BEACONS BLUETOOTH PARA LUDIFICACIÓN DE UN ESPACIO

Javier Nuevo Sánchez Autor
Dr. Ismael Abad Cardiel Director
Curso 2016-2017 – Convocatoria Junio

Máster Universitario en Investigación en Ingeniería de Software y Sistemas Informáticos
Itinerario Ingeniería de Sistemas Informáticos – Asignatura 105151

Estudios:	Máster Universitario de Investigación en Ingeniería de Software y Sistemas Informáticos
Itinerario:	Ingeniería de Sistemas Informáticos
Código Asignatura:	105151
Título del Trabajo:	Beacons Bluetooth para Ludificación de un Espacio
Tipo de Trabajo:	Tipo B
Nombre del estudiante:	Javier Nuevo Sánchez
Nombre del Director:	Dr. Ismael Abad Cardiel

Espacio reservado para la hoja de calificaciones.

DECLARACIÓN JURADA DE AUTORÍA DEL TRABAJO CIENTÍFICO, PARA LA DEFENSA DEL TRABAJO FIN DE MASTER

Fecha: 29 de mayo de 2017

Quién suscribe:

Autor: Javier Nuevo Sánchez
D.N.I/N.I.E/Pasaporte.: 28971677A

Hace constar que es el autor del trabajo:

Título completo del trabajo.
Beacons Bluetooth para ludificación de un espacio.

En tal sentido, manifiesto la originalidad de la conceptualización del trabajo, interpretación de datos y la elaboración de las conclusiones, dejando establecido que aquellos aportes intelectuales de otros autores, se han referenciado debidamente en el texto de dicho trabajo.

DECLARACIÓN:

- ✓ Garantizo que el trabajo que remito es un documento original y no ha sido publicado, total ni parcialmente por otros autores, en soporte papel ni en formato digital.
- ✓ Certifico que he contribuido directamente al contenido intelectual de este manuscrito, a la génesis y análisis de sus datos, por lo cual estoy en condiciones de hacerme públicamente responsable de él.
- ✓ No he incurrido en fraude científico, plagio o vicios de autoría; en caso contrario, aceptaré las medidas disciplinarias sancionadoras que correspondan.

Fdo.: Javier Nuevo Sánchez

Impreso TFdM05_Autor. Autorización de publicación

y difusión del TFdM para fines académicos

Autorización

Autorizo/amos a la Universidad Nacional de Educación a Distancia a difundir y utilizar, con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firma del/los Autor/es

Juan del Rosal, 16
28040, Madrid

Tel: 91 398 89 10
Fax: 91 398 89 09

www.issi.uned.es

Resumen

En este proyecto, proponemos un despliegue para ludificar un espacio mediante tecnología Bluetooth Smart.

Durante el desarrollo de este documento, realizamos una introducción teórica revisando el estado actual de las técnicas de ludificación y sus diferentes aplicaciones. Igualmente, realizamos una introducción teórica acerca de las tecnologías subyacentes utilizadas, concretamente la tecnología de balizas inalámbricas basadas en Bluetooth Smart.

La descripción teórica precede al diseño de la solución, que incluye una descripción de los componentes creados ex profeso junto con los componentes ya existentes que también forman parte del sistema.

Acercándonos al final del proyecto, procedemos a describir los detalles de implementación más relevantes.

Por último, extraemos conclusiones en base a la investigación, diseño y desarrollo realizados, e ideamos líneas de trabajo futuro.

Palabras clave

Marketing, Psicología, Gamificación, Computación Ubicua, Android, Bluetooth, Web Física, Python, Django, Diseño Adaptable

1 Índice

2 Tabla de ilustraciones

3 Introducción

3.1 Contexto

Posiblemente, una de las ramas más apasionantes de la informática y las telecomunicaciones en los últimos años sea la Computación Ubicua.

Esta disciplina, visionada por Mark Weiser en su artículo seminal de 1993 [1], describe y estudia la tendencia imparable de integrar la tecnología en el mundo físico de forma transparente. Gracias a las tecnologías englobadas dentro de la disciplina de la computación ubicua, los sistemas de información parecen disolverse en la realidad hasta dejar de ser percibidos por nuestros sentidos, siempre persiguiendo su misión de mejorar nuestro mundo.

Esta integración de la tecnología con el mundo físico cristaliza en distintas tendencias, entre las que se encuentran las balizas o *beacons* Bluetooth.

Las balizas Bluetooth utilizan una tecnología inalámbrica de corto alcance y bajos requisitos computacionales y bajo consumo energético, abriendo la posibilidad a que objetos del mundo físico anuncien su presencia a otros dispositivos dando pie a la aparición de la *Web Física* [2].

Durante el desarrollo de este TFM, vamos a estudiar el proceso de *gamificación* de un espacio físico apoyándonos en la tecnología de balizas Bluetooth.

Podemos definir [3] la *gamificación* o *ludificación* como la aplicación de mecánicas, dinámicas y elementos estéticos de juegos en entornos no lúdicos, con el objetivo de generar experiencias que aumenten el grado de interacción y compromiso entre el consumidor o usuario y el tópico *gamificado*.

Cabe destacar que no todas las aplicaciones de las técnicas de *gamificación* requieren un soporte o despliegue tecnológico; no obstante, las tecnologías de la información y la comunicación son las disciplinas que servirán como base al despliegue de *gamificación* que proponemos en este proyecto.

3.2 Objetivos

Durante el desarrollo del siguiente Trabajo Fin de Máster proponemos los siguientes objetivos:

- Realizar un análisis del estado de la cuestión relativo a las áreas del conocimiento subyacentes al proyecto, como son la *gamificación* y la tecnología de balizas Bluetooth.
- Diseñar un despliegue de balizas Bluetooth en un espacio físico para marcar lugares de interés.
- Diseñar e implementar una aplicación web consistente en un juego que requiera la cercanía del usuario respecto a dichas balizas.
- Planificar un caso de aplicación posible en entorno empresarial.
- Plantear experimento con el que averiguar si los usuarios de la actividad *gamificada* valoran más positivamente su experiencia en el establecimiento que los que no hayan participado.

4 Marco teórico y estado del arte

Tal cual hemos descrito en la introducción, en este trabajo fin de master proponemos la ludificación de un espacio físico mediante la utilización de balizas Bluetooth.

Como puede desprenderse por el tema elegido, este proyecto hace necesario trabajar con conocimientos y técnicas pertenecientes a disciplinas dispares.

A lo largo de este apartado, nos acercaremos con más detalle al estado del arte de las técnicas de *gamificación*, desarrolladas y estudiadas principalmente por las disciplinas de la Psicología y el Marketing.

Estas disciplinas nos ayudarán a definir el tipo de experiencia que queremos poner a disposición del usuario mediante su interacción con nuestro sistema.

Una vez que hayamos contextualizado y descrito estas técnicas, pondremos nuestra atención en las tecnologías que utilizaremos para construir la interacción con el usuario.

Las tecnologías que repasaremos, y que utilizaremos a modo de bloques constructivos para nuestro despliegue, serán el protocolo Bluetooth Smart, las principales implementaciones de balizas Bluetooth del sector y los diferentes frameworks de desarrollo de aplicaciones y servicios web.

4.1 Ludificación

Como hemos mencionado en la introducción, una de las disciplinas que sirven de base a este TFM es la *ludificación*, conocida habitualmente como *gamificación*.

Durante los últimos años, hemos podido ser testigos de la incorporación de mecánicas y técnicas de juegos en los contextos más dispares, con el objetivo de incrementar la actividad de los usuarios.

En este apartado, veremos como la *gamificación* encuentra su aplicación en otras disciplinas como la productividad, finanzas, salud, educación, sostenibilidad, gestión empresarial, etc.

El artículo de Deterding et al. citado en nuestra bibliografía [4] repasa el origen del término, así como su alcance y situación actual. El término “gamification”, comúnmente traducido al castellano como “gamificación” o “ludificación”, fue utilizado por primera vez de manera documentada en 2008, consiguiendo una adopción amplia en la segunda mitad de 2010.

Entre las definiciones más aceptadas, encontramos las siguientes:

“La adopción de tecnología y métodos de diseño de juegos fuera de la industria del juego.”

“El proceso de utilizar pensamiento lúdico y mecánicas de juego para resolver problemas e involucrar activamente a los usuarios.”

“La integración de dinámicas de juego en un sitio, servicio, comunidad, contenido o campaña, con el objetivo de fomentar la participación.”

“*Gamificación* es el uso de elementos de diseño de juegos en contextos no de juegos”

Estas ideas no son completamente nuevas, pudiendo encontrar inspiración en las técnicas de diseño de interfaz hombre máquina, que desde principios de los 80 del siglo pasado han buscado la motivación intrínseca de los usuarios introduciendo elementos de videojuegos en software no lúdico.

Cabe destacar que uno de los elementos de la *gamificación* es su acercamiento a la actividad recreativa mediante el uso de reglas y la consecución de objetivos. Así, los elementos recreativos en la *gamificación* estarían más cerca del *ludus* (juego con objetivos y reglas) que del *paidia* (juego libre improvisado, más creativo y expresivo).

Mientras que el concepto de *juego serio* [5] describe el diseño de juegos completos con propósito diferente al entretenimiento, la *gamificación* de una actividad únicamente necesita tomar prestados algunos de los rasgos y mecánicas de los juegos. Entre estos rasgos y elementos, podemos destacar los siguientes, según Reeves y Read [6]:

- La utilización de avatares para representar al jugador.
- Los entornos tridimensionales.
- El contexto narrativo.
- Un sistema de feedback para evaluar el progreso.
- Sistemas de reputación, rankings y niveles.
- Sistemas de comercio y economía.
- La competición siguiendo reglas explícitas.
- La posibilidad de juego en equipo.
- Sistemas de comunicación paralelos fáciles de utilizar.
- La disponibilidad de tiempo limitada para conseguir objetivos.

A la hora de diseñar la *gamificación* de la actividad que nos ocupe, es conveniente tener en cuenta los distintos niveles de abstracción sintetizados por Deterding et al., en base a los que podemos clasificar nuestras decisiones de diseño. Los niveles van de lo más abstracto a lo más concreto:

1. **Patrones de diseño de interfaz:** Son componentes de diseño de interacción y soluciones de diseño utilizadas comúnmente para un problema conocido en un determinado contexto, incluyendo implementaciones prototípicas. A modo de ejemplo, podemos mencionar las insignias, tablas de líderes o niveles.
2. **Patrones y mecánicas de diseño de juegos:** Son las partes recurrentes del diseño relacionadas con la jugabilidad. Entre ellas, encontramos los turnos o la limitación de recursos o tiempo.
3. **Principios y heurísticas de diseño de juegos:** Directrices evaluativas para abordar problemas de diseño o analizar una solución de diseño dada. Ejemplos: juego duradero, objetivos claros, variedad de estilos de juego.
4. **Modelos de juego:** Modelos conceptuales de los componentes o experiencia del juego. Podemos encontrar como ejemplos el *framework* MDA o el estudio de los Core Elements of the Gaming Experience (CEGE) [7].
5. **Métodos de diseño de juegos:** Prácticas y procesos específicos del diseño de juegos, por ejemplo el testeo, el diseño orientado al juego, o el diseño de juegos centrado en el valor.

4.1.1 Introducción desde la perspectiva de la Psicología

En este apartado realizaremos un análisis de los fundamentos de la motivación humana, para después extraer los fenómenos psicológicos aplicables en un proceso de *gamificación*.

La motivación es un constructo psicológico básico utilizado para explicar el comportamiento, definido por Fergusson como "un proceso dinámico interno que energiza y dirige las tendencias de acción de los individuos."

Los motivos humanos tienen dos orígenes fundamentales [8]:

- Biológicos, limitados en rango, pero compartidos por toda la especie. Estos motivos están relacionados con la supervivencia y la reproducción.
- Psicosociales, que exhiben una gran variabilidad entre individuos y entre diferentes culturas.

Podemos resumir la naturaleza y causas de los motivos y emociones humanos [8]:

1. Todo lo que hacemos tiene un origen biológico, pero es modelado por la cultura y la experiencia individual.
2. Los pensamientos suministran la dirección y las metas de la motivación.
3. Los sentimientos suministran la intensidad o fuerza de la motivación.
4. Tanto la motivación como las emociones trabajan conjuntamente para influir en el comportamiento.

Las teorías de la expectativa-valor, dentro de la perspectiva cognoscitiva, expresan la motivación como una función de:

- El valor que las personas asignan a la consecución de un determinado objetivo.
- La probabilidad de éxito a la hora de conseguirlo.

Rotter ideó el concepto de *locus de control* [9], estrechamente relacionado con la motivación. Según el sujeto atribuya los refuerzos y castigos recibidos a sus propias acciones o a factores ajenos a su persona, hablaremos de *locus de control interno* o *externo*, respectivamente. Los sujetos con un alto *locus de control interno* se sienten motivados a perseverar en la búsqueda activa de soluciones, mientras que los sujetos caracterizados por un *locus de control externo* pueden ser más difícil de motivar. Todo esto puede variar con el tiempo, ya que las experiencias de los sujetos pueden hacer cambiar sus atribuciones y, por tanto, su locus de control.

Otra clasificación importante es la que distingue la motivación *intrínseca* de la motivación *extrínseca* [10]. Según esta distinción, la motivación *intrínseca* se aplica a las acciones que se realizan por el disfrute o interés inherente a la propia acción. La motivación *extrínseca* aparece cuando el sujeto realiza una acción porque espera conseguir una recompensa externa asociada a la ejecución de la misma.

También a tener en cuenta al diseñar la *gamificación* de una actividad es el *Efecto de Sobrejustificación,* demostrado de manera experimental por Lepper et. al.[11]. Este efecto muestra cómo el hecho de ofrecer una recompensa o incentivo externo a una acción, puede reducir la motivación intrínseca para su ejecución. De esta manera, una acción que inicialmente se ejecutaba por motivación intrínseca, pasaría a ser ejecutada por motivación extrínseca, por lo que la ejecución de la acción pasaría a depender en gran medida de la existencia del incentivo o recompensa asociado a la misma.

4.1.2 Principios para gamificar aplicaciones

En el artículo *Gamification: State of the Art, Definition and Utilization* de Fabian Groh [12], el autor sintetiza los requisitos imprescindibles para comprender y poder diseñar la *gamificación* de una determinada aplicación o experiencia. Estos requisitos están relacionados con el estudio de la motivación que hemos realizado en el apartado anterior.

Puesto que el objetivo de la *gamificación* de una actividad es hacerla interesante en sí misma, utilizaremos tres principios adaptados de la *Teoría de la Autodeterminación* [13] de Deci y Ryan, que representan tres primitivas capaces de generar motivación intrínseca. Estas tres primitivas son tres necesidades psicológicas innatas, que todos los seres humanos intentamos satisfacer:

- **Relación**: Es la necesidad universal de interactuar y estar relacionados con otras personas. En nuestra aplicación, buscaremos:
    - Conectar con metas personales.
    - Conectar con una comunidad importante para el usuario.
    - Crear una historia significativa.
    - Tener en cuenta el contexto social.
- **Competencia**: Es la necesidad universal de ser efectivo a la hora de resolver un problema en un entorno determinado. Un buen juego debe permitir al sujeto progresar en sus habilidades para afrontar retos cada vez mayores. Idealmente, vamos a ajustar el nivel de dificultad buscando el concepto de *reto óptimo*. Al afrontar tareas de dificultad óptima, logramos que el usuario no se aburra ni se frustre, entrando en un estado de concentración y disfrute definido por Csikszentmihalyi [14] como flujo o *flow*.
- **Autonomía**: Representa la necesidad universal de controlar la propia vida. La mayor parte de los juegos representan una actividad elegida por el sujeto siguiendo sus motivaciones intrínsecas. Si ofrecemos incentivos para completar las metas del juego, el usuario las puede interpretar como una pérdida de autonomía, e incluso pueden hacer que la percepción del valor intrínseco de la actividad sea menor.

4.1.3 Ejemplos de aplicaciones gamificadas

Para redactar el siguiente apartado hemos realizado un recorrido exhaustivo por la literatura científica buscando aplicaciones de *gamificación* a problemas del mundo real. Hemos decidido organizar este apartado en subapartados con los distintos tópicos para los que hemos podido encontrar aplicaciones de estas técnicas.

Educación

La educación es una de las disciplinas que más importancia otorga al estudio de las técnicas de motivación y persuasión para dirigir a los sujetos hacia sus objetivos, debido a su rol crucial en el desarrollo de la sociedad moderna.

Comenzamos revisando cómo la *gamificación* tiene cabida en las aulas de educación primaria, según el artículo de Marín et. al. reseñado en la bibliografía [15]. En el estudio, realizan una encuesta entre futuros docentes para conocer su opinión sobre la idoneidad de incluir videojuegos en el aula de educación primara. A pesar de que la *gamificación* no consiste en la aplicación directa de videojuegos tal y como proponen en el artículo, el resultado positivo del estudio abre las puertas a estudiar nuevas formas de integrar las dinámicas de juego a través de la tecnología en el aula de primaria.

También dentro del ámbito de la educación primaria, el artículo de da Rocha et. al. [16] explica cómo los autores pusieron en marcha un sistema de insignias dentro de un aula de octavo curso de educación elemental. Los autores destacan el mayor compromiso de los alumnos en el aula, pero también reseñan que el papel del docente es muy importante a la hora de poner en marcha estas técnicas con éxito.

El artículo de Sera y Wheeler [17] describe el uso de técnicas de *gamificación* aplicadas a los estudios de Farmacia. Además, los autores mencionan otras aplicaciones de las técnicas de *gamificación* y de *juegos serios* en la enseñanza de las ciencias de la salud: farmacia, enfermería y medicina general. El artículo destaca las siguientes ventajas al utilizar técnicas de *gamificación*:

- Es sencillo adaptar el aprendizaje al nivel del estudiante.
- Permite un mayor grado de libertad al estudiante.
- Abre la puerta al diseño de actividades colaborativas.
- Mejora la motivación, mediante el uso de rankings, insignias, niveles, bonus, etc.

Rouse [18] estudia en su tesis doctoral la relación entre la motivación y las técnicas de *gamificación* en la enseñanza de materias relacionadas con las ciencias. Más concretamente, muestra cómo las técnicas de *gamificación* mejoran los resultados académicos de los alumnos en la asignatura de microbiología en un centro de educación superior.

Figueroa [19] estudia la aplicación de las técnicas de *gamificación* en la enseñanza de una segunda lengua. Uno de los efectos positivos de la *gamificación* para el autor, entre otros que ya hemos comentado, es la posibilidad de utilizar técnicas de recompensa inmediata, que ayudan a lograr mayor motivación en los alumnos. Otra importante implicación es que los objetivos educativos, que no siempre se comunican de manera explícita, pasan a ser objetivos de juego para el estudiante, convirtiéndolos en retos que deben ser superados antes de subir de nivel. El autor profundiza en el modelo de cinco pasos ideado por Yuang y Soman [20] para implementar la *gamificación* un proceso educativo:

1. Comprender la audiencia y el contexto.
2. Definir los objetivos de aprendizaje.
3. Estructurar la experiencia.
4. Identificar los recursos.
5. Aplicar los elementos de *gamificación.*

Para finalizar, el artículo repasa varias plataformas de *gamificación* para el aprendizaje de una segunda lengua en el aula.

Urh et al. [21] diseñan un modelo para introducir técnicas de *gamificación* y e-learning en espacios de educación superior. El modelo hace uso de algunos de los elementos de *gamificación* que hemos revisado hasta ahora, y queda perfectamente sintetizado en la siguiente ilustración extraída del artículo original.

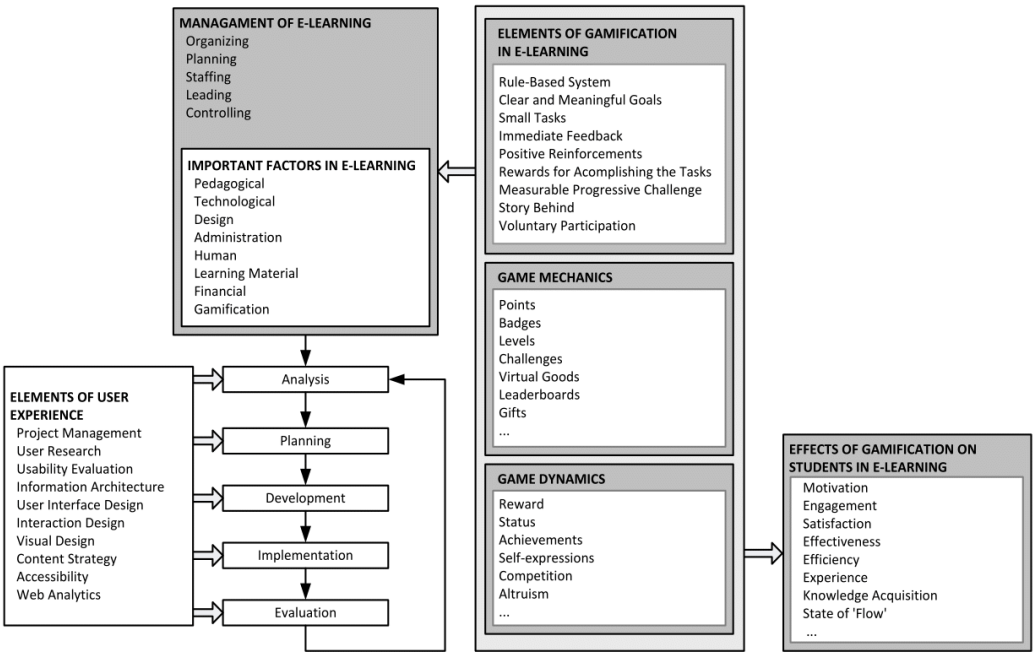

Ilustración 1: Modelo para uso de gamificación en e-learning - Extraído de Urh et al. [2015]

Por último, y de forma más general, el artículo de Quian y Clark [22] sintetiza los 29 estudios más representativos hasta la fecha que tratan acerca de las técnicas de juego aplicadas a la educación, obteniendo como conclusión que la utilización de estas técnicas es positiva para la adquisición de las competencias del siglo XXI.

Salud

Desde la perspectiva de las Ciencias de la Salud, la *gamificación* muestra su utilidad a la hora de motivar a los usuarios para conseguir cambios de comportamiento encaminados a conseguir mejorar su condición física y bienestar.

En nuestra revisión de la literatura científica relacionada, hemos podido encontrar dos artículos recientes (año 2016) a modo de resumen del estado del arte de la situación actual de las técnicas de *gamificación* aplicadas a la salud.

El primer artículo a modo de sumario que queremos mencionar fue redactado por Alahäivälä y Oinas-Kukkonen [23]. Este texto plantea la revisión de 15 artículos de la literatura existente hasta ese momento desde la perspectiva del análisis sistemático de los *contextos de persuasión*, obteniendo las siguientes conclusiones:

- Las herramientas encaminadas a la consecución de hábitos positivos mediante *gamificación* se implementan buscando diferentes cambios en el estilo de vida del sujeto, utilizando diferentes tratamientos y con soporte de diferentes tecnologías y métodos. Aún no existe una visión clara y aceptada de la interrelación entre el contexto, las estrategias de *gamificación* y los resultados.
- Según los autores, para poder avanzar en este campo del conocimiento, sería necesario:
    1. Identificar los agentes de persuasión externos.
    2. Decidir qué cambio se persigue y qué estrategias de *gamificación* utilizar.
    3. Comprender el área de aplicación y las acciones pertinentes a las que aplicar *gamificación*.

4. Prestar atención a los efectos potenciales de las características demográficas de la base de usuarios.
5. Decidir las tecnologías a utilizar basándose en otros factores contextuales.
6. Utilizar rutas de persuasión directas o indirectas, según sea adecuado.
7. Utilizar a modo de guía las teorías de cambio de comportamiento orientado a la salud.
8. Revisar las estrategias de gamificación basándonos en los factores anteriores.
9. Analizar y presentar los resultados con rigor mediante los instrumentos adecuados.

