {
```

```
      allow read: if request.auth != null;
      allow write: if isAdminUser();
      allow delete: if isAdminUser();
    }

    match /locations/{locationId} {
      allow read: if request.auth != null;
      allow write: if isAdminUser();
      allow delete: if isAdminUser();
    }

    match /events/{eventId} {
      allow read: if request.auth != null;
      allow write: if isAdminUser();
      allow delete: if isAdminUser();
    }

    match /userLevels/{userLevelId} {
      allow read: if request.auth != null;
      allow write: if isAdminUser();
      allow delete: if isAdminUser();
    }

    match /notifications/{notificationId} {
      allow read: if request.auth != null;
      allow write: if isAdminUser();
      allow delete: if isAdminUser();
    }

    function isAdminUser() {
      return request.auth != null && request.auth.token.isAdmin == true;
    }
  }
}
```

A4. Algoritmo de detección de los beacons

```
  export const addBeaconListeners = async (dee, ke, set, bd) => {
    const subscriptions = [];

  if (isAndroid) {
    subscriptions.push(
      dee.addListener('beaconDidAppear', ({beacon: newBeacon, region}) => {
        set(newBeacon);
```

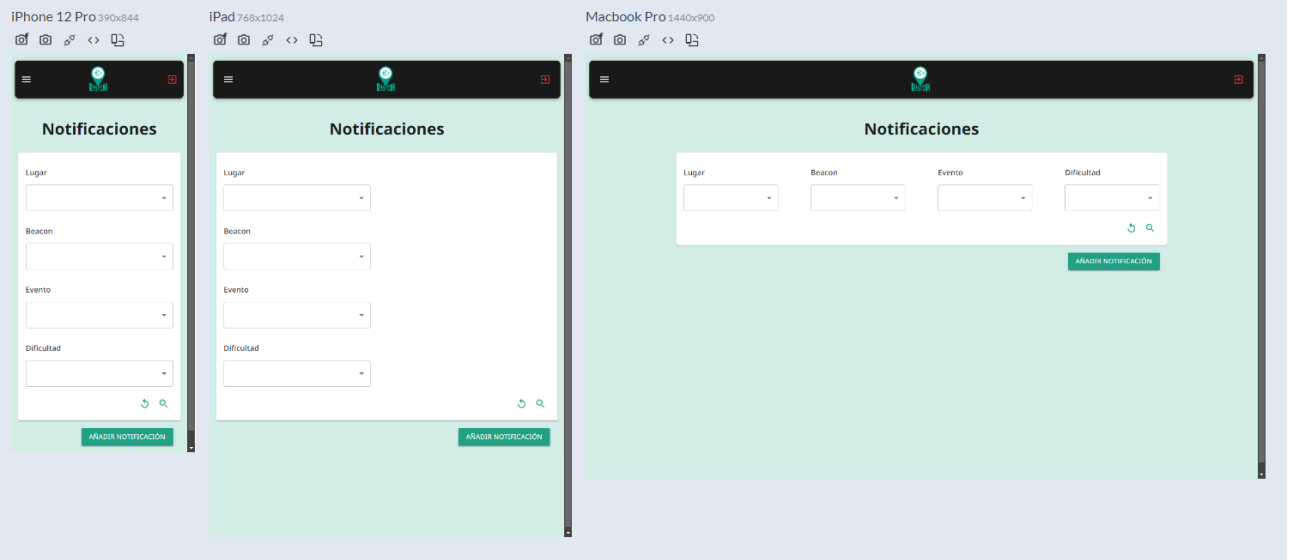
```
      }),
    );

    subscriptions.push(
      dee.addListener('beaconDidDisappear', ({beacon: lostBeacon, region})
=> {
        set({...lostBeacon, isDisappeared: true});
      }),
    );
  } else {
    ke.addListener('didDiscoverDevices', ({beacons}) => {});
  }

  return subscriptions;
};


  useEffect(() => {
    const processFoundBeacon = async beacon => {
      let beaconRegistered = getBeaconRegistered(beacon, beacons);
      if (
        userData &&
        beaconRegistered &&
        !isBeaconVisited(beaconRegistered, userData.visitedNotifications)
      ) {
        const isInCourse = await isNotificationInCourse(beaconRegistered);
        if (isInCourse) {
          setBeaconDetected(beaconRegistered);
        }
      }
    };

    if (beacons && beaconUnprocessed && !beaconUnprocessed.isDisappeared) {
      processFoundBeacon(beaconUnprocessed);
    } else if (
      beacons &&
      beaconUnprocessed &&
      beaconUnprocessed.isDisappeared &&
      beaconDetected &&
      beaconUnprocessed.uuid === beaconDetected.region.uuid &&
      beaconUnprocessed.major === beaconDetected.region.major &&
      beaconUnprocessed.minor === beaconDetected.region.minor
    ) {
```

```
      setBeaconDetected(null);
    }
  }, [beaconUnprocessed, beacons, userData]);
```

A5. Enlace a la página web de administración

https://beasitcon-webapp-seven.vercel.app/

A6. Ejemplo de responsividad



A7. Función de validación de formularios

```
    const validateForm = async () => {
      const validations =
getValidationsFromCollection(props?.collectionName);
      if (validations) {
        let valid = true;
        const newErrors = {};

        for (const key in validations) {
          const value = formData[key];
          const validation = validations[key];
          if (validation?.required?.value && !value) {
            valid = false;
            newErrors[key] = validation?.required?.message;
          }

          const pattern = validation?.pattern;
          if (pattern?.value && !RegExp(pattern.value).test(value)) {
            valid = false;
            newErrors[key] = pattern.message;
```

```
          }

          const custom = validation?.custom;
          if (custom?.isValid) {
            const isValidResult = await custom.isValid(value);
            if (!isValidResult) {
              valid = false;
              newErrors[key] = custom.message;
            }
          }
        }

        if (!valid) {
          setErrors(newErrors);
          return false;
        }
      }
      setErrors({});
      return true;
    };

const BEACON_VALIDATIONS = {
  identifier: {
    required: {
      value: true,
      message: "Este campo no puede estar vacío",
    },
    region: {
      custom: {
        isValid: (value) => {
          return value.major && value.minor && value.uuid;
        },
        message: "Los campos de la region no pueden estar vacíos",
      },
    },
  },
};
```

A8. Reglas de los niveles de usuario

```
  const validateUserLevels = (userLevels) => {
    console.log(userLevels.filter((level) => !level.toDelete));
    let processedUserLevels = userLevels
```

```
      .filter((level) => !level.toDelete)
      .sort((a, b) => Number(a.min) - Number(b.min));
    const numLevels = processedUserLevels.length;

    for (let i = 0; i < numLevels; i++) {
      const currentLevel = processedUserLevels[i];
      const nextLevel = processedUserLevels[i + 1];

      if (i === 0) {
        if (
          currentLevel.min !== 0 ||
          currentLevel.max === undefined ||
          currentLevel.isTopLevel !== false
        ) {
          return {
            isValid: false,
            errorMessage:
              "El nivel más bajo debe partir de 0 y tener un máximo",
          };
        }
      } else if (i === numLevels - 1) {
        if (
          currentLevel.min === undefined ||
          (currentLevel.max !== undefined && currentLevel.max !== null) ||
          currentLevel.isTopLevel !== true
        ) {
          console.log(currentLevel.min === undefined);
          return {
            isValid: false,
            errorMessage:
              "El nivel más alto debe tener un mínimo y el nivel máximo
activado",
          };
        }
      } else {
        if (
          currentLevel.max === undefined ||
          currentLevel.min >= currentLevel.max
        ) {
          return {
            isValid: false,
            errorMessage:
              "El nivel " +
```

```
            currentLevel.level +
            " debe tener un máximo y este debe ser superior al mínimo",
        };
      }

      if (
        nextLevel &&
        Number(currentLevel.max) !== Number(nextLevel.min - 1)
      ) {
        return {
          isValid: false,
          errorMessage:
            "El máximo del nivel " +
            currentLevel.level +
            " actual debe ser igual al mínimo - 1 del siguiente nivel",
        };
      }
    }
  }
```

Bibliografía

[1] Actualidad de los Beacons https://blog.beaconstac.com/2020/09/beacon-technology-updates/ ONLINE Accessed: 2023-14-04.

[2] PokemonGO https://vandal.elespanol.com/guias/guia-pokemon-go/informacion-basica ONLINE Accessed: 2023-11-07.

[3] Touristfy https://www.touristfy.com/ ONLINE Accessed: 2023-11-07.

[4] Frontend https://descubrecomunicacion.com/que-es-backend-y-frontend/ ONLINE Accessed: 2023-08-06.

[5] Backend https://rafarjonilla.com/que-es/backend/ ONLINE Accessed: 2023-08-06.

[6] React Hooks https://es.reactjs.org/docs/hooks-intro.html ONLINE Accessed: 2023-14-04.

[7] Kanban https://www.atlassian.com/es/agile/kanban/boards ONLINE Accessed: 2023-08-06.

[8] Material Design https://material.io/design ONLINE Accessed: 2023-08-06.

[9] Firebase https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0 ONLINE Accessed 2023-14-04.

[10] Beacon https://kontakt.io/ibeacon-and-eddystone/ ONLINE Accessed: 2023-14-04.

[11] NoSQL https://aws.amazon.com/es/nosql/ ONLINE Accessed: 2023-08-06.

[12] Firebase Cloud Firestore https://cloud.google.com/firestore/docs ONLINE Accessed: 2023-08-06

[13] React https://es.reactjs.org ONLINE Accessed: 2023-14-04

[14] React Native https://reactnative.dev ONLINE Accessed:2023-14-04.

[15] Firebase Authentication https://firebase.google.com/docs/auth ONLINE Accessed: 2023-14-04.

[16] Firebase Cloud Storage https://firebase.google.com/docs/storage ONLINE Accessed: 2023-14-04.

[17] Sueldo de un programador junior https://www.hackaboss.com/blog/salario-programador-espana ONLINE Accessed: 2023-25-03