- La investigación futura en este ámbito debería comparar las posibles combinaciones de factores contextuales, teorías relacionadas y estrategias de *gamificación* elegidas, para estudiar sus resultados de forma sistemática y poder entender cómo utilizar la *gamificación* de forma eficiente en los distintos aspectos de las ciencias de la salud.

El segundo artículo de revisión de la literatura también está fechado en 2016, y es obra de Johnson, Deterding et al. [24]. Para los autores de este estudio, el objetivo está en cuantificar la efectividad de las técnicas de *gamificación* para el cambio de comportamiento orientado a la mejora de la salud y el bienestar. Por tanto, el enfoque cuantitativo es la principal aportación de este documento.

Para lograr su objetivo, los autores identifican 7 métricas, que coinciden con las supuestas ventajas de la *gamificación* aplicada a este ámbito según la literatura existente hasta el momento. Las métricas seleccionadas son:

1. Capacidad de aumentar la motivación intrínseca.
2. Facilidad de acceso a través de tecnología móvil y ubicua.
3. Atractivo entre un amplio rango de audiencias.
4. Aplicabilidad para una gran variedad de factores y riesgos de salud y bienestar.
5. Eficiencia a nivel de costes a la hora de mejorar sistemas existentes.
6. Posibilidad de integrarse de manera transparente con la rutina diaria.
7. Mejora directa del bienestar, proporcionando experiencias positivas.

Una vez identificadas estas métricas, los autores seleccionan 19 artículos científicos basados en estudios empíricos, teniendo en cuenta la calidad de la evidencia demostrada, el tipo de efecto y el dominio de aplicación.

Como resultado, el 59 % de los artículos muestran evidencia positiva, y el 41 % restante muestran resultados mixtos, con evidencia moderada o de menor calidad. Los resultados no dejan lugar a dudas en cuanto a los comportamientos relacionados con la salud, pero no son tan claros en el plano cognitivo.

A modo de conclusión, los autores afirman que la *gamificación* puede tener un efecto positivo sobre la consecución de hábitos saludables. No obstante, recuerdan que las conclusiones deben ser tomadas con cautela, debido al reducido número de estudios y las limitaciones metodológicas de algunos de ellos.

Un punto de vista a interesante es el aportado por el artículo de Barratt [25] acerca de la *gamificación* de la práctica ciclista. Su investigación ilustra cómo una aplicación deportiva *gamificada* junto con un monitor de salud pueden ser utilizados con éxito para aumentar la actividad de una comunidad de ciclistas comprometidos. El autor menciona también algunos aspectos negativos, como la reticencia de los usuarios a la vigilancia y evaluación que pueden

suponer estas aplicaciones, o la posibilidad de crear deportistas *hiperconscientes* más movidos por la obsesión que por el disfrute, más propensos al sobreentrenamiento y a asumir riesgos innecesarios, lo que podría tener un impacto negativo en su salud física y mental.

Por último, y desde la perspectiva de la salud mental, revisamos la aplicación Challenger [26], diseñada para ayudar en el tratamiento del trastorno de ansiedad social mediante la utilización de elementos de juego. La aplicación propone retos adaptados a cada usuario, y utiliza elementos de juego (niveles, insignias, etc) para mantenerle motivado y acercarle a sus objetivos. Además de elementos de juego, la aplicación utiliza localización en tiempo real, registro de actividad, notificaciones, psicoeducación, e incluso incluye la funcionalidad de red social anónima. Por tanto, no resulta sorprendente que uno de los principales retos según los autores sea precisamente la seguridad de los datos personales, habida cuenta de la delicadeza de los asuntos relativos a la salud mental en nuestra sociedad. También es conveniente destacar que los autores consideran necesario investigar de manera más amplia la eficacia de la aplicación comparada con otros métodos como la terapia cognitivo-conductual utilizando ordenadores o la psicoterapia tradicional.

A la vista de estos artículos, podemos ver que aún existen retos importantes para aplicar las técnicas de *gamificación* con pleno éxito en las disciplinas relacionadas con las Ciencias de la Salud.

Organizaciones y Marketing

A pesar de que las primeras aplicaciones en aparecer fueron las educativas y las relacionadas con la salud, las técnicas de gamificación también están siendo aplicadas en el ámbito de las organizaciones, tanto para motivar a los trabajadores a conseguir los objetivos empresariales como en el campo del marketing, para guiar a los clientes hacia los objetivos de la empresa.

El primer artículo que revisaremos [27] trata sobre la *gamificación* en procesos de negocio. En el sector servicios, los primeros antecedentes de *gamificación* pueden encontrarse en los programas de puntos de las aerolíneas. Otro ejemplo, esta vez de cara al funcionamiento interno de las organizaciones, es la *gestión por objetivos*, que comparte con las técnicas de *gamificación* la existencia de metas que los sujetos deben cumplir. Según el artículo, una aplicación posible es utilizar la *gamificación* para detectar los empleados que peores resultados consigan. Más allá de las implicaciones éticas de utilizar la *gamificación* con este propósito, los autores defienden que en el sector servicios es habitual este tipo de prácticas para la optimización de recursos y reducción de costes. Menos habitual es encontrar aplicaciones de *gamificación* aplicadas a la gestión de recursos humanos en entornos de producción industrial, a pesar de que existen nuevas técnicas (captura de movimiento, integración de sensores) que están siendo estudiadas. Finalmente, el artículo concluye que los mayores retos para la aplicación de estas técnicas en un futuro son legales y éticos, más que tecnológicos.

El artículo de Cardador et al. [28] aborda el mismo tema, apuntando cómo la *gamificación* mejora la visibilidad, comparabilidad e inmediatez de la información de rendimiento en el entorno laboral. Según los autores, la motivación de los trabajadores aumenta, especialmente en el medio plazo, con la posibilidad de conocer su rendimiento comparado con el estándar. La mayor motivación también va acompañada de un mayor disfrute de las tareas encomendadas. El artículo también destaca la variabilidad de estos resultados según las diferencias individuales entre los trabajadores. Por último, el texto concluye advirtiendo de la necesidad de tener en cuenta varias consideraciones antes de implementar técnicas de gamificación en el entorno

laboral, como son la necesidad de liderazgo, las implicaciones éticas o la necesidad de asociar recompensas reales a la consecución de objetivos de juego.

Hofacker et al. [29] estudian las técnicas de *gamificación* en el marketing móvil. En el artículo, desarrollan el papel en la *gamificación* de los cuatro elementos de diseño que, según Schell [30] crean un ecosistema cognitivo y afectivo alrededor de un juego:

1. La historia o formato narrativo, que provee el contexto a un juego y añade significado a la experiencia.
2. La mecánica de juego, que se refiere a las reglas y aspectos estructurales, junto con los premios, incentivos y niveles del juego.
3. La estética, que refuerza el desarrollo del argumento creando una experiencia inmersiva.
4. La tecnología, que está relacionada en la manera en la que el medio modela la experiencia de juego.

Además, Hofacker et al. describen la posible influencia que diferentes tipos de productos y de consumidores pueden tener en la experiencia de gamificación:

- **Relacionados con el tipo de producto:** Los productos funcionales se benefician de una narrativa basada en las acciones del consumidor, mientras los productos hedónicos hacen más efectiva una narrativa sobre las reacciones a la experiencia con el producto. La mecánica debe depender de cómo se desee reforzar la identidad de los productos. La estética debería ser más importante para productos hedónicos, aunque también puede serlo en los utilitarios. Por último, algunas tecnologías, como la realidad virtual, puede ser más importante para productos altamente experienciales.
- **Relacionados con el consumidor:** Las metas del consumidor deben modificar la experiencia de *gamificación*. Así, el hecho de aumentar las metas de entretenimiento y disfrute, puede interferir con las metas de utilidad e instrumental. Igualmente, una historia rica puede contribuir a una meta de aprendizaje. También puede ser necesario tener en cuenta otros compromisos, por ejemplo, el uso de mecánicas que fomenten la interacción social pueden ser preferibles a las de aprendizaje, siempre que esto coincida con las metas de los usuarios.

Un ejemplo de aplicación de *gamificación* desde el punto de vista de los clientes puede verse en el artículo de Rodrigues et al. [31], que analiza su impacto en los usuarios de un servicio de banca electrónica. Estos autores también destacan la falta de evidencia empírica como uno de los problemas principales a la hora de plantearse utilizar técnicas de *gamificación*. Por esto, en su artículo proponen y evalúan un modelo basado en el Modelo de Aceptación de la Tecnología, que comprueban llevando a cabo un estudio entre 183 clientes de un servicio bancario. En su estudio, concluyen que la *gamificación* del servicio aumenta de forma significativa el sentido de interacción social del cliente, aumentando su intención de utilizar la aplicación.

Por último, nos centramos en el estudio de la gamificación en el ámbito del turismo. Revisando la literatura científica existente, vemos que existen representaciones virtuales 3D de atracciones turísticas en forma de juego. Una de ellas es la plataforma DynaMus [32], centrada especialmente en museos. La otra es una representación virtual del Ágora de Atenas [33]. En todo caso, estos proyectos están más cerca de la idea de *juego serio* que de la idea de *gamificación*, por lo que sólo los mencionamos. En ambos casos, no integran dinámicas de juego en el mundo físico, sino que crean un videojuego con su propia representación virtual de las atracciones.

Más relevante es la propuesta de Confalonieri [34] et al., que consiste en diseñar una experiencia interactiva dentro de un museo, según la cual los usuarios pueden formar un grupo y calificar conjuntamente sus imágenes favoritas. De esta forma, los autores se centran en la dimensión social del juego para incrementar el valor de la visita al museo. En la misma línea, Frasca et al. [35] gamifican la *Gallerie dell'Accademia* mediante el uso de rutas, juegos de memoria, tipo puzzle, búsqueda del tesoro y juego de las diferencias.

4.2 Introducción a Bluetooth Smart

Antes de repasar la tecnología de balizas Bluetooth y sus aplicaciones, realizamos una breve introducción a la tecnología subyacente, llamada *Bluetooth Low Energy* (BLE) y rebautizada recientemente como *Bluetooth Smart*.

Bluetooth es una tecnología inalámbrica para redes de área personal (WPAN), trabajando en la banda de frecuencia ISM de 2.4 GHz

La especificación *core* de Bluetooth [36] es revisada por el Bluetooth Special Interest Group y supervisada por los Bluetooth SIG Working Groups. La tecnología Bluetooth es un estándar con una amplia adopción por parte de la industria.

Las dos implementaciones principales de la especificación [37] son Bluetooth Basic Rate/Enhanced Data Rate (BR/EDR) y Bluetooth Low Energy (LE). Mientras que Bluetooth BR/EDR se utiliza en las versiones 2.0 y superiores, Bluetooth Low Energy fue introducido con la versión 4.0. Cada una de estas versiones tiene aplicaciones diferentes, por lo que su implementación requiere chips distintos, o chips capaces de trabajar en modo dual, que es lo más habitual en la actualidad.

La especificación original de Bluetooth es la implementada por las tecnologías BR/EDR, pensada para crear una conexión inalámbrica continua, lo que la convierte en una tecnología ideal para casos de uso como el streaming de audio.

La especificación Bluetooh Low Energy permite la conexión inalámbrica basada en ráfagas breves, por lo que es perfecto para aplicaciones que no requieran conexión continua y en las que maximizar la duración de la batería sea prioritario, como en sensores o dispositivos móviles. La tecnología subyacente a BLE fue introducida originalmente por Nokia con la denominación de Wibree, que a partir de 2007 [38] comenzó el proceso de integración en el *core* de Bluetooth.

Para poder utilizar tecnología Bluetooth, un dispositivo debe implementar alguno de sus *perfiles*. Además de implementar los perfiles clásicos, la introducción de BLE en las versiones Bluetooth 4.0 y siguientes conlleva la existencia de una pila de protocolo nueva para poder establecer enlaces simples de forma rápida y eficiente.

Así, en Bluetooth Smart existen dos perfiles obligatorios: Generic Access Profile (GAP) y Generic Attribute Profile (GATT).

El perfil GAP controla las conexiones y el descubrimiento de dispositivos. Es el perfil que describe la topología y define si dos dispositivos pueden interactuar entre sí. Este perfil, además, se encarga de definir los diferentes roles que pueden tener los dispositivos, especialmente si son dispositivos *centrales* o dispositivos *periféricos*.

- Los dispositivos periféricos tienen mayores limitaciones de potencia y consumo energético, y se conectan a un dispositivo central mucho más potente. Habitualmente son sensores, etiquetas de proximidad, etc.

- Los dispositivos centrales poseen habitualmente mayor potencia y memoria, y suelen ser smartphones, tablets y portátiles.

Durante el proceso de identificación, el periférico define un intervalo específico que dicta la periodicidad con la que emite un paquete que contiene información de descubrimiento (*advertising data*). La periodicidad de emisión de estos paquetes tiene relación con el consumo de electricidad.

Si un dispositivo a la escucha necesita más información acerca del periférico, puede solicitarle datos adicionales (*scan response data*).

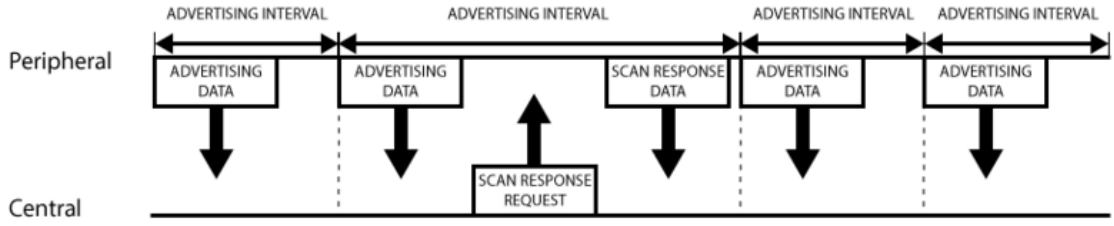

Ilustración 2: Proceso de descubrimiento en Bluetooth Smart. Fuente: adafruit.com

A pesar de que la mayoría de periféricos se anuncian con el objetivo de establecer una conexión entre dos dispositivos, existen casos de uso en los que es necesario enviar datos a varios dispositivos a la vez, lo que sería imposible de lograr en estado conectado. Para poder lograr este objetivo, se incluye información personalizada como *advertising data* o como *scan response data*. Esta técnica es la que se conoce como *difusión* o *broadcasting*, y es en la que se basa el funcionamiento de los *beacons* o balizas en *Bluetooth Smart*.

El perfil GATT de *Bluetooth Smart* define la manera en que dos dispositivos Bluetooth Smart intercambian datos entre sí, por medio de sus *servicios* y *características*. GATT entra en funcionamiento una vez que se ha establecido una conexión dedicada entre dos dispositivos, lo que implica que ya se ha realizado todo el proceso de conexión y descubrimiento mediante GAP. Recordamos que las conexiones de GATT son exclusivas, por lo que un periférico BLE no puede estar conectado a varios dispositivos centrales simultáneamente.

Por tanto, en modo conectado, un dispositivo central BLE puede estar conectado a varios dispositivos periféricos simultáneamente; sin embargo, cada dispositivo periférico sólo puede estar conectado a un dispositivo central. Cabe destacar que, una vez que la conexión ha sido establecida entre un periférico y un dispositivo central, la comunicación fluye en ambas direcciones, a diferencia de la implementación unidireccional que consistía sólo en enviar *advertising data* mediante GAP.

En las *transacciones* GATT se utiliza un modelo cliente/servidor, en el que el cliente, que es el dispositivo maestro (el teléfono o Tablet) realiza peticiones al servidor, que tomaría el rol de dispositivo esclavo (el periférico).

Al establecer una conexión, el periférico sugiere al dispositivo maestro un intervalo de conexión por el que regirse para comprobar si hay datos disponibles.

Las transacciones GATT están organizadas en objetos anidados de alto nivel llamados *Perfiles*, *Servicios* y *Características*.

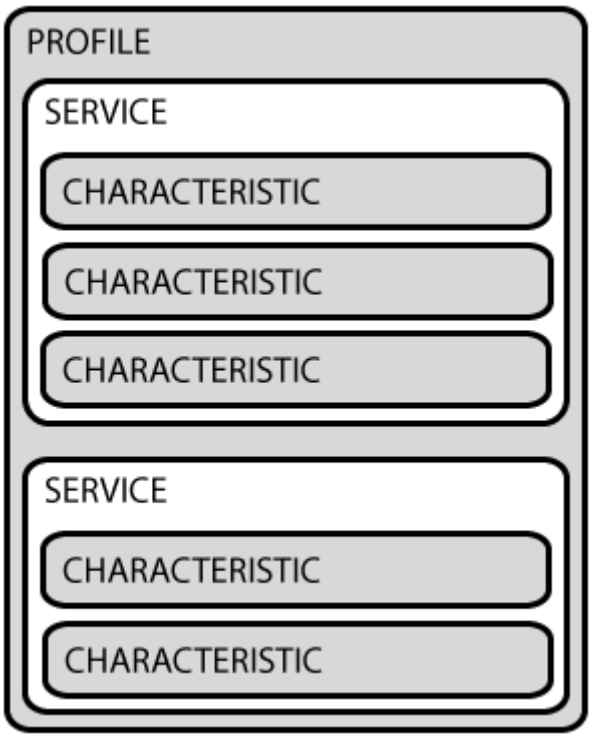

Ilustración 3: Capas de abstracción en GATT. Fuente: adafruit.com

Los Perfiles en los periféricos BLE representan colecciones de Servicios que van juntos para un determinado propósito.

Los Servicios se utilizan para dividir los datos en entidades lógicas, y contienen bloques específicos de información llamados Características.

Bluetooth SIG va añadiendo diferentes servicios a la especificación. Por ejemplo, existe un servicio llamado *Heart Rate Service*, que contiene tres Características: *Heart Rate Measurement*, *Body Sensor Location* y *Heart Rate Control Point*.

Las Características, por tanto, son el concepto de menor nivel dentro de la jerarquía, y también están estandarizadas por el Bluetooth SIG.

Tanto los Servicios como las Características se referencian mediante un identificador UUID. Cabe destacar que también es posible crear un identificador personalizado para utilizar con nuevos Servicios y Características no estandarizados hasta ahora por el Bluetooth SIG.

4.3 Tecnologías de balizas Bluetooth

Las balizas o *beacons* son transmisores unidireccionales que se utilizan para marcar lugares u objetos importantes [39], abriendo la puerta al desarrollo de aplicaciones y webs sensibles al contexto físico.

Las balizas Bluetooth utilizan el perfil GATT de Bluetooth Smart para su funcionamiento, de manera que no necesitan realizar el proceso de conexión ni sincronización entre dispositivos.

El soporte de Bluetooth 4.0 en los dispositivos móviles dio pie a la implementación de diferentes tecnologías de balizas Bluetooth, como vemos en la siguiente línea temporal extraída de [40]

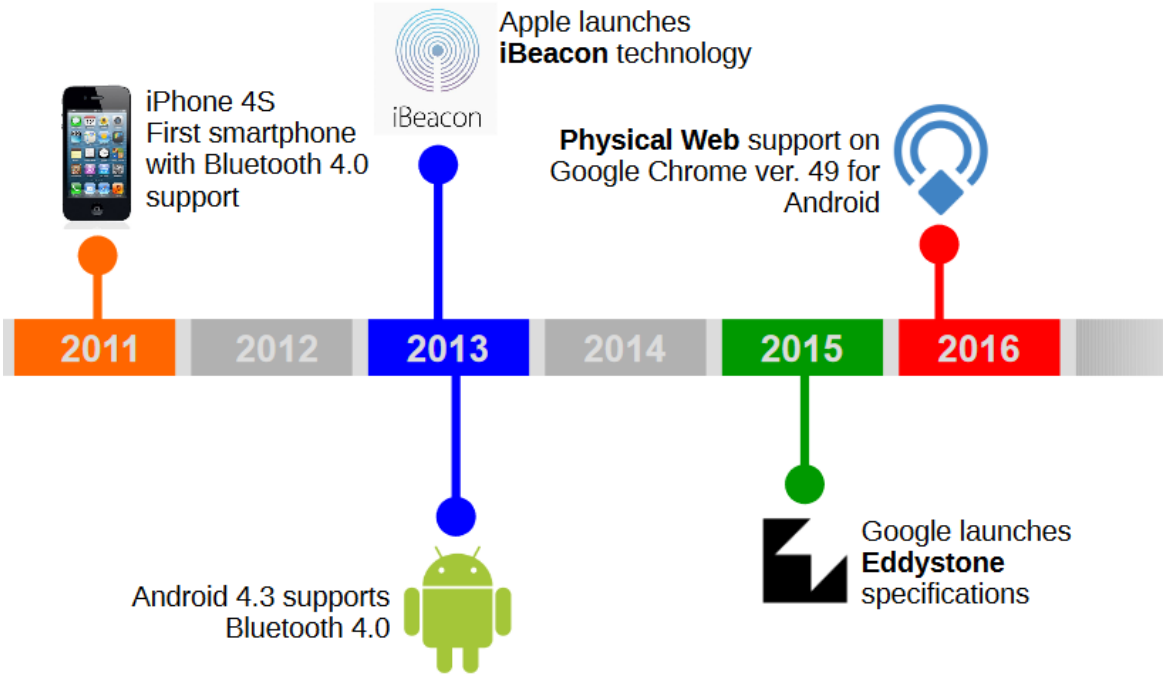

Ilustración 4: Línea temporal Beacons Bluetooth. Fuente: presentación DroidconIT

En la actualidad podemos nombrar dos implementaciones principales, en ambos casos, respaldadas por los recursos de grandes empresas: iBeacon, de Apple y Eddystone, de Google.

4.3.1 iBeacon

La implementación de balizas de Apple es la primera que vio la luz de entre las dos que vamos a estudiar, puesto que fue publicada en mayo de 2013.

La especificación requiere un intervalo de anuncio de 100 milisegundos y un paquete de difusión sin conexión.

El paquete de difusión de iBeacon incluye un identificador numérico único para la baliza, codificado mediante tres números (UUID de proximidad, número mayor y número menor). También incluye un registro de RSSI, utilizado para calcular proximidad y precisión.

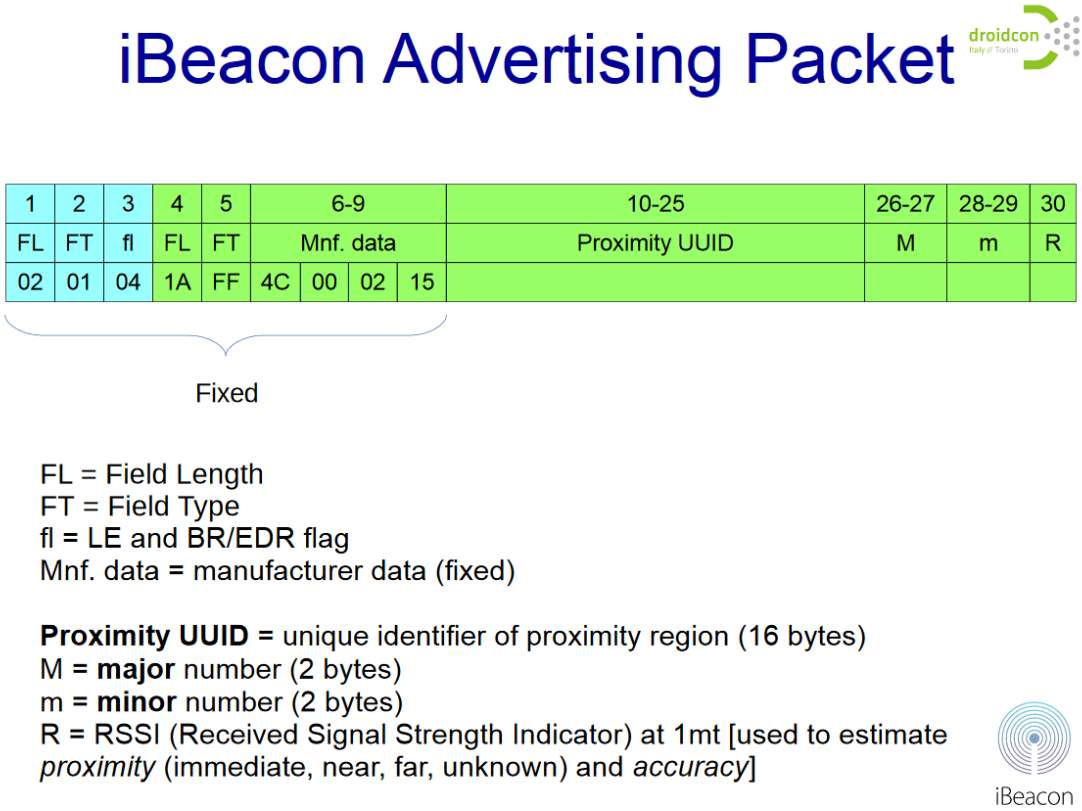

Ilustración 5: Formato de paquete iBeacon. Fuente: presentación DroidconIT

4.3.2 Eddystone

Eddystone es la implementación opensource de balizas Bluetooth creada por Google, cuya especificación fue lanzada en julio de 2015.

Eddystone define varios tipos diferentes de paquetes de difusión:

- **Eddystone-UID**: Es similar a iBeacon, al disponer de un identificador único opaco, pero su estructura de identificador es diferente.
- **Eddystone-EID:** Este tipo de balizas dispone de un identificador único encriptado para aumentar la seguridad del protocolo. Más allá de esto, funciona igual que la baliza UID.
- **Eddystone-URL:** Este tipo de baliza incluye una URL que apunta a un servidor que utiliza el protocolo SSL.
- **Eddystone-TLM:** Envía información acerca de la propia baliza para funciones de telemetría, como información de batería, datos de sensores, etc.

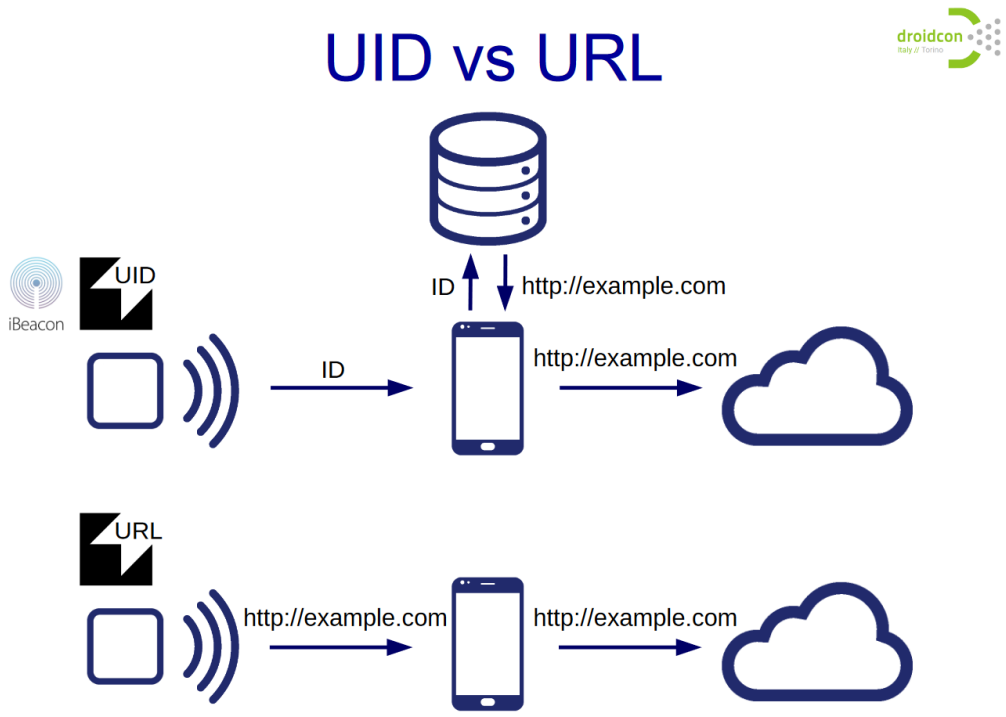

Ilustración 6: Comparación beacons UID y URL. Fuente: presentación DroidconIT

Los tipos de balizas UID y EID tienen un funcionamiento similar entre sí: en ambos casos, es necesario interpretar el código del identificador, para lo que Google lanzó los APIs *Proximity Beacon* y *Nearby*. De este modo, el dispositivo debe disponer de una aplicación específica que, haciendo uso de estas herramientas, es capaz de leer e interpretar las balizas.

Sin embargo, las balizas de tipo URL ya incluyen su carga de información dentro de la propia retransmisión, por lo que no necesitan ser descodificadas mediante una aplicación específica.

Las balizas de tipo URL son el fundamento de la *Web Física*, puesto que generan notificaciones en el navegador Google Chrome tanto en Android 4.4+ como en iOS. El navegador *Physical Web* también es capaz de mantenerse en segundo plano y mostrar las URL anunciadas una vez que el dispositivo iOS o Android entre en el área de alcance de un beacon Bluetooth.

Cabe destacar que Google Chrome no tiene por qué mostrar todas las balizas en el radio de alcance del dispositivo, puesto que antes las contrasta con un servicio de filtrado propio, con el objetivo de evitar el SPAM y otros contenidos peligrosos según sus propios criterios.

4.4 Estado del arte y aplicaciones de balizas Bluetooth

A fecha de elaboración de este documento, la tecnología de balizas Bluetooth sigue siendo relativamente reciente, tomando la presentación de la tecnología iBeacon de Apple en 2013 como referencia por su relevancia en la industria.

La respuesta por parte de Google con el formato Eddystone se presentó hace poco más de un año, lo que sigue dando una idea de la incipiencia de estos estándares de balizas Bluetooth.

Sin embargo, hemos podido encontrar un número suficiente de referencias en la literatura científica como para tomar consciencia de que se está realizando trabajo en este ámbito para conocer mejor la tecnología e idear nuevas aplicaciones que la integren en nuestra vida cotidiana.

4.4.1 Localización

Las limitaciones de la tecnología GPS en espacios cerrados hace que se estén estudiando diversas alternativas para suplir estas carencias, entre las que encontramos la utilización de radiofrecuencia en interiores.

Antes de la aparición de la tecnología Bluetooth Smart, una técnica común para calcular la localización de un dispositivo móvil en un espacio cerrado era el análisis de escena Wi-Fi, calculando la posición estimada de un dispositivo mediante la triangulación de su distancia a puntos de acceso inalámbricos utilizando dicha tecnología [41].

En 2012, Zhu et Al. [42] idearon la manera de mejorar sistemas de localización existentes basados en tecnología inalámbrica Wi-Fi mediante la utilización de balizas Bluetooth. Una de las fases de un proceso de localización mediante Wi-Fi consiste en la creación de un mapa del espacio físico, que puede ser realizado por expertos o mediante datos de los propios usuarios. El artículo de Zhu demuestra como el uso de las balizas Bluetooth ayuda especialmente en las fases iniciales de creación de este mapa con la colaboración de los usuarios, permitiendo mantener una buena precisión en la localización.

El estudio de Chiu et Al. presentado en 2016 [43] ahonda en las técnicas híbridas de localización, esta vez combinando la localización Wi-Fi con balizas Bluetooth Smart del protocolo iBeacon. Su sistema utiliza técnicas de *fingerprinting*, mediante las cuales se asocian de antemano distintas lecturas a una determinada posición (fase *offline*). Posteriormente (fase *online*), para calcular la posición de un dispositivo, se comparan sus lecturas con las de estas *huellas* existentes en el sistema. Este cálculo se puede realizar con varios algoritmos, siendo elegido en este caso el algoritmo *k*-nn (*k*-Nearest Neighbor), por ser el más preciso y razonablemente eficiente. La prueba experimental de su estudio verifica cómo el sistema híbrido de posicionamiento en interiores consigue mayor precisión (0.81) que la localización únicamente mediante balizas iBeacon (0.72) o mediante Wi-Fi (0.54).

Otra aplicación de la tecnología de balizas Bluetooth para la localización es la de contribuir a los resultados de un sistema de navegación por estima (*Pedestrian Dead Reckoning*). En este tipo de sistema se utilizan los sensores de un Smartphone (acelerómetro, giróscopo, brújula, etc.) para estimar la trayectoria de un peatón en interiores. Chen et Al. [44] proponen compensar el error de deriva que se produce en los sistemas de tipo PDR mediante el uso de balizas Bluetooth. Comparado con las técnicas de Wi-Fi *fingerprinting*, su propuesta es más precisa, rápida y eficiente en el uso de energía.

De interés resulta también el estudio de Varela et al. [45], en el que los autores prueban experimentalmente que es posible detectar de manera robusta un grupo de personas caminando juntas mediante la tecnología iBeacon. Para esto, cada peatón debe llevar encima una baliza Bluetooth y un Smartphone equipado con una aplicación al efecto.

Queremos concluir este apartado con el estudio de Bouchard et Al. [46] sobre el comportamiento de los beacons Bluetooth y lo predecible de su comportamiento. Los autores realizan una evaluación empírica sobre las estimaciones de distancia mediante métodos de lectura de intensidad de señal RSSI, concluyendo que ofrecen serias dificultades debido a las características de propagación multicamino de Bluetooth. Las conclusiones de este artículo vienen a confirmar la importancia de los métodos híbridos mencionados anteriormente.

4.4.2 Otras aplicaciones

La capacidad de localización en interiores de personas y objetos que hemos reseñado en el punto anterior hace que la tecnología de balizas Bluetooth sea uno de los habilitadores del IoT. En este tipo de despliegues, las balizas Bluetooth se utilizan para digitalizar el contexto en el que se relacionan personas, dispositivos y espacios.

La aplicación BeaSmart [47] forma parte de una iniciativa para dotar de inteligencia las instalaciones del ARC de IBM en California. La aplicación se conecta con las balizas Bluetooth instaladas en el centro de investigación para guiar a sus usuarios por las instalaciones del centro, indicar dónde encontrar determinados artículos del almacén, conocer la temperatura percibida por los usuarios en determinadas zonas o comunicar notificaciones importantes a los usuarios que se encuentren en el radio de alcance de una determinada baliza.

Encontramos otro ejemplo de aplicación de balizas Bluetooth en Zhao et al. [48] En este caso, los emisores de balizas son adjuntados a grandes piezas recién fabricadas, dotándolas de identidad digital. De esta manera, es posible conocer en todo momento su ubicación a lo largo del recorrido por un almacén industrial. Mediante triangulación, los miniordenadores distribuidos por todo el almacén son capaces de interpolar la posición de las piezas. De esta forma, los autores han medido una considerable reducción del tiempo necesario para localizar las piezas, pasando de 17.2 minutos a 3.1 minutos.

Otra de las propuestas [49] realiza el control de asistencia automático de los estudiantes en un campus universitario. Las balizas Bluetooth son utilizadas para localizar a los alumnos, además de servir para enviar a los alumnos información como tareas o recursos de interés. Para despliegues similares a este, en los que la alta densidad de puntos BLE pueda generar complicaciones, Wang y Brassil [50] describen la posibilidad de utilizar un intervalo de anuncio variable en la emisión de balizas Bluetooth, teniendo en cuenta el estado de los dispositivos y de la propia red. En el mismo artículo concluyen que, con las tecnologías actuales, la mínima separación entre balizas Bluetooth debería ser de 0.5 metros por canal para evitar problemas.

Siguiendo con las aplicaciones de balizas Bluetooth, el artículo de Wakao et al. [51] describe la posibilidad de realizar un juego de *búsqueda del tesoro* en un centro comercial. El artículo confirma que el método propuesto motiva a los participantes a caminar dentro del centro comercial buscando el *tesoro*, fomentando la participación grupal en el proceso de compra de las familias. Este es un buen ejemplo mezclando las técnicas de gamificación con la tecnología de balizas Bluetooth.

También con el objetivo de mejorar la experiencia de los usuarios de un servicio, He et al. [52] proponen el desarrollo de un sistema de información utilizando balizas iBeacon en un museo. Los visitantes del museo tendrían que instalar en sus dispositivos móviles una aplicación capaz de interpretar la señal de las balizas instaladas en determinadas salas o colecciones. Al detectar dicha señal, el dispositivo móvil podría mostrar información relevante para los visitantes del museo gracias a su información sobre el contexto.

Cerramos este apartado mencionando una de las aplicaciones de las balizas Bluetooth que pueden tener un carácter más disruptivo por su cercanía al usuario y por la simplicidad de su implementación: la Web Física o *Physical Web* [2].

La Web Física busca una manera fácil de interactuar de forma digital con lugares y objetos físicos. Respecto a otras implementaciones de balizas Bluetooth, cuenta con la ventaja de no necesitar instalar una aplicación específica para interactuar con cada objeto, puesto que la interacción se

realiza a través de tecnologías web. La web física es un estándar abierto, basado a su vez en varias tecnologías abiertas, como la implementación de balizas Bluetooth mediante el protocolo *Eddystone* impulsado por Google.

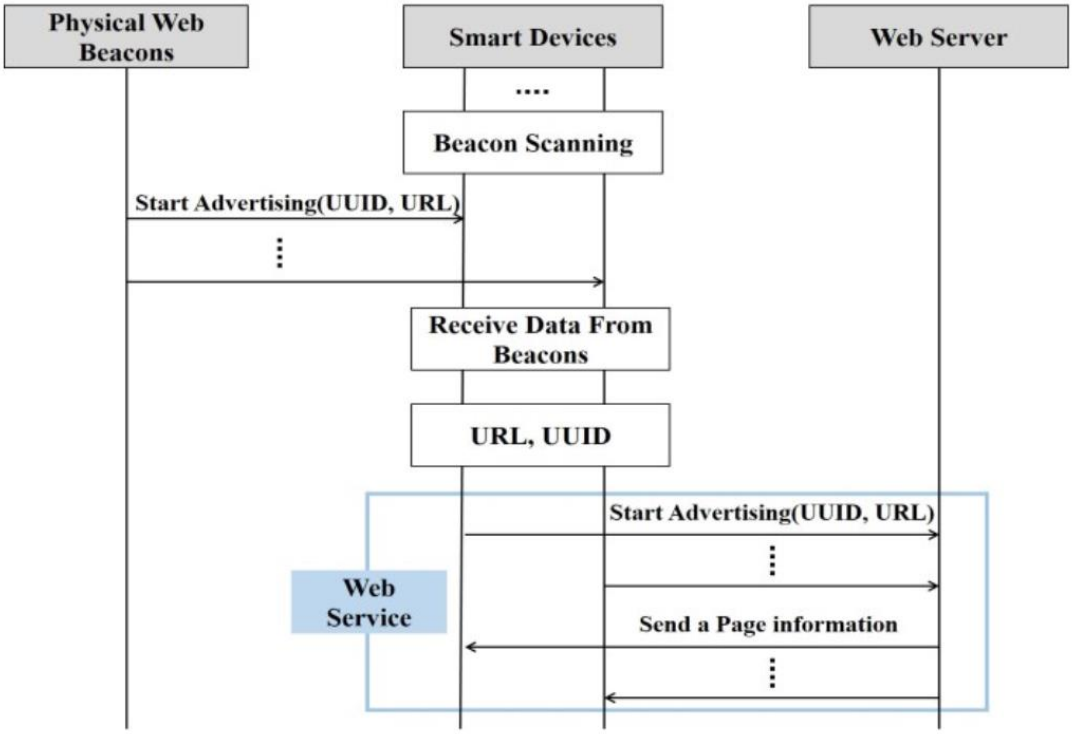

Ilustración 7: Web Física - diagrama de secuencia. Fuente: Lee et al.

Como vemos en la ilustración anterior, obtenida de Lee et al. [53], la baliza que recibe el dispositivo ya incluye directamente una URL para ser accedida sin necesidad de que una aplicación decodifique el UUID de la baliza. El artículo de Lee et al. también presenta la plataforma ALLFIT, que utiliza la tecnología de Beacons Bluetooth para hacer que todos los elementos principales de equipamiento de un gimnasio emitan una URL con información sobre su uso.

El desarrollo de nuestro proyecto utilizará las tecnologías de la Web Física para implementar una experiencia *gamificada* en un espacio físico.

5 Diseño de la solución

5.1 Objetivos

En la introducción al TFM ya planteamos los objetivos generales del presente estudio. Sin embargo, en este apartado concretamos los objetivos específicos de nuestra solución software:

- Planificar un despliegue de balizas inalámbricas en un espacio físico: Utilizaremos tecnología Bluetooth Smart, con balizas del tipo Eddystone-URL, para marcar posiciones de interés en espacios cerrados.
- Estudiar las diferentes alternativas para realizar el despliegue de balizas.
- Definir las mecánicas de juego dentro de las limitaciones técnicas y de tiempo de desarrollo, y teniendo en cuenta las reglas estudiadas durante la introducción teórica.
- Diseñar una aplicación web consistente en un juego basado en la ubicación del usuario. La ubicación vendrá marcada por la cercanía del usuario a las balizas Bluetooth desplegadas en el punto anterior. Utilizaremos los principios de *gamificación* estudiados durante la introducción teórica para aplicar estos elementos a la visita a un establecimiento.

5.2 Visión general

La solución propuesta consta de varios elementos con similar importancia interrelacionados entre sí.

A continuación, describimos los elementos que forman parte de la solución, junto con la relación entre ellos:

- Mecánicas de juego: Las mecánicas de juego son el conjunto de reglas y objetivos propuestos para *gamificar* la actividad.
- Características de la sede: El tamaño, distribución de espacios y actividad principal son los atributos fundamentales de la sede.
- Despliegue de balizas: El despliegue de balizas se caracteriza por el número, posición, tipo y URLs de las balizas distribuidas por el espacio.
- Aplicación web: la aplicación web depende de todos los demás elementos, e implementa el sistema de información con el que interactúa el usuario final.

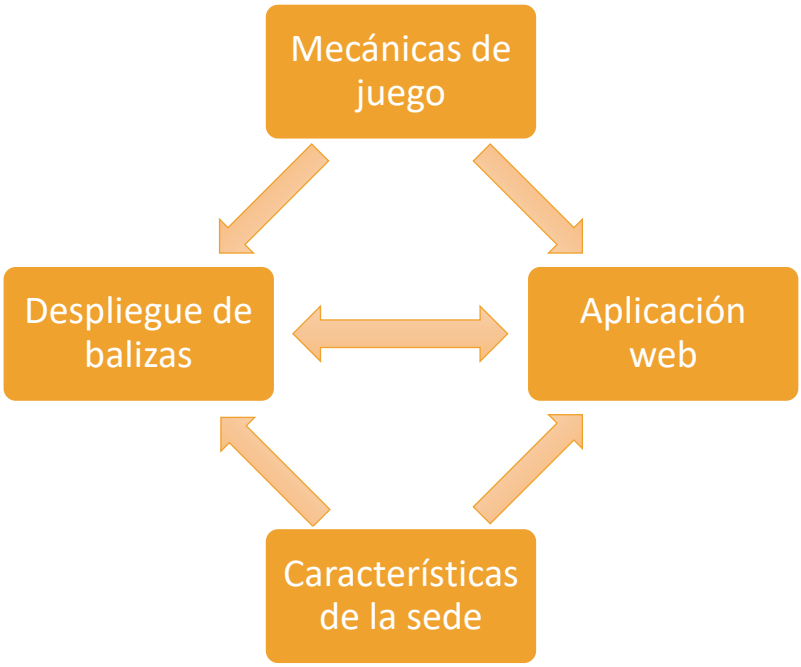

Ilustración 8 - Esquema general de los elementos de la solución.

En el esquema anterior vemos cómo las mecánicas de juego y la morfología de la sede definirán las características del despliegue de balizas. Así, las características de la sede definirán el número, alcance y autonomía requerida de las balizas, y las características de la sede, en conjunción con las mecánicas de juego, definirán la localización física y el tipo de las balizas.

Desde la perspectiva de la aplicación web, parte de su funcionalidad vendrá dictada por las dinámicas de juego a implementar. Las características de la sede modelarán el contenido de la aplicación web, puesto que las balizas contienen información sobre el punto del edificio en el que se encuentran. Por último, la aplicación web también está estrechamente relacionada con las balizas, puesto que el contenido que emitan las balizas será una URL existente en la aplicación web.

5.3 Requisitos funcionales

El conjunto de nuestra solución software deberá ofrecer las siguientes funcionalidades a los distintos usuarios del sistema:

Desde el punto de vista del visitante del espacio *gamificado*:

- Registrarse e iniciar sesión para participar en el juego.
- Ver su progreso durante la actividad.
- Comparar su puntuación con la de los demás jugadores.
- Conocer más información sobre el espacio visitado.

Desde el punto de vista del administrador del sistema:

- Gestionar los usuarios registrados.
- Gestionar el contenido de la página asociada a cada baliza.

Pasamos a desarrollar con más detalle cada uno de estos requisitos:

Nº Requisito: 1	**Clasificación**: desde el punto de vista del jugador
Descripción: Registro e inicio de sesión del usuario en el sistema para participar en el juego.	
Razón: Es imprescindible poder hacer seguimiento de las interacciones de cada usuario para llevar el cómputo de sus respuestas y poder calcular sus puntuaciones. Para esto, es necesario que el usuario pueda registrarse e iniciar sesión en el sistema.	
Dependencias: Gestión de usuarios de Django.	
Posibles problemas: Funcionamiento de cookies de sesión con Physical Web.	

Nº Requisito: 2	**Clasificación**: desde el punto de vista del jugador
Descripción: Indicador de puntuación acumulada durante el desarrollo de la actividad.	
Razón: Como hemos visto en la introducción teórica, una de las maneras de aumentar la motivación intrínseca de un usuario para participar en una actividad es la evaluación inmediata de su rendimiento.	
Dependencias: Registro e inicio de sesión de usuarios.	
Posibles problemas: no descritos	

Nº Requisito: 3	**Clasificación**: desde el punto de vista del jugador
Descripción: Comparar la puntuación con la de otros jugadores.	
Razón: Otro elemento que contribuye a aumentar la motivación son las tablas de máximas puntuaciones, que permiten comparar un resultado con el de los demás participantes.	
Dependencias: Registro e inicio de sesión de usuarios.	
Posibles problemas: Que la diferencia de rendimiento entre unos jugadores y otros no fomente la competitividad, por no suponer una actividad de "reto óptimo".	

Nº Requisito: 4	**Clasificación**: desde el punto de vista del jugador
Descripción: Conocer más información sobre el espacio visitado	
Razón: Siguiendo motivaciones culturales y de branding, pretendemos que el usuario aprenda datos sobre el espacio gamificado durante la actividad.	
Dependencias: Gestor de contenidos web. Navegador en el dispositivo del usuario.	
Posibles problemas: La cantidad de información debe ser la mínima suficiente para ser relevante sin perder la atención del jugador.	

Nº Requisito: 5	**Clasificación**: desde el punto del administrador
Descripción: Gestionar los usuarios registrados, pudiendo realizar operaciones del tipo CRUD (crear, leer, actualizar, borrar).	
Razón: Es necesario para asegurar que la información contenida en el sistema sea correcta, y para poder utilizarla técnicamente con fines diferentes al desarrollo de la actividad.	
Dependencias: Utilizaremos la funcionalidad de *backend* por defecto de Django.	
Posibles problemas: no descritos	

Nº Requisito: 6	**Clasificación**: desde el punto del administrador
Descripción: Gestionar el contenido de cada baliza asociada a la actividad: Fotografía, descripción y preguntas.	
Razón: Permite una relativa flexibilidad a la hora de realizar modificaciones en el juego soportado por nuestro despliegue.	
Dependencias: Utilizamos la funcionalidad *backend* de Django y sus operaciones CRUD.	
Posibles problemas: no descritos	

5.4 Requisitos no funcionales

Entre los requisitos no funcionales que deberá cumplir nuestro despliegue, estos serían los más importantes según la categorización de Merkow [54]:

- **Disponibilidad**: Nuestro despliegue no requiere de una disponibilidad crítica, por lo que será suficiente con poder reparar cualquier posible incidencia dentro del mismo día de trabajo.
- **Interoperabilidad**: Uno de nuestros objetivos es que el mayor número de usuarios pueda disfrutar de la experiencia que proponemos, por lo que utilizaremos la tecnología Bluetooth Smart para las balizas, así como tecnologías web basadas en estándares del lado de la aplicación. Nuestro objetivo es que se pueda participar en la actividad con móviles con sistema operativo Android e iOS.
- **Posibilidades de gestión**: La aplicación web desde la que gestionaremos la *gamificación* tendrá acceso desde cualquier lugar con conexión a internet.
- **Mantenibilidad**: para facilitar el mantenimiento futuro del software, reutilizaremos librerías de código y *frameworks* siempre que sea posible, elaborando además la suficiente documentación interna en el código.
- **Seguridad**: Nuestra aplicación web requerirá un sistema de login que confiaremos al framework de desarrollo web. Además, dicho framework posee funciones de mitigación para amenazas frecuentes.

5.5 Mecánica de juego

Uno de los primeros elementos a diseñar es la propia experiencia de *gamificación* en nuestro espacio físico.

En un despliegue de balizas Bluetooth como el que planteamos, las balizas se utilizan fundamentalmente para detectar la cercanía del usuario a distintos elementos o zonas de un determinado espacio cerrado.

Siguiendo los principios reflejados en el apartado 4.1. de introducción teórica acerca de la gamificación, incorporamos los siguientes elementos típicos de juegos:

- El participante se registrará para participar en el juego, de manera que el sistema pueda hacer seguimiento de su progreso.
- El participante recibe un *feedback* instantáneo sobre su actuación cada vez que se enfrenta a una prueba.
- Se establece un ranking con los jugadores que hayan conseguido la máxima puntuación.

Sobre la mecánica del juego, la definimos como sigue:

- El usuario se registra en cualquier punto asociado a una baliza. En la baliza se le explicará que debe buscar los puntos marcados con balizas por el espacio *gamificado* para ir superando pruebas.
- Cada vez que el usuario encuentre una baliza, se le planteará una pregunta que debe responder con éxito. El sistema marcará la baliza como visitada y sumará un punto al usuario en el caso de que la pregunta haya sido correctamente respondida en el primer intento.
- Tras responder la pregunta de cada baliza, se mostrará una pista para encontrar otra que aún no haya sido visitada por él.

- En todo momento, el usuario conocerá su puntuación acumulada, el número de balizas que le queda por visitar y el ranking de puntuaciones de los mejores 5 jugadores de esa sede.
- Para calcular la puntuación del usuario, definimos las siguientes variables:

  b es el número total de balizas de juego distribuidas por el establecimiento.

  i_k es el número de puntos conseguidos en la baliza de juego *k*. Recordamos que acertar la pregunta en el primer intento sumará 1 punto. No acertar a la primera, sumará 0 puntos.

  Por tanto, la puntuación total ***P*** de cada jugador se calcula como:

$$P = \sum_{k=1}^{b} (i_k)$$

  Una vez que una determinada baliza sea visitada y su cuestionario sea contestado por el usuario, en las siguientes visitas no podrá volver a responder la pregunta.

5.6 Despliegue de balizas Bluetooth

5.6.1 Introducción

El despliegue de balizas *Bluetooth Smart* sirve en este proyecto para localizar a los participantes en nuestra actividad. Como ya hemos mencionado, al depender la actividad *gamificada* de una interfaz web para su desarrollo, estamos ante un caso de aplicación de la *Web Física*.

La solución de balizas Bluetooth debe cumplir las siguientes características:

- **Económica**: Para *gamificar* un espacio, será necesario utilizar varias balizas Bluetooth. Por tanto, el precio unitario de cada dispositivo puede tener un impacto significativo en el despliegue de proyectos grandes.
- **Robusta**: Al estar las balizas dispersas en el espacio *gamificado*, su funcionamiento debe requerir la menor intervención posible por parte del equipo de soporte.
- **Bajo consumo de energía**: Un bajo consumo de energía haría posible el funcionamiento mediante baterías, abaratando el coste de despliegue.
- **Interoperable**: Utilizaremos balizas del tipo Eddystone-URL, por ser compatibles con dispositivos iOS y Android sin necesidad de instalar ninguna aplicación específica para la propia actividad.

5.6.2 Soluciones comerciales

Hemos revisado diferentes soluciones comerciales de balizas Bluetooth, con el objetivo de encontrar la más adecuada para nuestro despliegue.

En nuestra búsqueda, hemos descartado las balizas de los siguientes tipos:

- Balizas sin soporte para el protocolo Eddystone-URL.
- Balizas que requieren alimentación por USB o transformador eléctrico.
- Balizas que priorizan la delgadez, pequeñas dimensiones o formato *wearable* sobre la vida útil de la batería.

Hecho esto, hemos podido componer una tabla con las soluciones comerciales que hemos encontrado cumpliendo nuestros requisitos:

Imagen	Marca	Modelo	Alcance (metros)	Duración batería (meses)	Precio feb. 2017
	AnkhMaway	AKMW-iB003N	100	58	32,08 €
	Axaet	PC037-ULP	30	25	13,50 €
	Axaet	PC063-LP	60	48	23,63 €
	Sensoro	SmartBeacon-4AA	80	60	31,03 €
	Iotton	TON9218	100	120 (batería recargable con panel solar)	35,66 €
	Sky	Forecum 201	100	24	23,87 €
	Axaet	PC061E	30	18	19,18 €

	April Brother	Energy Efficient King	35	60	24,15 €
	Axaet	PC023E	30	12	19,18 €
	AprilBrother	202	30	14	22,02 €
	Estimote	Proximity Beacons	70	24	18,70 €
	Accent Systems	iBKS Plus	70	104	25 €

Ilustración 9 - Comparativa de balizas Bluetooth comerciales

Todos los precios han sido obtenidos en febrero de 2017 desde la tienda online www.beaconzone.co.uk, y convertidos de libras a euros al tipo de conversión del momento.

Para ayudarnos a elegir una solución de entre todas las candidatas, debemos tener en cuenta dos aspectos:

- Al utilizar los beacons para detectar la proximidad del usuario, conseguiremos **más precisión cuanto menor sea el alcance** de la baliza. Por tanto, hemos resaltado positivamente en la tabla aquellas balizas con un alcance de hasta 35 metros.
- De igual forma, una larga **duración de batería** hace más rentable el despliegue, debido al menor coste de mantenimiento en concepto de mano de obra. Hemos resaltado de forma positiva en la tabla todos los emisores con una duración de batería superior a 50 meses.

A la vista de la tabla comparativa, y por ser una solución destacada en los dos criterios técnicos anteriores, seleccionaríamos las balizas *Energy Efficient King* de la marca *April Brother* para nuestro despliegue. Como podemos ver, el coste unitario de cada baliza (menor de 25 €) también está cercano al promedio de las soluciones comparadas.

Cada solución de las que hemos comparado utiliza el software de gestión específico de su fabricante.

Así, en el caso de las balizas *Energy Efficient King*, es necesario utilizar su propia aplicación *H3C* para gestionar los parámetros de las balizas [55].

Al ejecutar la aplicación *H3C* en un dispositivo móvil que esté en el rango de la baliza *Bluetooth* que queremos configurar, podemos acceder a los parámetros de la baliza desde el dispositivo. Entre estos parámetros podemos seleccionar el tipo de baliza deseada, en nuestro caso, Eddystone-URL.

Cabe destacar que este proceso deberá repetirse con cada baliza de nuestro despliegue, asignando a cada baliza la URL que le corresponda a su ubicación en nuestra aplicación *gamificada*.

En este tipo de equipamiento simple priorizando la duración de la batería, es habitual que la gestión se realice mediante Bluetooth, al no disponer de ningún otro medio de conectividad (Wifi, ethernet, etc.)

Ilustración 10 - Configuración baliza. Fuente: https://blog.aprbrother.com

5.6.3 Implementación piloto de baliza para Web Física

Con el objetivo de poder realizar pruebas sin tener que adquirir balizas Bluetooth Smart, vamos a poner en marcha un prototipo de implementación de baliza *Bluetooth* sobre hardware de propósito general.

A pesar de que podríamos haber desplegado el servidor de balizas piloto en nuestro ordenador personal de desarrollo, hemos decidido que es más positivo poner en marcha dicho servidor en una placa de desarrollo independiente.

De esta forma, tenemos un dispositivo autónomo con el que podemos hacer pruebas de campo en distintos espacios para estudiar el comportamiento de la tecnología de balizas Bluetooth.

En cuanto a la elección de la placa de desarrollo, hemos valorado las siguientes posibilidades:

Tipo de placa	Ventajas	Inconvenientes
Placas basadas en Arduino	• Menor consumo eléctrico • Funcionamiento en tiempo real	• Las placas con BLE son caras • Menos opciones de conectividad • Desarrollo y depuración de software más costosos en el tiempo • Menos librerías predefinidas
Placas Linux ARM	• Poseen un sistema operativo completo. • Más potencia de procesamiento • Compatibles con cualquier disp. USB soportado por Linux • Funcionan con Linux: gran variedad de lenguajes y librerías para programar	• Consumo eléctrico moderado • Tiempo de procesamiento no determinista • Mayor complejidad intrínseca, al disponer de un S.O. completo

Ilustración 11 - Familias de placas de desarrollo. Elaboración propia.

Para este caso, hemos preferido utilizar una placa ARM con sistema operativo completo para el desarrollo del despliegue piloto. Sin embargo, en producción, un sistema basado en microcontroladores, como los sistemas Arduino, pueden conseguir una mayor simplicidad y estabilidad, junto con un consumo eléctrico notablemente menor.

Una vez que hemos tomado la determinación de utilizar una placa ARM con Linux, proponemos una tabla comparativa de las soluciones disponibles en el mercado:

	Raspberry Pi 3 **[56]**	Beaglebone Black **[57]**	Odroid C2 **[58]**
CPU	4×ARM Cortex-A53, 1.2GHz	ARM Cortex-A8, 1GHz	4xARM Cortex-A53, 1.5GHz
RAM	1GB LPDDR2	512MB DDR3	2GB DDR3
Ethernet	10/100	10/100	10/100/1000
Wifi	802.11n 2.4 GHz	n.d.	n.d.
Bluetooth	Bluetooth 4.1 Classic y BLE	n.d.	n.d.
Precio	35 €	$ 54.95	$ 46.00

Ilustración 12 - Principales placas de desarrollo ARM con S.O. Linux. Precios marzo 2017.

La tabla anterior no pretende ser exhaustiva, sino recoger las placas de desarrollo más ampliamente extendidas en la actualidad

En nuestra tabla comparativa de soluciones ARM con Linux, podemos ver cómo la Raspberry Pi 3 es la única de las propuestas que incluye Bluetooth SMART integrado sin necesidad de adquirir ningún periférico separado, por lo que será la opción elegida para nuestra baliza piloto.

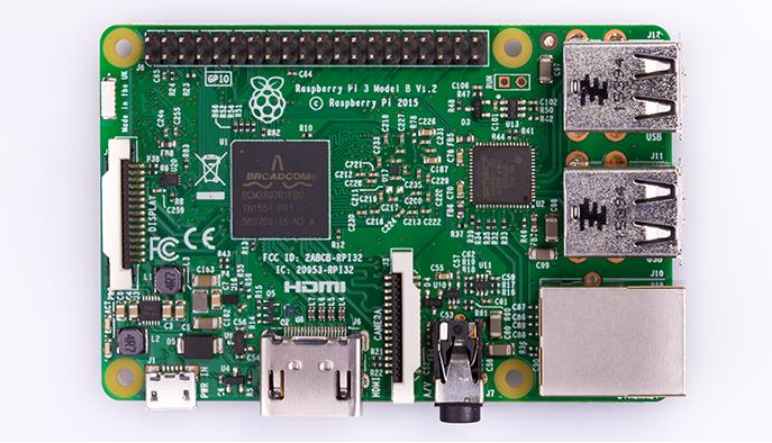

Ilustración 13 - Placa de desarrollo Raspberry PI 3. Fuente: raspberrypi.org

Utilizaremos la implementación *node-eddystone-beacon* [59] para emitir nuestra señal *Bluetooth*. Esta librería se apoya a su vez en la librería *Bleno* [60], que aporta funcionalidad BLE al entorno de ejecución *Node.js* [61]

Para instalar nuestro servidor de balizas y todas sus dependencias, seguiremos este orden:

1. Instalar la distribución Raspbian en la placa. Raspbian es un derivado de Debian compilado para Raspberry Pi.
2. Instalar las librerías de desarrollo y soporte bluetooth y la implementación *Bluez* para Linux.
3. Instalar Node.js y su gestor de paquetes *npm*. Preferimos el repositorio *NodeSource* al incluido en la distribución *Raspbian*.
4. Instalar *Bleno* y, por último, la implementación *eddystone-beacon* utilizando el gestor de paquetes *npm*.

A la hora de realizar la prueba, tenemos que recordar varios detalles importantes de la tecnología de *Web Física* en su estado actual bajo Android:

- Las balizas Eddystone-URL deberán apuntar a una dirección web segura (prefijo https).
- La dirección referida por las balizas no podrá tener más de 40 caracteres.
- Según nuestras pruebas, el sistema es capaz de interpretar correctamente las direcciones de internet abreviadas mediante el servicio *bit.ly*.
- Para interpretar las balizas de la web física necesitamos un dispositivo equipado con Android 4.3.2+ o Google Chrome para iOS 8+ [62]

Para probar el prototipo, configuramos nuestro servidor de prueba basándonos en un ejemplo incluido de serie en el paquete *eddystone-beacon:*

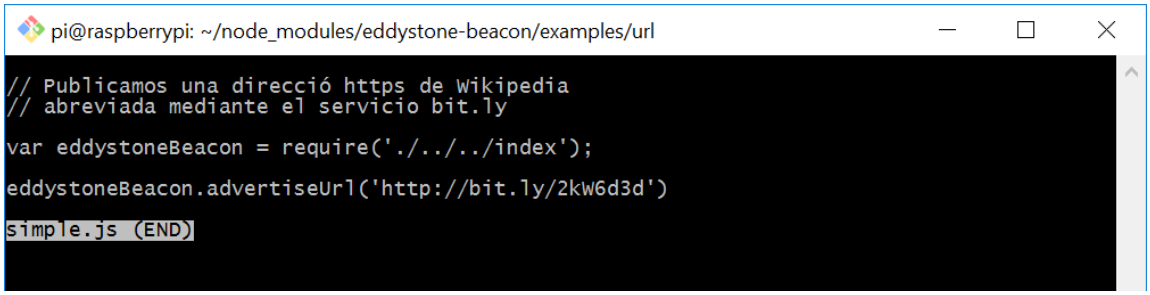

Ilustración 14 - Fichero de configuración del servidor de balizas

Hemos incluido la dirección acortada de un artículo de Wikipedia. Hemos utilizado el servicio bit.ly para acortar la URL, puesto que tenía una longitud mayor de 40 caracteres.

Ponemos en funcionamiento el servidor de balizas con el siguiente comando:

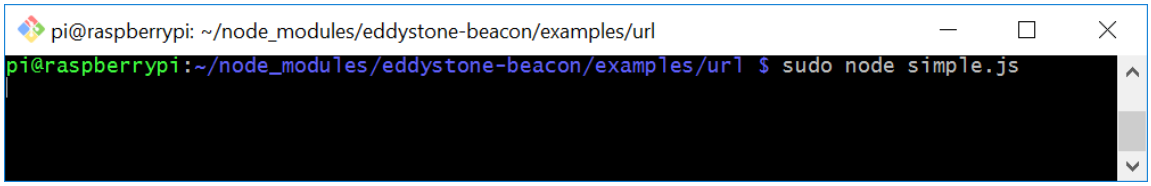

Ilustración 15 - Comando para iniciar el servidor de balizas

Para probar si es posible leer la baliza, utilizamos un smartphone que reúne las siguientes características técnicas y condiciones:

- Tiene instalado Android 7.1 *Nougat*, con las últimas versiones de los Google Play Services (10.2) y de Google Chrome (56) en el momento de hacer las pruebas. En cuanto a hardware, dispone de Bluetooth 4.1
- Tenemos activado *Bluetooth*, el servicio de ubicación del sistema operativo Android y las notificaciones de *Physical Web*.

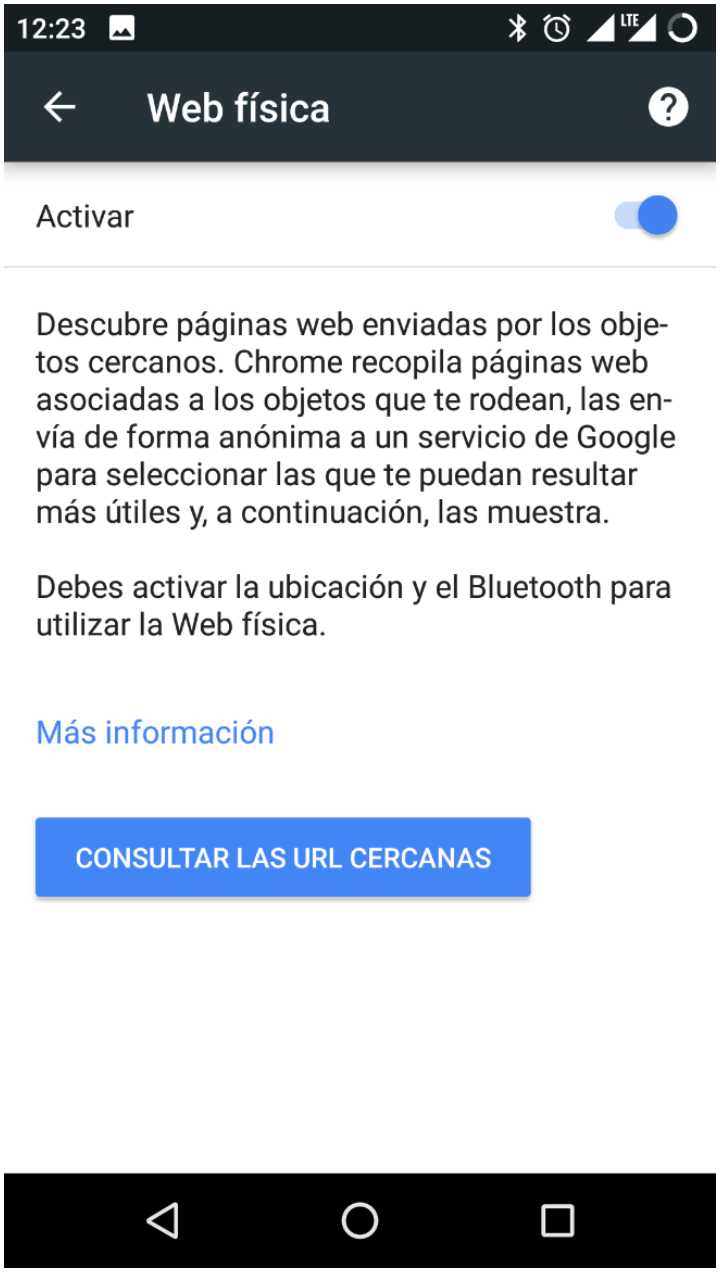

Ilustración 16 - Pantalla de activación de notificaciones de Web Física en Android

Como podemos ver en las siguientes imágenes, en el momento de desbloquear la pantalla del dispositivo, deslizamos la barra de notificaciones y aparece el contenido de la URL que habíamos abreviado: un artículo de Wikipedia sobre H.R. Hertz. También vemos el mismo artículo al entrar en la aplicación *Nearby*:

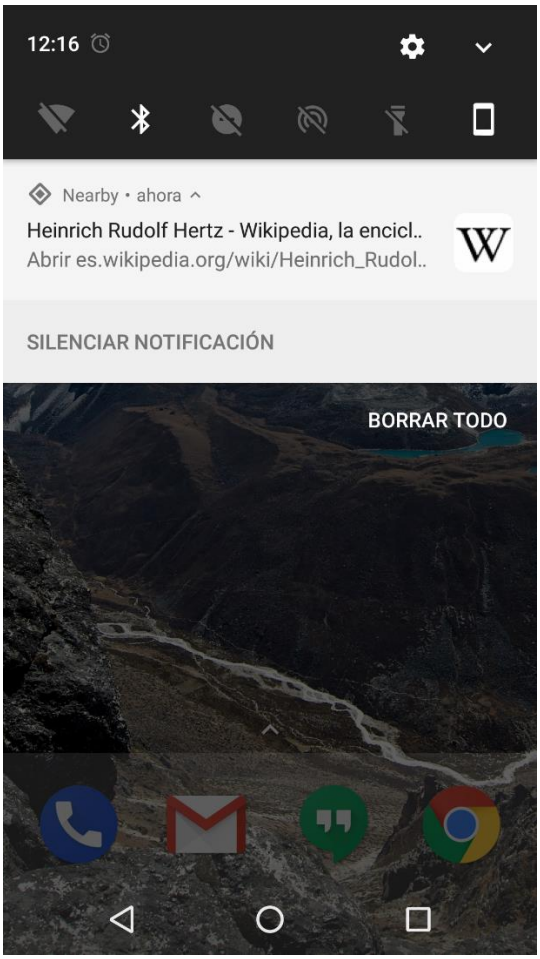

Ilustración 17 - Notificación en Android

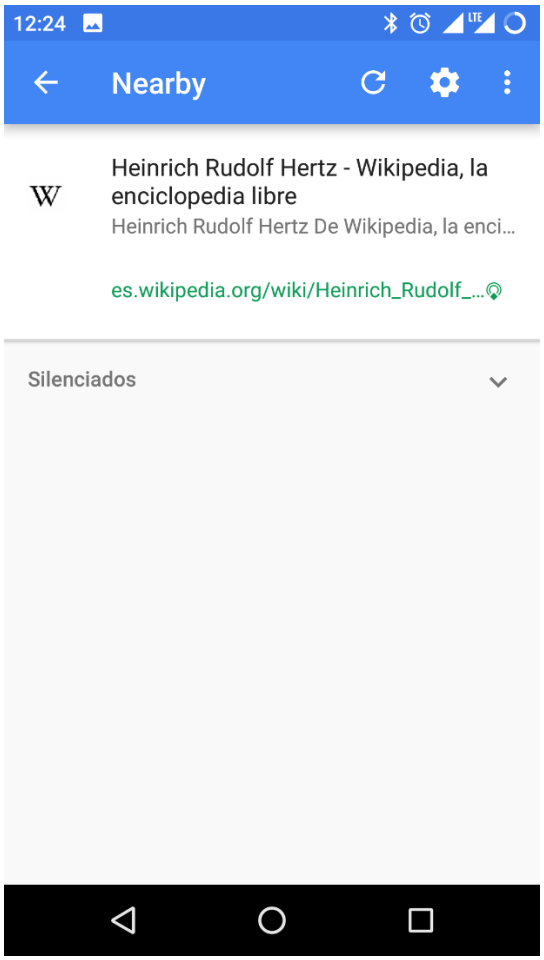

Ilustración 18 - Contenido de la baliza en Nearby

5.7 Aplicación web

5.7.1 Frameworks de desarrollo

Para desarrollar la aplicación web, utilizaremos fundamentalmente dos librerías con las que agilizar y hacer más robusto y mantenible el desarrollo:

- **Django**: Este servidor de aplicaciones web se basa en el lenguaje Python, y contiene la lógica de la aplicación.
- **Bootstrap 3**: Esta librería proporciona elementos para diseñar una interfaz web adaptable o también llamada *responsive*.

La elección de Django viene dada por estar basado en el lenguaje de programación Python. Python es un lenguaje de alto nivel, expresivo y con una sintaxis sencilla y potente.

Utilizamos Bootstrap 3 para nuestra interfaz por ser uno de los frameworks de desarrollo web más extendidos, bien documentado y centrado en el desarrollo web adaptable.

5.7.2 Django

Como hemos mencionado brevemente en el punto anterior, Django [63] es un framework de desarrollo web de código abierto basado en el lenguaje de programación Python. Django utiliza el lenguaje Python para desarrollar la lógica de la aplicación, los ficheros de configuración y modelos de datos.

Django, según su propia nomenclatura, utiliza un patrón de diseño MVT (Model, View, Template), que es similar en su forma de trabajar al habitual modelo MVC (Model, View, Controller).

A grandes rasgos, los componentes principales de Django son los siguientes:

- Un ORM (Object-Relational Mapper), que gestiona el modelado de datos (definidos como clases de Python) y su representación en forma de base de datos relacional. Este sería el *Modelo* según el patrón MVC.
- Un sistema para generar respuestas HTTP mediante un sistema de plantillas web, que funcionaría como la *Vista*.
- Un gestor de URLs basado en expresiones regulares, que funciona a modo de *Controlador*.

Aparte, Django incluye otros elementos importantes para nuestro desarrollo, como serían:

- Un sistema de autenticación extensible, para permitir el registro de los participantes en nuestra actividad.
- Una interfaz de administración web, con las operaciones habituales *CRUD* (Create, Read, Update, Delete) para gestionar de manera simple las instancias de nuestro modelo.
- Implementaciones para paliar los problemas de seguridad más habituales en el ámbito del desarrollo web: cross-site scripting, inyección de SQL y gestión de contraseñas.
- Un sistema de caché opcional para optimizar el funcionamiento de la aplicación.

5.7.3 Bootstrap 3

Para optimizar el desarrollo de la capa de presentación de nuestra web, utilizaremos la librería de desarrollo web Bootstrap [64].

Bootstrap es un conjunto de herramientas *opensource* que incluye código HTML, CSS y Javascript. Bootstrap facilita la programación de interfaces web adaptables, proporcionando también al programador plantillas tipográficas, una rejilla de diseño, formularios, botones y otros componentes de interfaz.

Uno de los motivos principales para utilizar este tipo de librerías es que logran conseguir la correcta visualización del desarrollo en distintos dispositivos y navegadores, con especial hincapié en dispositivos móviles y táctiles.

Para utilizar Bootstrap, la opción más habitual es la de importar sus estilos prediseñados y plantillas manualmente en el código HTML que arroja la aplicación web. En nuestro caso, incluiremos las llamadas a Bootstrap en el código de las plantillas de nuestra aplicación.

5.7.4 Servidor web

En los puntos anteriores del apartado de diseño ya hemos mencionado cómo las aplicaciones de la web física, en su implementación actual en los dispositivos equipados con Android e iOS,

requieren que las páginas web anunciadas por las balizas Bluetooth sean accesibles a través del protocolo SSL.

Por tanto, necesitamos que nuestro servidor web reúna las siguientes condiciones:

1. Debe soportar Python, más concretamente el framework de desarrollo web Django.
2. Valoramos la existencia de un plan de prueba gratuito para utilizarlo durante el desarrollo, con la posibilidad de contratar planes que pongan a disposición de la aplicación más recursos a medida que sea necesario, una vez que sea puesta en producción.
3. Debe ofrecer una administración simple, y la posibilidad de sincronizar el código local mediante FTP o el sistema de control de versiones GIT.
4. Es un factor positivo que sea una solución gestionada, para evitar las tareas adicionales de puesta en marcha y mantenimiento del servidor.
5. Como hemos comentado al inicio de este apartado, es requisito indispensable el tener acceso a las páginas web públicas generadas por nuestra aplicación web mediante el protocolo seguro HTTPS.

A la vista de los requisitos anteriores, hemos encontrado que el servicio Azure Web APP [65] de Microsoft es capaz de cumplirlos todos, por lo que será el que utilicemos durante el desarrollo y despliegue de la aplicación.

Cabe destacar también que la versión más reciente de Visual Studio Community integra soporte para trabajar con Django [66], así como funciones para sincronizar y publicar los proyectos directamente en los diferentes servicios de Azure. Además, el soporte para Django de Visual Studio proporciona un servidor web integrado, de forma que podemos probar y depurar las aplicaciones Python sin necesidad de instalar un servidor aparte.

5.7.5 Diagramas UML

Con el objetivo de tener una visión gráfica de los componentes de la aplicación web, su comportamiento y su interrelación, procedemos a la elaboración de tres diagramas UML que nos facilitarán el posterior proceso de desarrollo. Hemos elaborado los diagramas con la herramienta StarUML.

Diagrama de Casos de Uso:

En este diagrama, realizamos una descripción del comportamiento del sistema en sus actividades principales para sus dos tipos de usuario (Jugador y Administrador):

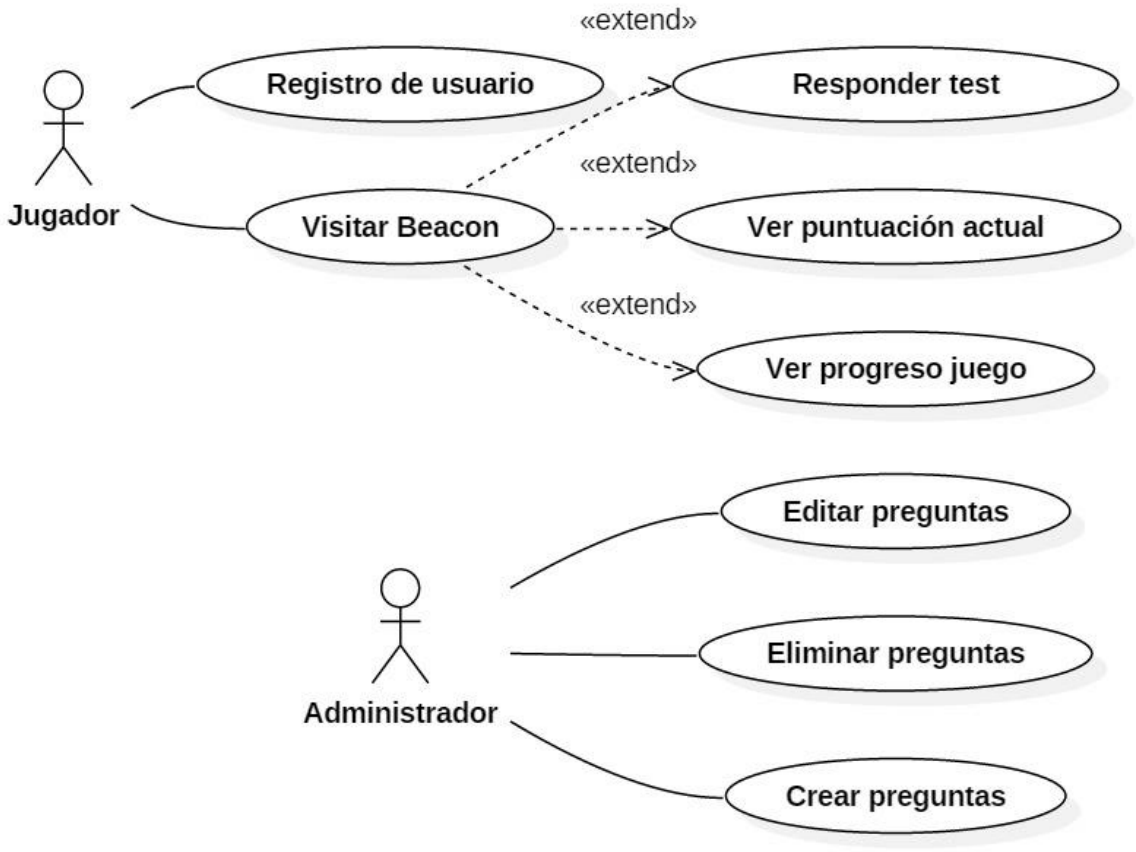

Ilustración 19 - Diagrama de casos de uso del sistema.

Diagrama de Secuencia:

En el diagrama de secuencia vemos el comportamiento de los distintos elementos del sistema en el tiempo. Destacamos el bucle de *juego*, en el que el usuario irá recorriendo las diferentes balizas y sumando puntos al hacerlo.

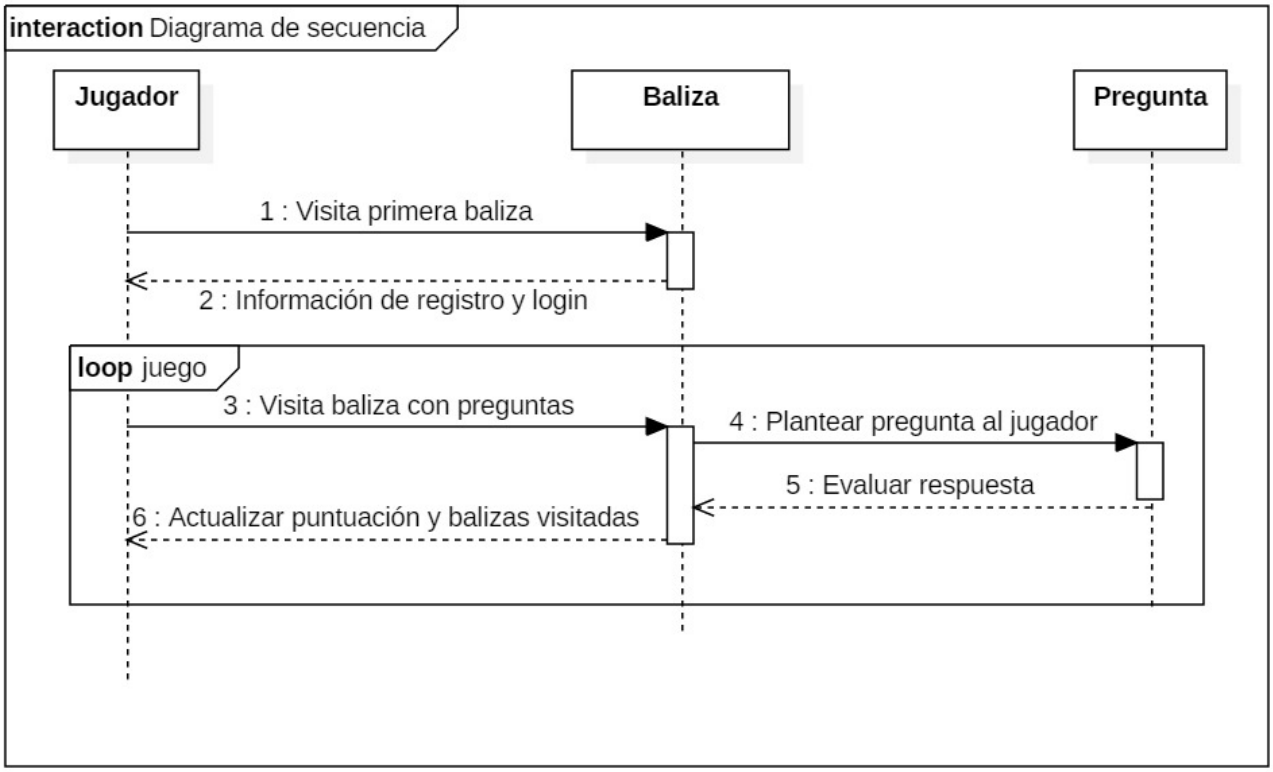

Ilustración 20 - Diagrama de secuencia del programa

Diagrama de Clases:

Por último, creamos un diagrama de clases en el que reflejamos qué información vamos a almacenar para el funcionamiento de nuestro despliegue y cómo vamos a estructurarla utilizando programación orientada a objetos.

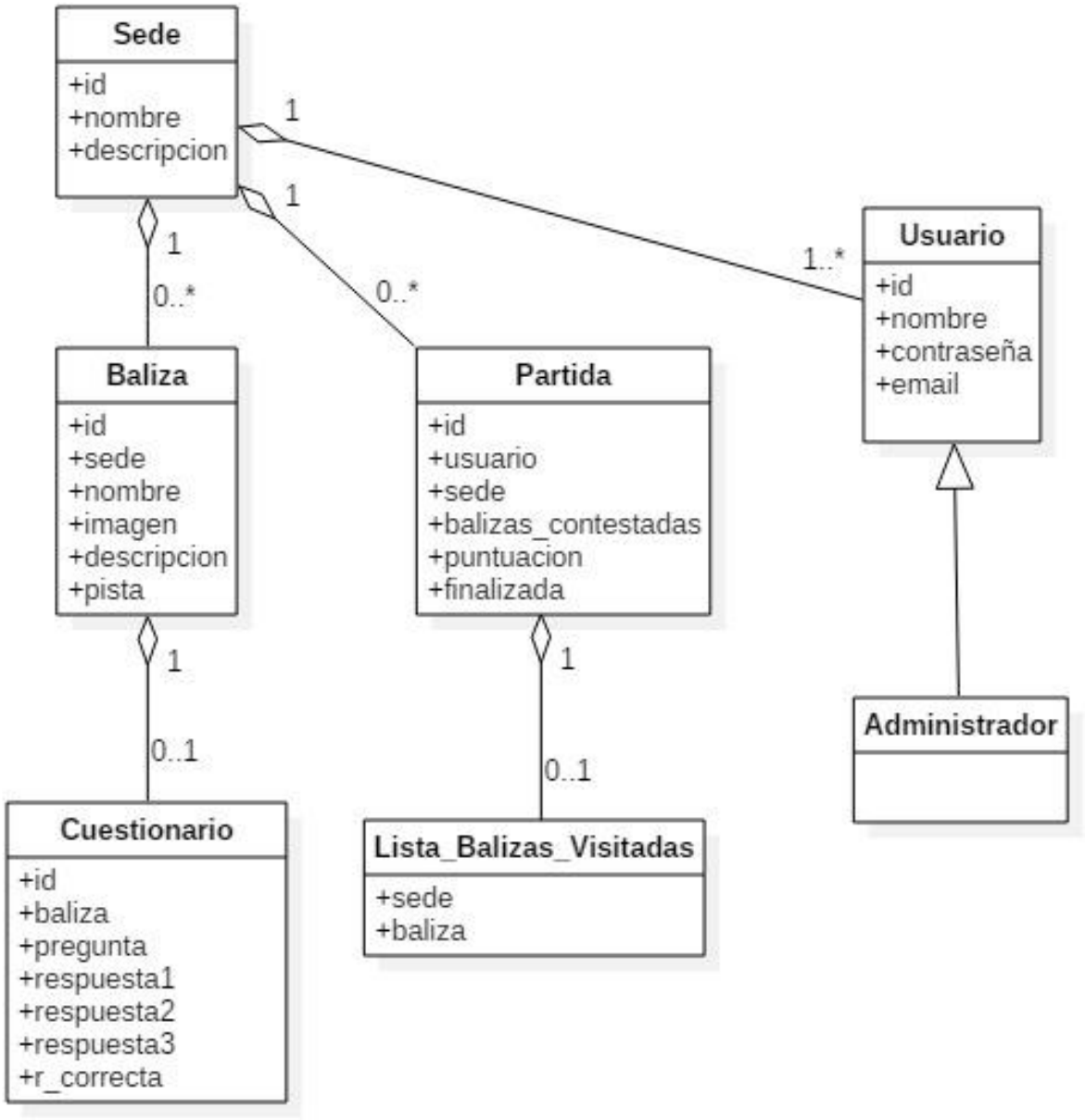

Ilustración 21 - Diagrama de clases.

6 Implementación de aplicación web

6.1 Creación de la aplicación Django

Durante los apartados anteriores, hemos descrito de forma breve el funcionamiento del *framework* de desarrollo web Django. No obstante, toda la información sobre Django que ampliaremos en el presente apartado ha sido obtenida y/o contrastada con la edición actualizada y en español del libro "The Definitive Guide to Django"[67], cuyos autores principales son además los creadores iniciales del framework de desarrollo.

Para crear nuestra aplicación, utilizamos el asistente incluido en Visual Studio Community 2015.

Al crear el nuevo proyecto de Django, el asistente de Visual Studio crea un entorno *virtualenv* de Python. Este entorno contiene su propia instancia de Python, del gestor de paquetes *pip* y de los programas que instalemos mediante este gestor, entre los que se encuentra, por supuesto, la distribución de Django.

El asistente también genera un armazón con un código mínimo de ejemplo de solución Python, como el que obtendríamos con la utilización del siguiente comando:

```
$ django-admin.py startproject SmartGame,
```

donde SmartGame es el nombre que recibirá nuestro proyecto.

En esta estructura de archivos por defecto de Django destacamos por su importancia los siguientes archivos y directorios:

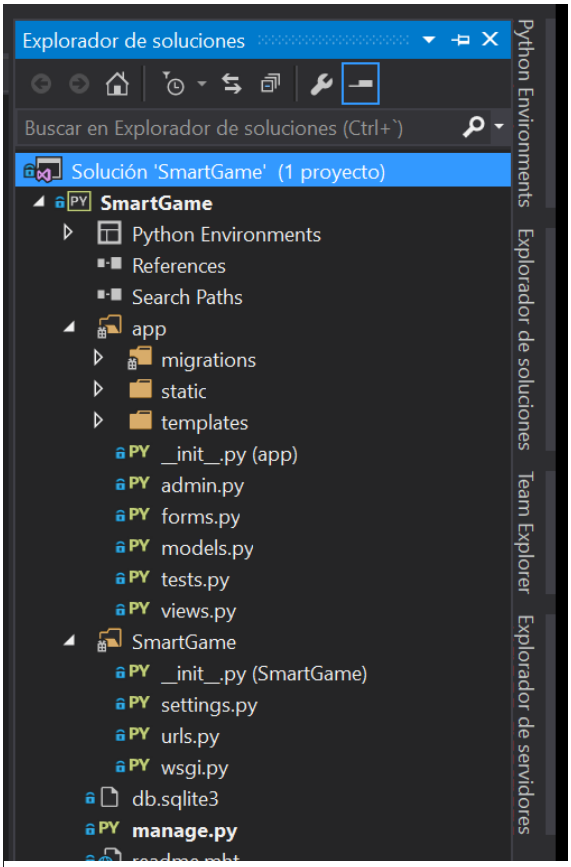

Ilustración 22 - Estructura de archivos del proyecto Django "SmartGame" en VS 2015.

Directorio app: Es habitual subdividir los proyectos Django en *aplicaciones* separadas. Cada aplicación está alojada en un directorio, y contiene funcionalidades relacionadas que pueden ser reutilizadas en otros proyectos. El directorio app, a su vez, contiene los siguientes archivos y subdirectores más relevantes:

- **migrations**: Trataremos este asunto más adelante, pero una *migración* en Django es el mecanismo utilizado para trasladar los cambios del modelo a la estructura de la base de datos.
- **templates**: En este directorio incluimos las plantillas de nuestra solución. Las plantillas están creadas en una mezcla entre html y el sistema de etiquetas propio de Django.
- **admin.py:** En este fichero incluimos las clases que definen el comportamiento de la interfaz de administración por defecto de Django con respecto a nuestros modelos.
- **forms.py:** Este fichero define los formularios de entrada de datos que

utilizará nuestra aplicación para cada uno de los modelos. El fichero hace uso de la librería por defecto para manejo de formularios de Django, que posee, entre otras ventajas, la validación automática de los datos de entrada, generación automática de código y conversión de los datos de formularios a los tipos de datos de Python.
- **models.py:** Aunque veremos los modelos en el siguiente apartado, podemos adelantar que son la manera de crear las estructuras de datos de Django que luego se sincronizarán con la base de datos.
- **views.py:** Este fichero contiene las vistas de la aplicación. Una vista es un fragmento de código que recibe una petición web y devuelve una respuesta, que en la mayoría de las ocasiones será la salida de un documento html.

Directorio SmartGame: este es el directorio mínimo de la solución Django, incluso aunque no hubiésemos definido ninguna aplicación. Sus ficheros más importantes son:

- **settings.py:** recoge todas las configuraciones que definamos para nuestro proyecto.
- **urls.py:** en este fichero, mediante el uso de expresiones regulares, indicamos a Django qué vista tendrá que mostrar en cada posible url de entrada del usuario.

Directorio raíz del proyecto: fuera de los directorios descritos anteriormente, encontramos los siguientes ficheros:

- **db.sqlite3:** mientras que estamos desarrollando la solución, utilizaremos, por su simplicidad y frugalidad en el uso de recursos, una base de datos de tipo SQLite contenida en este fichero.
- **manage.py:** por último, este fichero realiza las mismas tareas que django-admin, pero cargando previamente las configuraciones de nuestro proyecto e incluyéndolo en el PATH del sistema.

6.2 Creación del modelo de datos

Después del repaso por los diferentes archivos que componen nuestro proyecto, pasamos a la creación del *modelo*.

Recordamos que Django incluye funcionalidad de ORM, de manera que su *modelo* representa una abstracción basada en programación orientada a objetos para simplificar y desacoplar los datos de la aplicación de la tecnología de bases de datos utilizada para almacenarla.

Así, para cada entidad de nuestro modelo crearemos una clase que hereda de (models.Model). Los atributos de la clase pueden ser de unos tipos predefinidos por Django, que serán traducidos a su equivalente más próximo en las distintas bases de datos soportadas por el ORM de Django.

Para la creación del modelo nos hemos apoyado en una herramienta llamada DjangoBuilder[68]. Esta herramienta facilita la creación del modelo mediante una interfaz gráfica de usuario, en la que creamos las diferentes clases, sus atributos y las relaciones entre ellas.

La herramienta provee funcionalidad de generación automática de código, obteniendo como salida un fichero models.py para utilizar como base. La herramienta también genera automáticamente código para otras tareas comunes a la hora de programar una aplicación Django.

Al dar la opción de seleccionar el tipo de cada atributo y las relaciones entre clases mediante elementos de interfaz de usuario, el proceso de creación del modelo es menos propenso a errores de lo que sería escribiendo código manualmente. Otra gran ventaja de esta herramienta

es la creación automatizada de una gran cantidad de código repetitivo, por ejemplo, las funciones que definen parámetros como la ordenación de tuplas de cada modelo en las vistas.

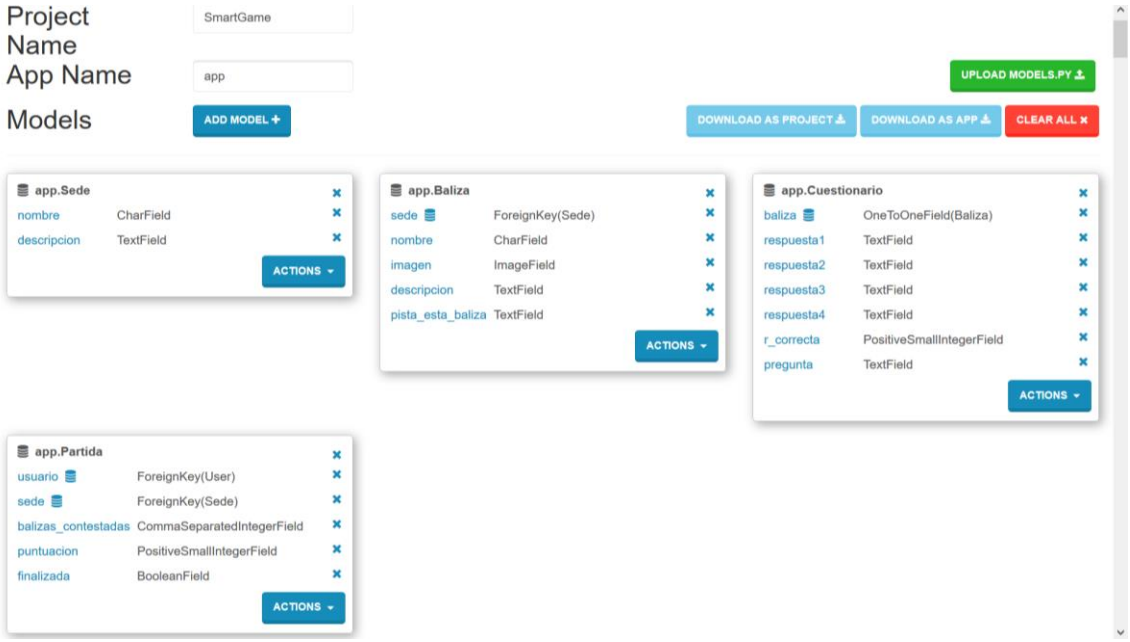

Ilustración 23 - Modelado mediante DjangoBuilder. Fuente: elaboración propia.

Nos parece ilustrativo ver el código fuente completo del modelo Partida:

```
class Partida(models.Model):

    # Campos de datos
    balizas_contestadas = models.ManyToManyField(Baliza)
    puntuacion = models.PositiveSmallIntegerField(default=0)
    finalizada = models.BooleanField(default=False)

    # Campos de relaciones
    usuario = models.ForeignKey(settings.AUTH_USER_MODEL, )
    sede = models.ForeignKey('app.Sede', )

    class Meta:
        ordering = ('-pk',)

    def __unicode__(self):
        return u'%s' % self.pk

    def __str__(self):
        return self.pk

    def get_absolute_url(self):
        return reverse('app_partida_detail', args=(self.pk,))

    def get_update_url(self):
        return reverse('app_partida_update', args=(self.pk,))
```

Queremos llamar la atención sobre un aspecto que consideramos interesante: el campo *balizas_contestadas* aparece en la captura de DjangoBuilder como un *CharField* con una validación del tipo *comma_separated_integer_list*. Sin embargo, podemos ver en el fragmento de código inmediatamente anterior que la implementación realmente se realizó utilizando un campo ManyToManyField respecto a Baliza. Este tipo de campo hace que, cada partida pueda

estar relacionada con varias balizas y al revés, cada baliza puede ser visitada por los jugadores de diferentes partidas.

Los métodos __unicode__ y __str__ sirven devuelven el nombre del modelo utilizando Unicode o cualquier otro conjunto de caracteres indicados. Estos métodos están pensados para asegurar la compatibilidad del desarrollo con las versiones de Python tanto de la rama 2.X como la rama 3.X.

6.3 Migraciones: del modelo a la base de datos

Django utiliza el concepto de *migraciones* (*migrations*) para realizar la sincronización entre los cambios realizados en los modelos y el esquema de la base de datos.

El proceso es fundamentalmente automatizado, pero cuenta con varios comandos que deberán ser invocados cuando se requiera las siguientes acciones:

- *makemigrations*, para crear las nuevas migraciones basadas en los cambios realizados a los modelos.
- *migrate*, para aplicar las migraciones pendientes.
- *sqlmigrate*, para mostrar las sentencias SQL de una migración.
- *showmigrations*, lista las migraciones de un proyecto y su estado.

Nuestro proyecto utiliza una base de datos de tipo SQLite3. Las bases de datos de este tipo incluyen toda su información en un único fichero, lo que las hace muy útiles durante el desarrollo por su simplicidad.

A continuación, vemos el contenido de la base de datos antes de realizar la primera migración y lo comparamos con su contenido tras ejecutar por primera vez los comandos *makemigrations* y *migrate*:

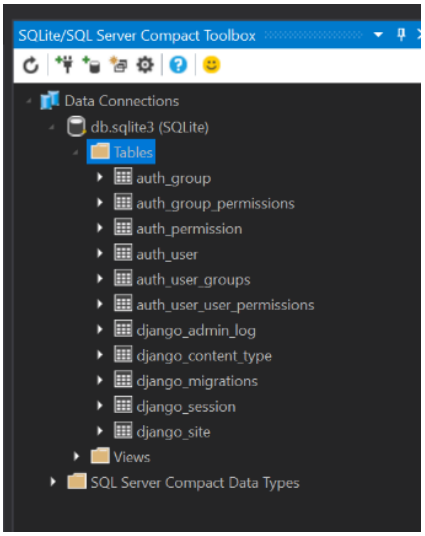

Ilustración 24 - Contenido de BD antes de la primera migración.

Lanzamos las órdenes *makemigrations* y *migrate* y podemos ver la nueva estructura de tablas de nuestra base de datos, incluyendo las relativas a nuestros modelos, y la tabla que implementa la relación *many to many* entre balizas y partidas.

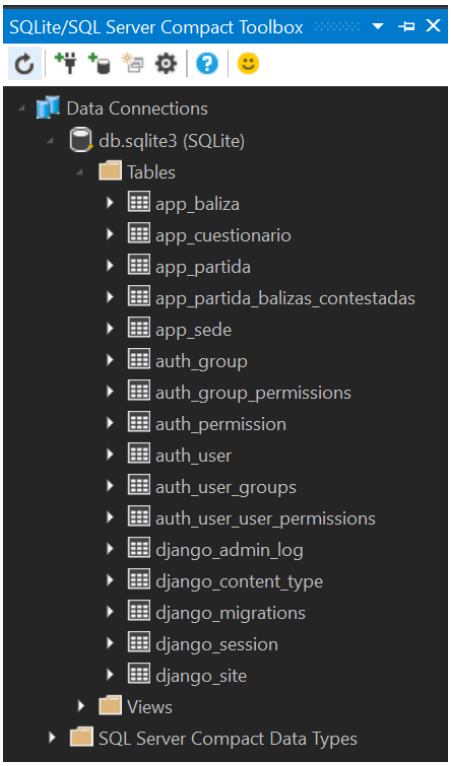

Ilustración 25 - Salida de los comandos makemigrate y migrate y nueva estructura en BD.

6.4 Interfaz administrativa

Django es un framework de desarrollo web pensado para ser modular y para evitar la escritura de código innecesario.

Estas decisiones de diseño cristalizan en la inclusión de multitud de funciones en Django que pueden incluirse en nuestros desarrollos con un ahorro de tiempo considerable.

Una de estas funcionalidades incorporadas por Django es la de interfaz administrativa.

Para añadir alguna de estas funcionalidades a una solución de Django, es necesario que la configuremos el fichero *settings.py*. Dentro de INSTALLED_APPS incluiremos nuestras propias aplicaciones utilizadas en el proyecto, además de dichas funcionalidades predefinidas que se encuentran en el módulo django.contrib.

```
INSTALLED_APPS = [
    'app',
    # Add your apps here to enable them
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
]
```

Para poner en marcha la interfaz administrativa, es necesario que importemos los componentes *django.contrib.admin* y *django.contrib.auth*.

Para definir el comportamiento de la interfaz administrativa, tenemos que editar el fichero *admin.py*. En nuestro caso, y al haber utilizado la herramienta DjangoBuilder para crear la estructura de clases, ya genera automáticamente este fichero para poder editar los datos de

nuestro modelo mediante la utilización de la interfaz administrativa. No obstante, lo personalizamos para que se ajuste plenamente a nuestros requisitos.

Dentro del fichero *admin.py*, tenemos el siguiente código:

```
class SedeAdmin (admin.ModelAdmin):
    view_on_site = False

class BalizaAdmin (admin.ModelAdmin):
    view_on_site = False

class CuestionarioAdmin (admin.ModelAdmin):
    view_on_site = False

admin.site.register(Sede, SedeAdmin)
admin.site.register(Baliza, BalizaAdmin)
admin.site.register(Cuestionario, CuestionarioAdmin)
```

Podemos ver que el fichero está dividido en dos bloques:

En el primero, creamos nuestras propias clases derivadas de admin.ModelAdmin. En estas clases marcamos el atributo *view_on_site* como False, para evitar que la interfaz administrativa muestre botones de edición en lugares donde es preferible que no sea así.

En el segundo bloque, utilizamos las clases que acabamos de crear y las registramos para que se muestren en la interfaz administrativa permitiéndonos trabajar con los objetos instancia de Sede, Baliza y Cuestionario.

La interfaz también mostrará por defecto la funcionalidad de gestión de usuarios y permisos, que mantendremos activa.

ACCESO Y VISIÓN GENERAL DE LA INTERFAZ ADMINISTRATIVA

La interfaz administrativa es invocada por sus usuarios mediante una URL, por lo que también tendremos que configurarla en el fichero *urls.py:*

```
    # Urls de la interfaz administrativa:
    url(r'^admin/', include(admin.site.urls)),
```

Con la configuración que acabamos de ver, damos acceso a la interfaz administrativa desde la url */admin* de nuestro proyecto. No obstante, describiremos con más detalle el fichero urls.py en su propio apartado.

A continuación, vemos cómo, una vez logueados como administradores, tenemos acceso a edición de todos los tipos de datos del modelo de nuestra aplicación desde la interfaz administrativa. También podemos gestionar los distintos usuarios y grupos de la solución:

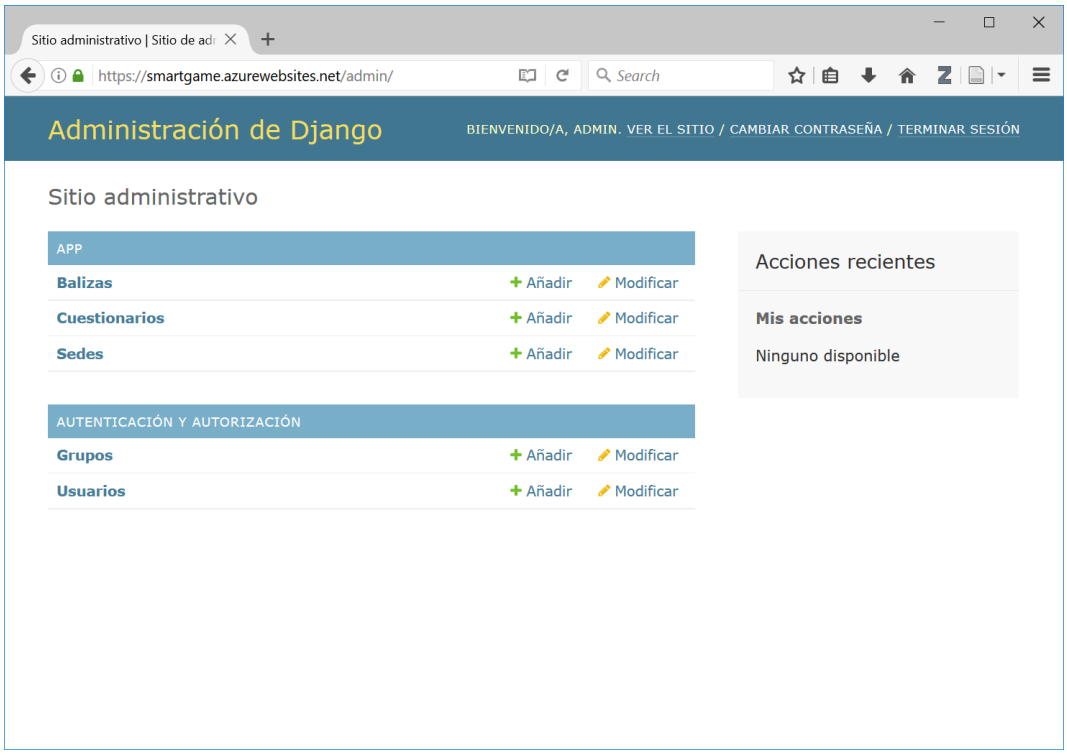

Ilustración 26 - Vista general de la interfaz administrativa de Django en nuestra aplicación

A modo de ejemplo, también ilustramos la creación de una nueva sede en la aplicación:

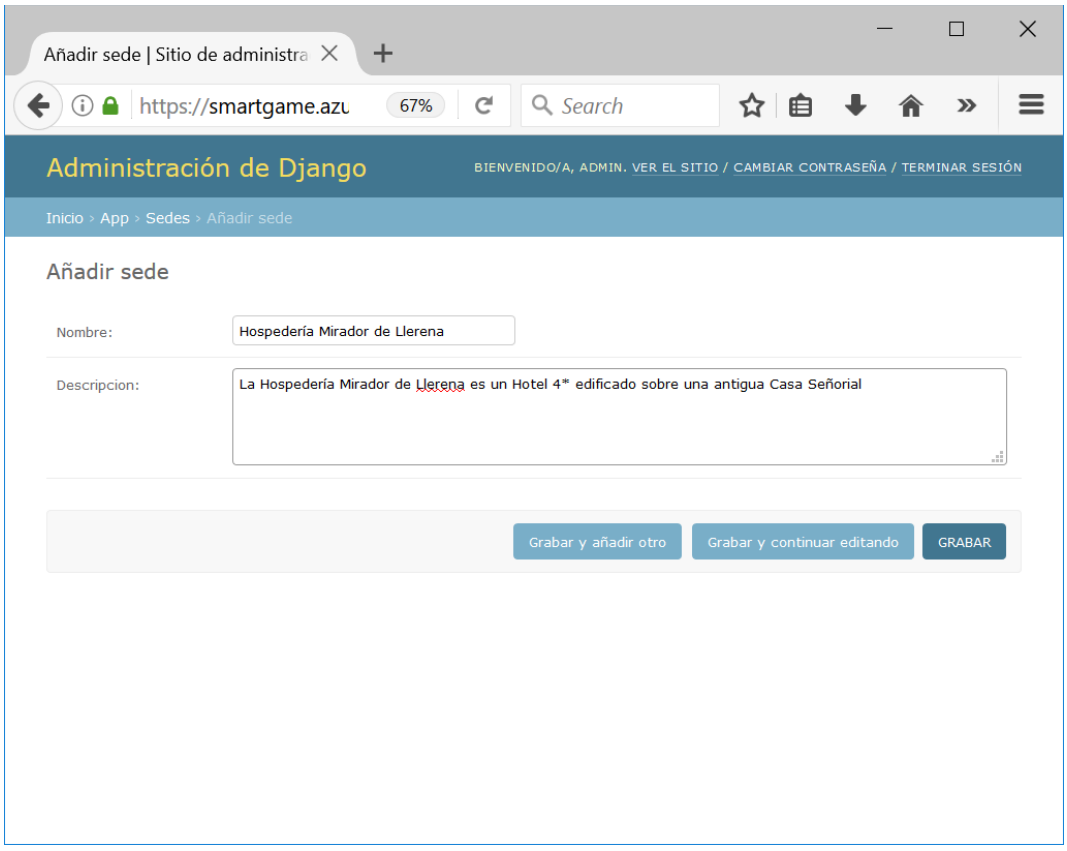

Ilustración 27 – Pantalla de creación/edición de sede

De igual forma, en la interfaz administrativa podemos listar y editar cualquiera de los elementos cuyos modelos hemos registrado:

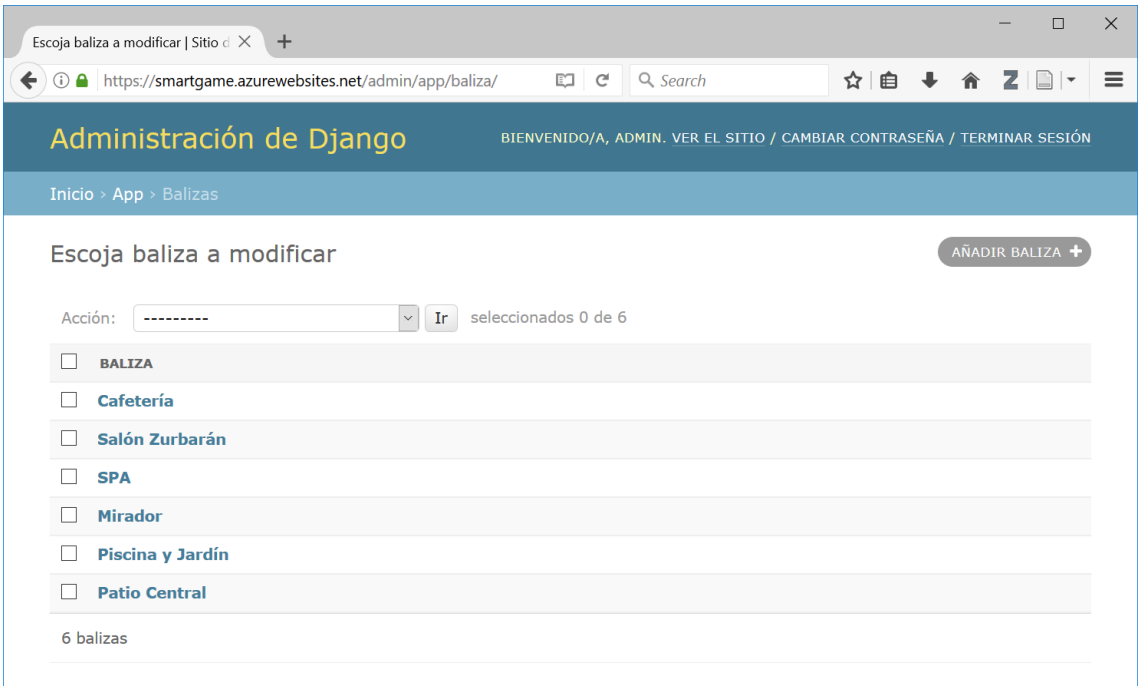

Ilustración 28 – Listado de balizas de ejemplo

Vemos también el cuestionario asociado a una de las balizas:

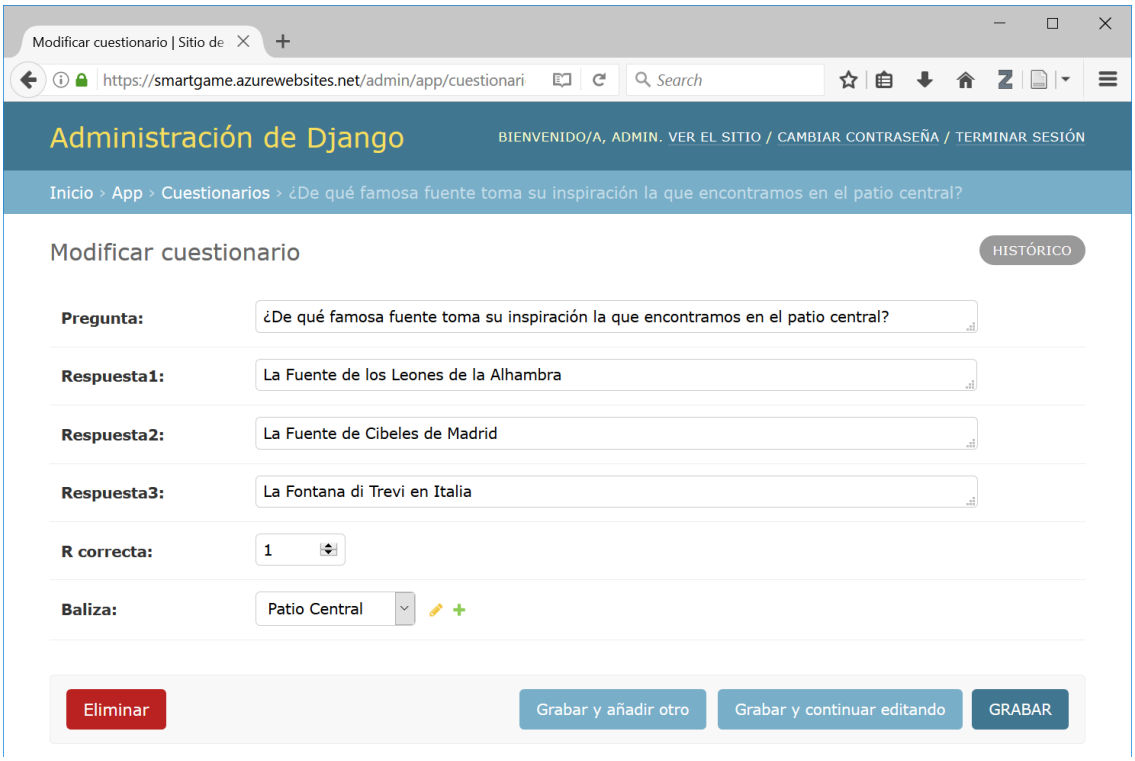

Ilustración 29 – Pantalla de creación/edición de cuestionario de baliza

6.5 Autenticación y gestión de usuarios

Hasta ahora, hemos accedido a la interfaz administrativa con la cuenta de superusuario que hemos creado a través de las herramientas de línea de comandos de Django.

Sin embargo, la aplicación debe permitir que los jugadores se registren rellenando ellos mismos un formulario. También debe permitir la posibilidad de crear administradores sin necesidad de recurrir a la línea de comandos.

Los jugadores son usuarios normales de Django, con permisos restringidos que no podrán acceder a la interfaz de administración.

Mientras tanto, los administradores podrán ser dados de alta manualmente por cualquier otro administrador a través de la interfaz de gestión de Django.

Para crear convertir a un usuario en administrador desde la interfaz administrativa, es necesario asertar su atributo *staff*, que le otorgará los permisos para acceder a la interfaz de administración.

REGISTRO DE USUARIOS

Para poder permitir el registro de usuarios en nuestra aplicación, hemos creado un formulario de registro extendiendo el formulario predefinido "UserCreationForm" [69], incluido en la librería django.contrib.auth.forms.

Nuestro formulario aumenta la funcionalidad de dicho formulario de registro predefinido, solicitando también el email del usuario, para su utilización futura.

```
class FormularioRegistro (UserCreationForm):
    email = forms.EmailField(required=True)

    class Meta:
        model = User
        fields = ("username", "email", "password1", "password2")

    def save(self, commit=True):
        user = super(UserCreateForm, self).save(commit=False)
        user.email = self.cleaned_data["email"]
        if commit:
            user.save()
        return user
```

Habitualmente, el registro de usuarios en un servicio puede efectuarse de dos maneras:

- Por un lado, está el registro en un paso en el que, en el momento de introducir los datos de registro, la sesión del usuario queda iniciada.
- Por otra parte, puede utilizarse el registro con verificación en dos pasos, siendo necesario que el usuario confirme su dirección de correo electrónico siguiendo un enlace que recibe en dicha dirección.

La decisión acerca del tipo de registro más apropiado para este sistema no es trivial. La verificación en dos pasos supone la ventaja de que se verifica que el usuario es propietario de la dirección de correo electrónico que utiliza en el registro.

Sin embargo, nuestra aplicación hace uso de la Web Física, por lo que es posible que el enlace de verificación de correo electrónico se marque desde un navegador (o *webview*) diferente del utilizado para mostrar las páginas de las balizas, detectadas a través de Bluetooth.

Este inconveniente haría el proceso de registro más tedioso para el usuario, al perder la posibilidad de utilizar el login automático en el momento de registrarse.

Por tanto, tomamos la determinación de utilizar el registro en un único paso en nuestra aplicación.

INICIO DE SESIÓN

A la hora de realizar el inicio de sesión, utilizamos una aproximación similar, utilizando un objeto ya existente, pero esta vez sin necesitar extenderlo con nuevos campos.

Para esto, creamos nuestro formulario de inicio de sesión basándonos en la clase AuthenticationForm:

```
class FormularioInicioSesion (AuthenticationForm):
    username = forms.CharField(max_length=254,
                               widget=forms.TextInput({
                                   'class': 'form-control',
                                   'placeholder': 'User name'}))
    password = forms.CharField(label=_("Password"),
                               widget=forms.PasswordInput({
                                   'class': 'form-control',
                                   'placeholder':'Password'}))
```

Este formulario de autenticación es uno de los parámetros que necesita la vista predefinida django.contrib.auth.views.login para completar el proceso de inicio de sesión.

Por último, el gestor de URLs de Django invoca a esta vista toda vez que la página solicitada por el usuario coincida con un determinado patrón, como veremos en el siguiente apartado.

6.6 Estructura de URLs

Otro importante elemento de Django es su manejador de URL.

El fichero urls.py de nuestro proyecto refleja la asociación entre determinados patrones de URL, definidos mediante el uso de expresiones regulares, y las vistas que contienen la lógica de negocio.

De esta forma, definimos las direcciones que llevarán a las distintas partes de nuestra página:

- Páginas de contenido, con las url: home, /contact y /about
- Páginas de registro y login, con: /login, /signup y /logout
- Páginas de las balizas: /b/<identificador>
- Página de la interfaz administrativa: /admin
- Páginas de funcionalidades especiales: /limpiar y /prototipo

En la siguiente tabla, recogemos de manera visual la relación entre URL y vistas en nuestra aplicación:

Tipo	Vista asociada	Expresión regular	Descripción
Contenido	home	r'^$'	Página principal.
Contenido	about	r'^about'	Ayuda para usar la aplicación.
Contenido	contact	r'^contact$’	Información de contacto.
Usuarios	signup	r'^signup/$'	Vista del formulario de registro.
Usuarios	Formulario inicio sesión	r'^login/$'	Vista creada sobre el formulario de inicio de sesión.
Usuarios	Vista cierre sesión	r'^logout$'	Vista creada sobre el formulario de cierre de sesión.
Balizas	baliza	r'^b/(?P<primarykey>\S+)/$'	URLs publicadas por las *balizas* que formen parte de la actividad. El parámetro <primarykey> es el identificador de la baliza a mostrar.
Interfaz adm.	admin	r'^admin/'	Acceso a la interfaz administrativa de Django.
Func. especial	limpiar	r'^limpiar/$'	Elimina todos los datos de partidas y balizas visitadas.
Func. especial	prototipo	r'^prototipo/$'	Elimina datos existentes y crea la sede y balizas de prueba.

Ilustración 30 - Correspondencia entre URL y vistas

6.7 Vistas y sistema de plantillas

Tal y como ya hemos adelantado, las vistas en Django son invocadas por el manejador de URLs cuando una petición HTTP desde el navegador coincide con el patrón definido por alguna de las expresiones regulares de urls.py.

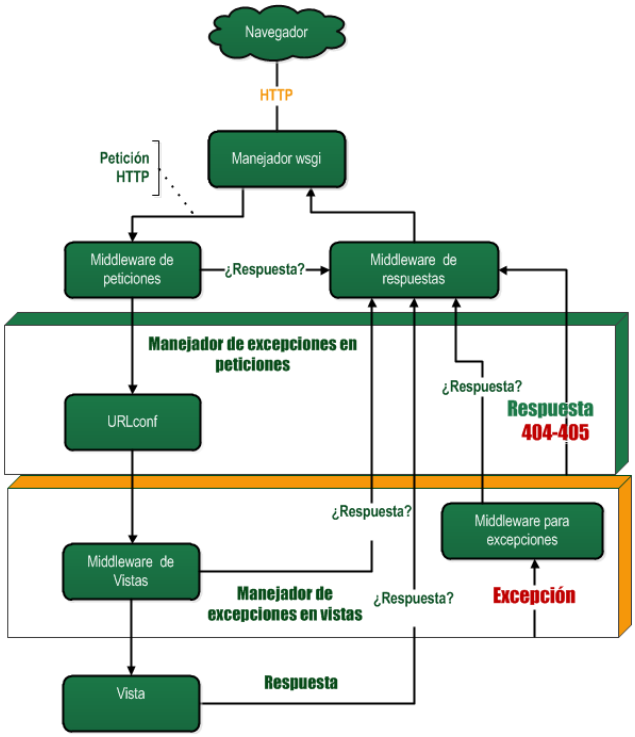

Ilustración 31 - Manejo de peticiones en Django

La vista contiene el código necesario para generar la página que será devuelta de nuevo al usuario, con la colaboración del sistema de plantillas. No obstante, cabe destacar que, aunque lo más normal es utilizar las plantillas para renderizar los datos generados por una vista, ambos tipos de componentes pueden ser utilizados de forma independiente.

En la ilustración, obtenida del libro ‘Guía Definitiva de Django’ [67], podemos ver el recorrido de una petición a través de los distintos bloques funcionales de Django, incluyendo el manejo de excepciones.

Las vistas de nuestra aplicación se encuentran en el fichero views.py. En todos los casos, nuestras vistas son funciones que aceptan como parámetro de entrada un objeto de tipo *HttpRequest* y devuelven un objeto de tipo *HttpResponse* generado por el método *Template.render()*.

Estos objetos de peticiones y respuestas son utilizados por Django para conservar el estado a través del sistema.

Consideramos interesante detenernos en el método render(). Su sintaxis es la siguiente:

```
render (request, template_name, context=None, content_type=None, status=None, using=None)
```

A continuación, explicamos sus argumentos como parte fundamental de la creación de las vistas:

- **request**: Es la petición para la que una vista genera la respuesta. Los objetos de tipo HttpRequest contienen información acerca de la petición. Ejemplos de esta información serían la URL solicitada por el usuario, el método HTTP (GET o POST) y sus parámetros (por ejemplo, campos de respuesta a formularios), las cookies, la información META extraída de las cabeceras del protocolo HTTP, e información sobre la sesión y el usuario actual.
- **template_name**: Hace referencia a la plantilla que se utilizará para renderizar la respuesta. Las plantillas sirven para generar HTML dinámico, y son ficheros que incluyen código HTML y etiquetas del motor de plantillas utilizado. Dicho motor de plantillas sustituye estas etiquetas específicas por el contenido pertinente.
- **context**: Esta variable consiste en un diccionario con datos que la vista le pasa a la plantilla. Este diccionario puede contener objetos de cualquier clase e incluso funciones, que serían invocadas justo antes de renderizar la plantilla.
- **status**: Es el código de estado de la respuesta, que será el código 200 por defecto.
- **using**: Esta variable indica el motor de plantillas que se utilizará para renderizar. En la versión 1.11, Django provee su propio sistema de plantillas (DTL), aunque soporta también el sistema Jinja2.

Anteriormente, hemos enumerado las diferentes vistas utilizadas en el desarrollo y qué URLs las disparan. En los siguientes apartados vamos a describir una vista sencilla, '*home*' y la vista más compleja, '*baliza*', junto con sus plantillas correspondientes.

6.8 Vista 'home' y plantilla asociada

Para ilustrar el funcionamiento del sistema de vistas y plantillas, comenzamos describiendo una de las más simples, la vista *home*:

```
def home(request):
    # Renderiza la página principal
    assert isinstance(request, HttpRequest)
    return render(
        request,
        'app/index.html',
        {
            'title':'SmartGame - Home',
            'balizas_activas' : Baliza.objects.all(),
        }
    )
```

En este caso, vemos cómo la vista *home* recibe un único parámetro, que contiene la petición.

Lo primero que realiza la vista es marcar la petición como instancia de HttpRequest, para poder invocar los atributos y métodos de esta clase.

A continuación, la vista devuelve el resultado de invocar al método *render()* descrito en el apartado anterior. Como argumentos, pasamos a este método la petición, el nombre del fichero html con el código que forma la plantilla y un diccionario con los datos de contexto.

En este caso, los datos de contexto que enviamos a la plantilla serán las variables 'title' y 'balizas_activas'. La variable 'title' es una cadena que se mostrará en la plantilla con el título de la página; balizas_activas es el resultado de una consulta al ORM de Django, que lista todos los elementos del tipo *Baliza* de nuestro modelo de datos.

A continuación, vemos y explicamos el código de la plantilla asociada *index.html*:

```
{% extends "app/layout.html" %}
{% block content %}
<div class="jumbotron">
    <h1>Bienvenido a SmartGame</h1>
    <p class="lead">Diviértete compitiendo con otros participantes. ¿Quién ganará?</p>
    <p><a href="{% url 'about' %}" class="btn btn-primary btn-large">Saber más &raquo;</a></p>
</div>
    {% if user.is_authenticated %}
        <h2>Listado de balizas:</h2>
        <ul>
        {% for b in balizas_activas %}
            <li><a href="b/{{b.pk}}">{{b.sede.nombre}}: {{b.nombre}}</a></li>
        {% endfor %}
        </ul>
        {% if user.is_superuser %}
        <div class="row">
            <div class="col-md-4">
                <h2>Interfaz administrativa</h2>
                <p>
                    Hola, {{user.username}}. Pulsa aquí para acceder a la interfaz
administrativa.
                </p>
                <p><a class="btn btn-warning" href="/admin">Administración &raquo;</a></p>
            </div>

            <div class="col-md-4">
                <h2>Borrar y crear piloto</h2>
                <p>
                    Pulsa para eliminar datos existentes y crear piloto.
                </p>
                <p><a class="btn btn-danger" href={% url 'prototipo' %}>Restablecer
&raquo;</a></p>
            </div>

            <div class="col-md-4">
                <h2>Reiniciar partidas</h2>
                <p>
                    Pulsa para eliminar todos los datos de partidas.
                </p>
                <p><a class="btn btn-danger" href={% url 'limpiar' %}>Eliminar partidas
&raquo;</a></p>
            </div>
        </div>
        {% endif %}
    {% endif %}
{% endblock %}
```

En primera instancia, podemos observar que el fichero incluye código html y etiquetas del sistema de plantillas delimitadas por los caracteres {% %} y {{ }}.

La primera etiqueta en la que nos detenemos es:

```
{% extends "app/layout.html" %}
```

Esta etiqueta invoca el fichero que incluye el código html común a todas las páginas de la aplicación.

Así, el fichero *layout.html* facilita la reutilización de código y la consistencia entre todas las páginas de la aplicación, realizando tareas comunes como:

- Incluir el código Javascript y CSS asociados a la librería Bootstrap 3.
- Crear el menú principal, con la estructura de navegación y gestión de usuarios del sitio.
- Definir el bloque de contenido que reescribirán el resto de plantillas.

Dentro del sistema de plantillas, podemos utilizar condiciones y bucles con las etiquetas {% if %} y {% for %}.

En esta vista, hay funciones que sólo mostramos en el caso de que la petición proceda de un usuario registrado, y que el usuario sea administrador. Para esto, utilizamos las etiquetas {% if user.is_authenticated %} y {% if user.is_superuser %}

El sistema de plantillas también ofrece la posibilidad de iterar sobre elementos pasados como variables de contexto. Por ejemplo, el bucle:

```
{% for b in balizas_activas %}
  <li><a href="b/{{b.pk}}">{{b.sede.nombre}}: {{b.nombre}}</a></li>
{% endfor %}
```

recorre las balizas devueltas como resultado de la consulta ejecutada en la vista, mostrando su nombre y enlace asociado.

Finalizamos este apartado fijándonos en las clases que aplicamos a los elementos html. Estas clases son propias de Bootstrap3, y sirven para dar estilo a cada uno de los elementos.

Este apartado ha servido a modo de ejemplo para explicar mejor la relación entre las vistas y plantillas en nuestra aplicación. En el siguiente, explicaremos la vista principal del sistema, que es la que gestiona las peticiones a las URL que emiten las balizas Bluetooth Low Energy.

6.9 Vista ‘baliza’ y plantillas asociadas

La vista que vamos a describir en este apartado es la principal de nuestra aplicación, puesto que su contenido será el mostrado a los usuarios cuando visiten cada una de las balizas distribuidas por el espacio *gamificado*. La relativa extensión de la vista y su plantilla, sin suponer una excesiva complejidad, ha hecho recomendable dividir el código en funciones y ficheros de apoyo.

Las distintas funciones utilizadas por la vista ‘baliza’ añaden información al diccionario de contexto, que luego es representado a través del sistema de plantillas por diferentes bloques.

Estos serían los bloques utilizados por la vista *baliza* y su plantilla asociada:

Función	Plantilla	Descripción
	bloque-explicación.html	Contiene una breve explicación del funcionamiento del juego.
	bloque-usuario.html	Se muestra en el caso de que el usuario no tenga iniciada sesión, para animarle a que lo haga o se registre.
generar_progreso (Partida, Baliza)	bloque-partida.html	Muestra una barra indicadora del progreso del jugador a lo largo de la actividad, con el número de balizas contestadas y la puntuación lograda. También muestra la pista para encontrar la siguiente baliza.
generar_cuestionario (Cuestionario)	bloque-cuestionario.html	Este bloque visualiza el formulario con la prueba asociada a una determinada baliza. Al responder el cuestionario, el navegador realiza una nueva petición a la vista de la baliza, pero incluyendo la respuesta del usuario como parámetro POST.
procesar_respuesta (Cuestionario, Partida, Baliza, num_respuesta)	bloque-cuestionario.html	La función procesa la respuesta del usuario al cuestionario generado en el apartado anterior. También contiene la lógica que actualiza la información de la partida en la base de datos. Por último, el bloque muestra el cuestionario recién contestado sin posibilidad de interacción, junto con la evaluación de la respuesta dada.
generar_ranking (Sede)	bloque-ranking.html	Muestra los 5 usuarios con mejores puntuaciones de esa sede.

Ilustración 32 - Elementos de la vista 'Baliza'

Gracias al nivel de abstracción que proporciona la utilización de estos bloques, la plantilla principal de la vista baliza es breve y fácil de comprender, depurar y modificar:

```
{% extends "app/layout.html" %}
{% load static %}
{% block content %}
    <div class="col-md-4">
        <h2>{{ nombre_baliza }}</h2>
        <img src="/{{ imagen_baliza }}" />
        <p>{{ descripcion_baliza }}</p>
        {% if not user.is_authenticated %}
            {% include 'app/bloque-explicacion.html' %}
        {% endif %}
        {% include 'app/bloque-cuestionario.html' %}
        {% include 'app/bloque-partida.html' %}
        {% include 'app/bloque-ranking.html' %}
    </div>
{% endblock %}
```

Igualmente, el código principal de la vista 'baliza' en el fichero views.py resulta relativamente compacto para ocuparse de la funcionalidad principal de nuestro despliegue:

```
def baliza(request, primarykey):
    # Obtenemos los datos de la baliza nº 'primarykey' especificada en la URL
    try:
        b = Baliza.objects.get(pk=primarykey)
    except:
        raise Http404()

    try:
        s = Sede.objects.get(pk=b.sede.pk)
    except:
        raise Http404()

    # Comprobamos si el usuario está registrado y obtenemos su identificador
    registrado = request.user.is_authenticated()
    usuario_sesion = request.user
    mostrar_cuestionario = False

    contexto =  {
                'title': b.nombre + " en " + s.nombre,
                'nombre_usuario': usuario_sesion.username,
                'nombre_baliza': b.nombre,
                'imagen_baliza': b.imagen,
                'descripcion_baliza': b.descripcion,
                }

    # LÓGICA PRINCIPAL DE LA VISTA 'BALIZA'
    # Comprobamos si el usuario está registrado
    if registrado:
        # Usuario registrado. Comprobamos si tiene alguna partida en esta sede
        try:
            p = Partida.objects.get(usuario=usuario_sesion.id, sede=b.sede)
        except Partida.DoesNotExist:
            # El usuario NO tiene partida. La creamos
            p=Partida(usuario=usuario_sesion, sede=b.sede)
            p.save()

        # Si el usuario tiene partida comprobamos si está finalizada
        if not p.finalizada:
            # Comprobamos si el usuario ya ha contestado esta baliza
            contestada = p.balizas_contestadas.filter(id=b.id)
            if (contestada.count() < 1):
                mostrar_cuestionario = True

                id_cuestionario = Cuestionario.objects.get(baliza__id=b.id).id
                c = Cuestionario.objects.get(baliza__id=b.id)
                contexto.update(generar_cuestionario(id_cuestionario))

                # Si el jugador acaba de enviar una respuesta,
                if 'r_sel' in request.POST and request.POST['r_sel']:
                    contexto.update (procesar_respuesta(c,p,b,request.POST['r_sel']))

        # Creamos las variables para el sistema de plantillas.
        # Si estamos en una partida, mostramos el progreso.
        contexto.update(generar_progreso(p,b))
    # En todo caso, mostramos el ranking de puntuaciones de la sede
    contexto.update(generar_ranking(s.id))

    assert isinstance(request, HttpRequest)
    return render(request,'app/baliza.html',contexto)
```

A la vista del código, podemos deducir cómo la vista muestra unos u otros elementos al usuario dependiendo de su progreso en la partida.

Recordamos también que, en el caso de que el desarrollo de la actividad requiera mostrar un cuestionario, la acción de envío asociada al mismo generará una petición que será gestionada

por la misma vista *baliza*, pero esta vez incluyendo la respuesta del usuario codificada en la petición de tipo POST.

Para procesar la respuesta del usuario, comprobamos si, efectivamente, la petición incluye la clave 'r_sel' generada al responder el cuestionario, en cuyo caso, invocamos a la función que procesa la respuesta:

```
if 'r_sel' in request.POST and request.POST['r_sel']:
    contexto.update (procesar_respuesta(c,p,b,request.POST['r_sel']))
```

6.10 Seguridad

El framework de desarrollo web Django posee medidas de seguridad específicas que pasamos a describir:

Protección contra XSS:

Los ataques de tipo Cross Site Scripting consisten en la inyección de scripts del lado del cliente en el navegador de otros usuarios, habitualmente almacenando el script malicioso en la base de datos o haciendo que la víctima pulse un enlace de manera que el código del atacante se ejecute en el navegador de la víctima.

El sistema de plantillas de Django protege al usuario contra la mayoría de ataques de este tipo, escapando la mayoría de caracteres potencialmente peligrosos.

Protección contra CSRF:

Los ataques Cross Site Request Forgery permiten a un usuario ejecutar acciones utilizando las credenciales de otro usuario sin su conocimiento ni consentimiento.

Para mitigar este tipo de ataques, Django utiliza por defecto un módulo que envía un secreto en cada petición POST. De esta forma, cada vez que entra una petición al sistema, éste comprueba si es legítima verificando el token CSRF, que es específico de cada usuario.

Protección contra inyección SQL:

En este tipo de ataques, el usuario malicioso es capaz de ejecutar código SQL arbitrario contra una base de datos, integrando su código en formularios, parámetros, etc. El ORM de Django escapa correctamente el código SQL antes de pasarlo al driver de la base de datos utilizada, minimizando además la necesidad de trabajar con peticiones en crudo y código SQL personalizado.

Uso de SSL/HTTPS:

La utilización de estos protocolos dificulta la intercepción del tráfico de red (contraseñas u otra información sensible) por parte de usuarios malintencionados.

Django ofrece la posibilidad de forzar una redirección de todas las conexiones HTTP hacia el protocolo HTTPS. También ofrece la posibilidad de utilizar cookies seguras y de utilizar HSTS (HTTP Strict Transport Security), una cabecera HTTP que informa al navegador de que todas las futuras conexiones a un determinado sitio deberán de realizarse mediante HTTPS.

Contenido subido por el usuario:

En la versión actual de Django (1.11), existiría la posibilidad de subir código HTML malicioso camuflado como una imagen de tipo PNG en un campo de subida de imágenes por parte del usuario. Actualmente, no hay soluciones técnicas infalibles a nivel framework para mitigar estos ataques. La documentación oficial acerca de seguridad de Django [70] recomienda, por un lado, incluir el contenido multimedia en un subdominio o CDN, para dificultar el ataque al código principal de la aplicación, y por otro lado, configurar de manera más restrictiva el tipo de archivos que los usuarios pueden subir y que el servidor web puede entregar.

6.11 Publicación en el servidor web

ENTORNOS VIRTUALES EN PYTHON

Antes de describir la publicación de nuestro desarrollo, es interesante detenerse en el concepto de entornos virtuales de Python, que ya mencionamos brevemente al comienzo del epígrafe de implementación.

Los entornos virtuales son espacios independientes de otros entornos virtuales y de los paquetes instalados globalmente en nuestra instalación o instalaciones de Python. De esta manera, nuestra aplicación tiene su propio entorno virtual *env1*, basado en Python 3.4 y en el que está instalada la versión de Django 1.11, junto con otras librerías imprescindibles para el buen funcionamiento de la aplicación: la librería de manejo de imágenes *pillow*, la de elementos de diseño web responsive *Bootstrap3* y la de presentación de formularios *django-crispy-forms*.

El hecho de utilizar estos entornos virtuales simplifica la publicación de nuestra aplicación, puesto que no será necesario instalar las dependencias de la aplicación manualmente en el servidor, sino que subimos al servidor la aplicación completa con su entorno virtual.

SUBIDA A SERVIDOR PÚBLICO

A la hora de poner en producción nuestro despliegue, utilizamos los servicios cloud de Microsoft Azure, como ya adelantamos en el apartado de diseño.

Dentro de los diferentes recursos disponibles en Azure, encontramos el de App Services. Este servicio es un servidor virtualizado administrado que ofrece una solución de plataforma como servicio (PaaS). Dentro de los App Services, tenemos la opción de alojar varios tipos de aplicaciones:

- **Mobile Apps**, para hospedar backends de aplicaciones móviles.
- **API Apps**, para alojar APIs de tipo RESTful.
- **Logic Apps**, para automatizar procesos empresariales e integrar sistemas y datos mediante tecnología cloud.
- **Web Apps**, para alojar sitios y aplicaciones web como la que nos ocupa.

Las Web Apps de Microsoft Azure permiten publicar aplicaciones web sin tener que gestionar la infraestructura, con despliegues automatizados mediante GIT u otras opciones de publicación manuales, como FTP o Web Deploy.

Además, las Web Apps ejecutan aplicaciones escritas en .NET, Node.js, PHP, Java, HTML o Python, como es el caso del framework Django utilizado en nuestra solución.

Para proceder a la publicación, es necesario crear una Web App desde la consola de administración de Azure. Hemos creado una web app llamada *smartgame*, dentro de un plan de tipo F1 de App Service.

El plan F1 es un plan gratuito, que limita los recursos que la aplicación puede utilizar a un almacenamiento de 1GB en infraestructura compartida y sin posibilidad de usar un dominio propio. Aun así, el plan F1 está perfectamente dimensionado para nuestra aplicación en su estado actual de prototipo. Es posible, a través de la propia interfaz de Azure, pasar a un plan superior en el caso de que las necesidades del proyecto así lo requieran:

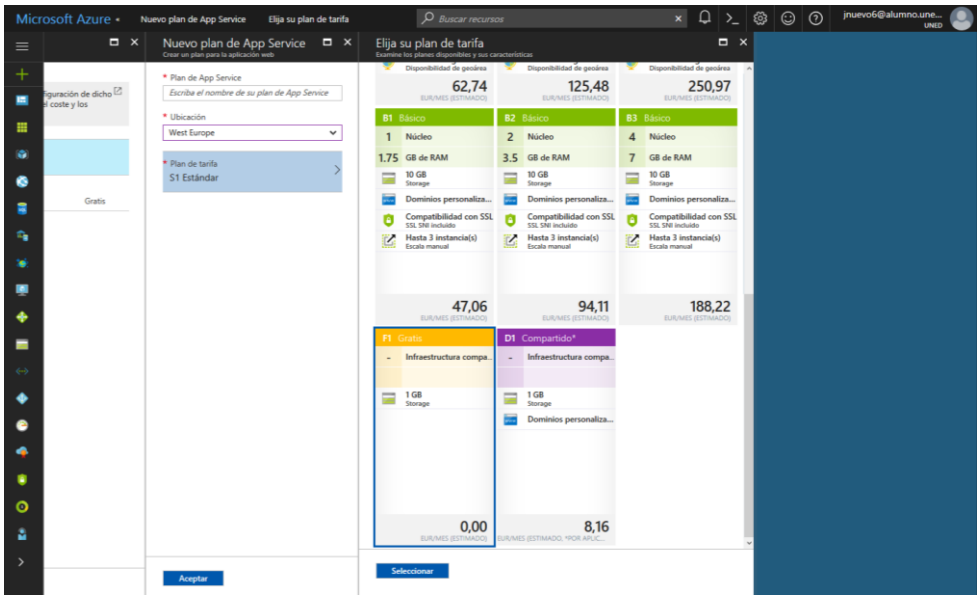

Ilustración 33 - Planes de Azure Web Apps

Una vez creada nuestra Web App dentro del plan de tipo F1, podemos acceder a configurar sus distintos parámetros, como redes, bases de datos o, como vemos en la siguiente pantalla, configuración general de la aplicación:

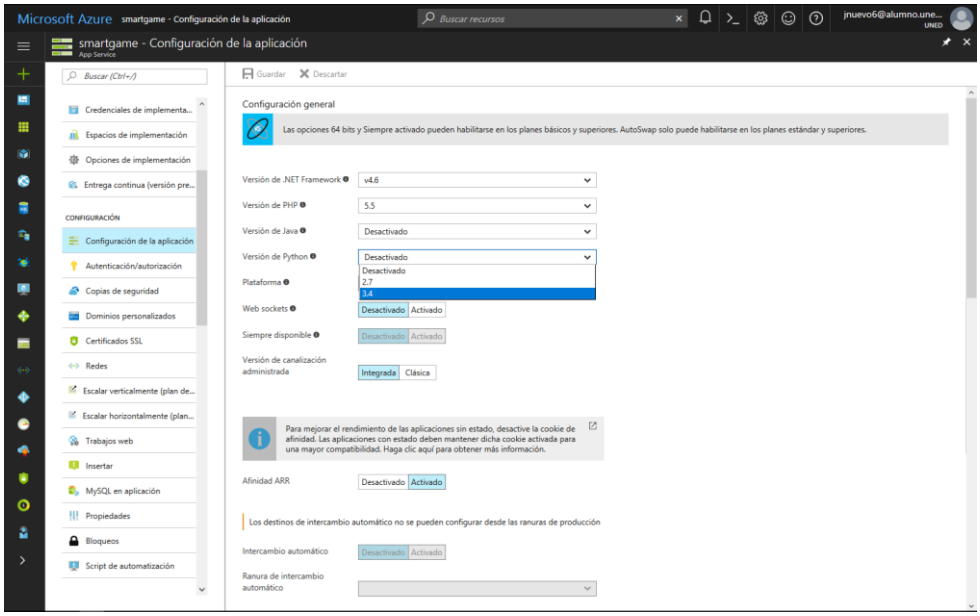

Ilustración 34 - Configuración general de Web App en la consola de Azure

Ya hemos mencionado que hay distintos métodos de despliegue de la aplicación web. En nuestro caso, vamos a utilizar la funcionalidad Web Deploy integrada en Visual Studio Community 2015.

Para esto, marcamos la opción 'Publicar' de nuestro proyecto, marcando como destino el Servicio de aplicaciones de Microsoft Azure. Al haber iniciado sesión previamente en el servicio de Azure en Visual Studio, podremos seleccionar la web app creada anteriormente como destino al que subirá el proyecto:

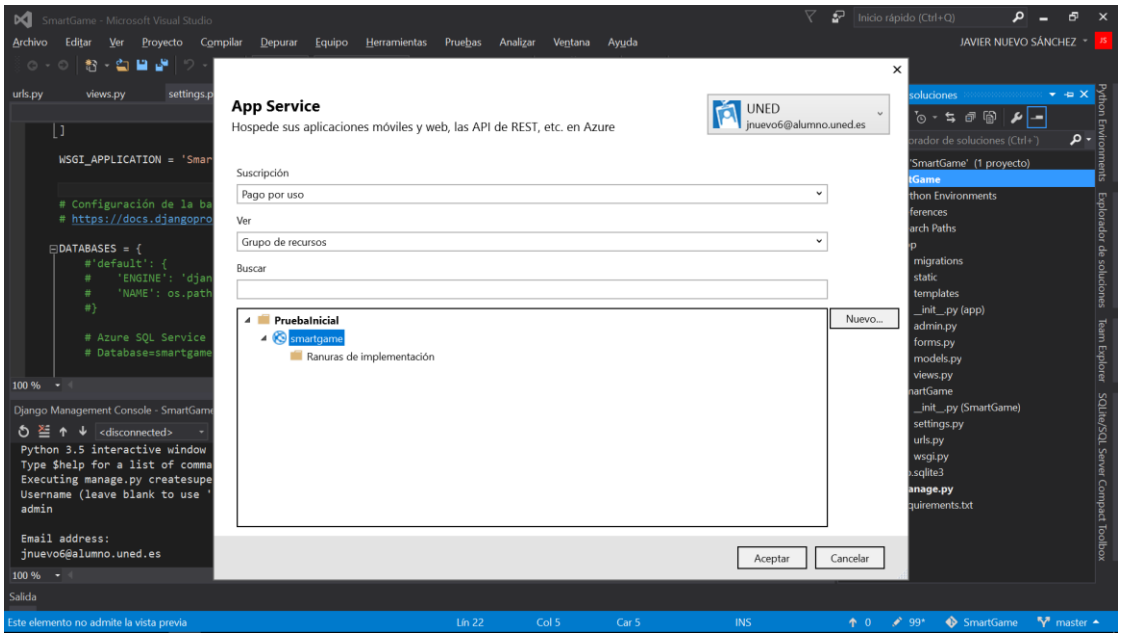

Ilustración 35 - Selección destino de publicación desde VS2015

En la siguiente pantalla, vemos cómo el entorno de desarrollo selecciona automáticamente los parámetros de configuración necesarios para subir la aplicación al servicio de alojamiento.

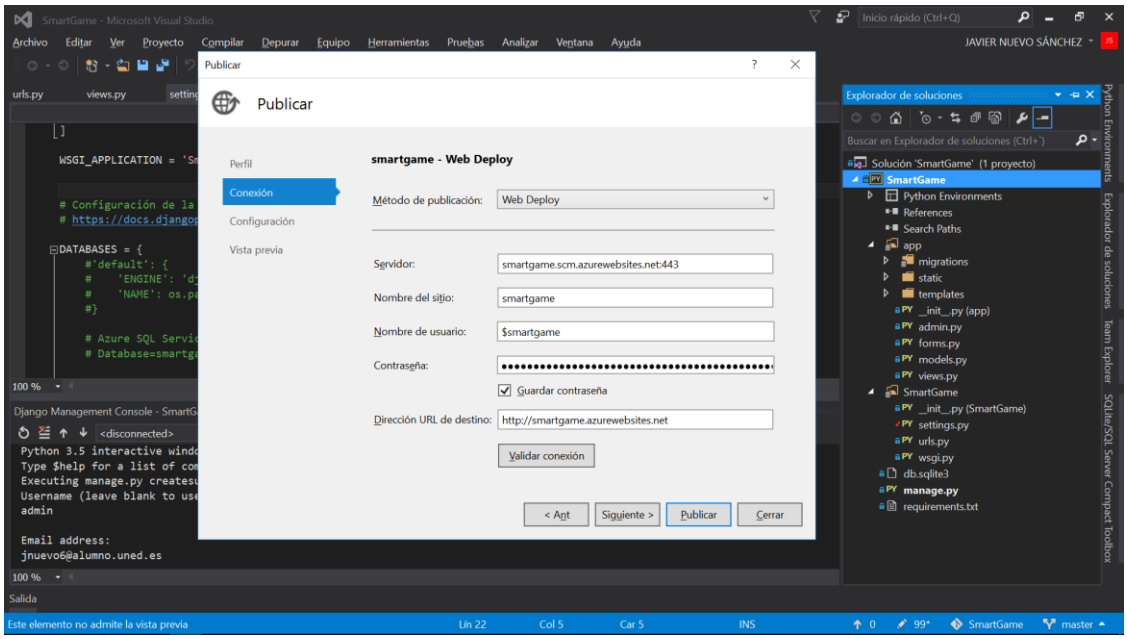

Ilustración 36 - Parámetros de publicación de la aplicación

Tras completar el proceso de publicación, la aplicación está disponible en la siguiente URL:

https://smartgame.azurewebsites.net

El código completo está alojado en el repositorio privado de Github:

https://github.com/neoproducciones/SmartGame

7 Implementación piloto

7.1 Introducción

Proponemos realizar un despliegue piloto de la solución SmartGame aplicado al turismo.

La Red de Hospederías de Extremadura es una cadena de hoteles 4* construidos sobre edificios históricos y singulares gestionados por la Junta de Extremadura.

Según la Tesis Doctoral de Abujeta [71] , entre los objetivos originales de la Red de Hospederías de Extremadura está la puesta en valor del patrimonio arquitectónico de la región mediante procesos de intervención con criterios utilitaristas, siguiendo un modelo similar en su concepción al de los Paradores de Turismo de España.

Ilustración 37 - Actuaciones en la rehabilitación del patrimonio. Fuente: Abujeta 2015

Así, entre los edificios que sirvieron como base para la construcción de los hoteles podemos encontrar conventos, palacios, factorías, molinos hidráulicos o casas señoriales. Los edificios originales han sido objeto de cuidadosos trabajos de rehabilitación y adaptación a su nueva actividad.

Creemos que este despliegue piloto sería interesante por varios motivos:

- Los edificios originales poseen relevancia histórica a nivel regional. El despliegue objeto de este proyecto puede servir como herramienta para la puesta en valor del proceso de rehabilitación y la difusión de la historia de las edificaciones entre la población.
- Los edificios poseen una distribución y calidad constructiva (edificios grandes y, en la mayoría de los casos, con muros de piedra) que hacen posible la instalación de balizas Bluetooth y la separación efectiva de rango entre las diferentes balizas emisoras.
- Serviría para comprobar si la gamificación de un espacio físico mediante balizas Bluetooth aporta valor a la actividad principal desarrollada en este tipo de establecimientos, como es el alojamiento hotelero.

- En caso afirmativo, pretendería reforzar un segmento de mercado estratégico para Extremadura, como es el turismo familiar.
- Otro beneficio esperado importante sería el refuerzo de la marca del establecimiento, asociándola a los valores de innovación, capacidad tecnológica y labor de difusión cultural.
- La actividad no tendría por qué resultar invasiva ni molesta de cara a los demás clientes.

7.2 Descripción del montaje

Para la prueba piloto, crearíamos una sede en la Hospedería Mirador de Llerena.

Este hotel 4* de 25 habitaciones se sitúa en Llerena, localidad de la Campiña Sur de la provincia de Badajoz. El hotel está edificado sobre una antigua casa señorial construida en 1899.

Proponemos la instalación de las siguientes balizas:

# Baliza	Nombre
0	Recepción
1	Patio Central
2	Piscina
3	Mirador
4	SPA
5	Salón Zurbarán
6	Restaurante

Ilustración 38 - Listado de balizas piloto

Para el montaje, proponemos las balizas Energy Efficient King elegidas en el apartado de Diseño, con un alcance de 35 m y una duración de batería estimada de 60 meses.

Cada baliza tiene un importe aproximado de 25 €, por lo que el importe en balizas Bluetooth ascendería a un total de 175 €.

Puesto que estas balizas funcionan mediante el uso de baterías, la instalación de cada una de ellas se reduce a fijarlas en un lugar elevado, para evitar su sustracción o vandalización. La fijación puede ser realizada por el personal de mantenimiento del propio hotel mediante el uso de elementos adhesivos, para hacer el proceso reversible y respetuoso con el edificio.

También es importante señalar el punto donde se encuentra la baliza, a una menor altura, con un cartel. Este cartel, además, incluirá un código QR con la misma URL a la que apunta la baliza, para poder seguir el juego en el caso de que alguna baliza falle o el usuario no disponga de la tecnología Bluetooth Smart en su dispositivo.

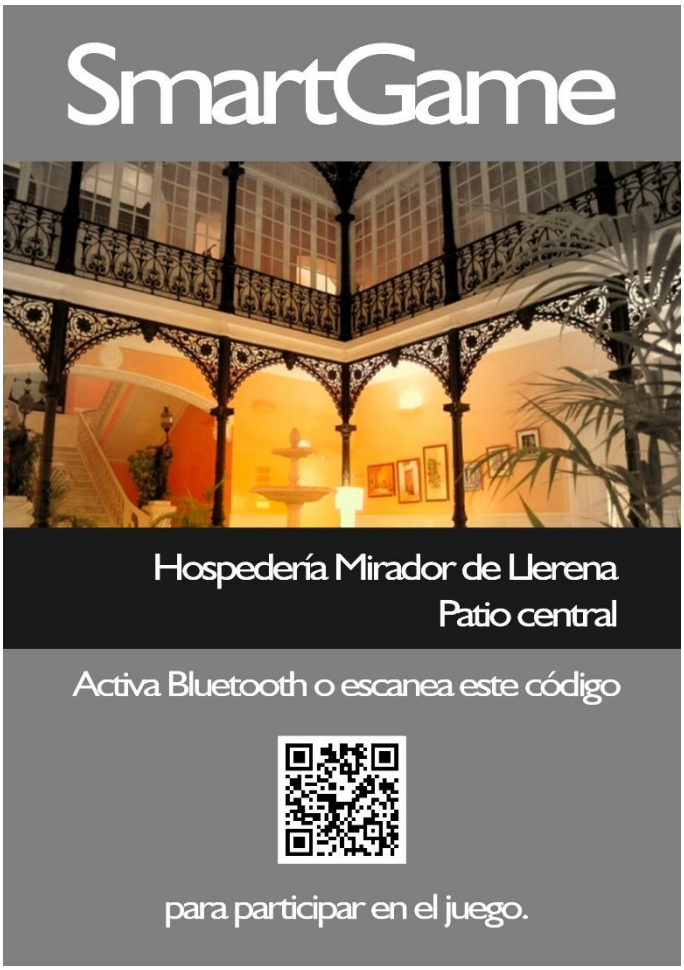

Ilustración 39 - Cartel indicando la posición de una baliza

7.3 Contenido de ejemplo en la aplicación web

La actividad que planteamos en nuestra aplicación se basa en contestar preguntas durante la visita al establecimiento.

Desarrollamos un ejemplo con el contenido de las 6 balizas de juego propuestas para su despliegue. La baliza 0 (recepción del hotel) emitiría la url de la página home de nuestro desarrollo, con la explicación de cómo participar en la actividad.

En el prototipo de software creado, hemos incluido la funcionalidad de crear la sede correspondiente a este hotel, así como las balizas y sus preguntas y respuestas asociadas que podemos ver a continuación:

Nº Baliza: 1	**Nombre:** Patio Central
Descripción:	El patio central de la Hospedería es un espacio luminoso que comunica las dependencias principales de la casa.
Pregunta:	¿De qué famosa fuente toma su inspiración la que encontramos en el patio central?
Respuesta 1:	La Fuente de los Leones de la Alhambra
Respuesta 2:	La Fuente de Cibeles de Madrid
Respuesta 3:	La Fontana di Trevi en Italia
Respuesta Correcta:	1
Pista para encontrar esta baliza:	Mirando hacia arriba, encontrarás luz y color.

Nº Baliza: 2	**Nombre:** Piscina y Jardín
Descripción:	El antiguo patio exterior, ahora zona de piscina, alojaba las caballerías, cocinas y almacenes de la casa señorial.
Pregunta:	En este espacio se pueden observar grandes tinajas, que se usaban en la bodega de la vivienda original. ¿Qué sustancia se guardaba en ellas?
Respuesta 1:	Vinagre
Respuesta 2:	Vino
Respuesta 3:	Aceite
Respuesta Correcta:	3
Pista para encontrar esta baliza:	Cuando viene el calor, todos preguntan por este lugar.

Nº Baliza: 3	**Nombre:** Mirador
Descripción:	Uno de los sitios significativos de la casa es el mirador, formado por un salón de grandes ventanales y una terraza descubierta.
Pregunta:	¿En qué año se inició el proyecto de construcción del edificio original?
Respuesta 1:	1745
Respuesta 2:	1899
Respuesta 3:	1952
Respuesta Correcta:	2
Pista para encontrar esta baliza:	El punto más alto de la casa.

Nº Baliza: 4	**Nombre:** SPA
Descripción:	La casa original es uno de los principales ejemplos de la arquitectura modernista extremeña, reflejando el interés de sus propietarios por el arte y el confort.
Pregunta:	¿Cuál es el apellido de la familia que mandó construir la casa?
Respuesta 1:	Zambrano
Respuesta 2:	Ruiz
Respuesta 3:	Serrano
Respuesta Correcta:	1
Pista para encontrar esta baliza:	En invierno o verano, el lugar donde el estrés se disuelve.

Nº Baliza: 5	**Nombre:** Salón Zurbarán
Descripción:	Este precioso salón toma su nombre de Francisco de Zurbarán, que vivió en la localidad de Llerena.
Pregunta:	¿Cuál fue la ocupación de este artista del Siglo de Oro?
Respuesta 1:	Escritor
Respuesta 2:	Escultor
Respuesta 3:	Pintor
Respuesta Correcta:	3
Pista para encontrar esta baliza:	Un salón con nombre de artista.

Nº Baliza: 6	**Nombre:** Cafetería
Descripción:	La cafetería está situada en la planta baja y evoca el ambiente de la casa original.
Pregunta:	Las coloridas baldosas de la cafetería fueron creadas en Sevilla y aún se mantienen en uso. ¿Cómo se llama este lujoso tipo de suelo?
Respuesta 1:	Pavimento neumático
Respuesta 2:	Pavimento hidráulico
Respuesta 3:	Ninguna de las anteriores
Respuesta Correcta:	2
Pista para encontrar esta baliza:	Sigue el olor a café para encontrar el siguiente espacio.

A modo de ejemplo, vemos algunas capturas del despliegue piloto de la aplicación en nuestro dispositivo de pruebas, con Google Chrome 58 en Android 7.1.2:

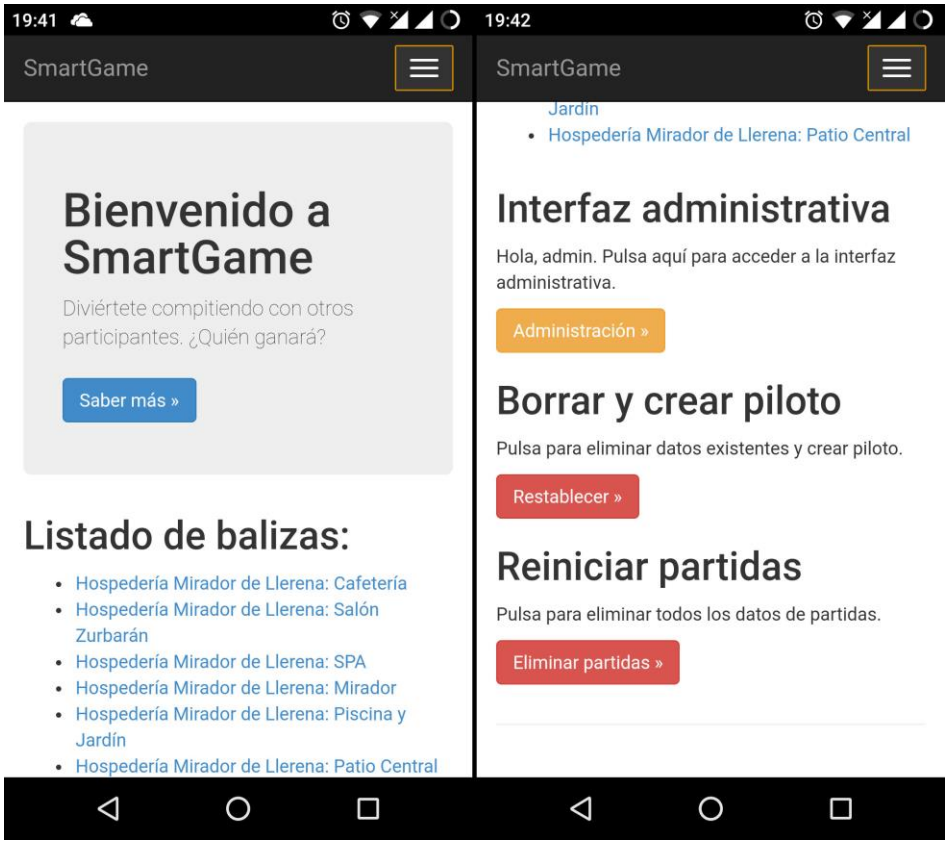

Ilustración 40 - Capturas vista 'home'

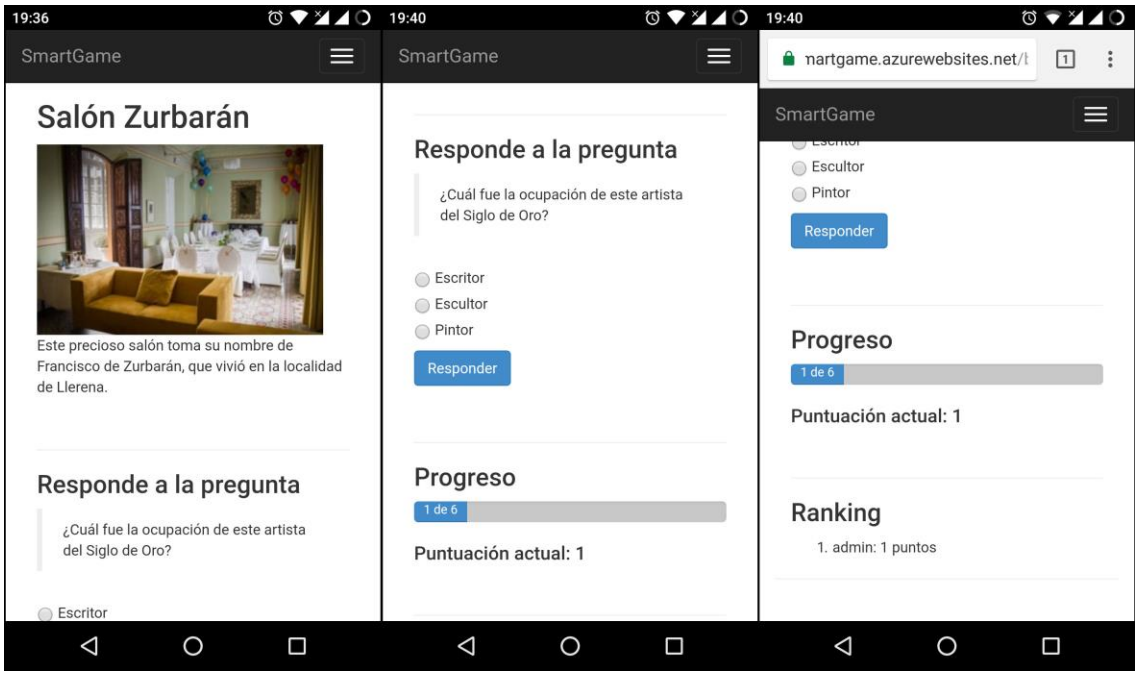

Ilustración 41 - Capturas vista 'baliza'

7.4 Líneas de investigación sobre la implementación piloto

Dado el carácter de prototipo de esta propuesta, resulta interesante buscar evidencia estadística de su acogida por parte de los clientes, y de su contribución a una experiencia positiva de los clientes en el establecimiento.

Para esto, proponemos el envío de una encuesta de satisfacción a todos los clientes tras alojarse en el hotel, incluyendo preguntas como:

- ¿Cuál ha sido el grado de satisfacción global de su estancia?

Para poder buscar correlación estadística entre la participación de los clientes en la actividad y la de su satisfacción, incluiremos en el cuestionario los siguientes campos:

- ¿Usted o sus acompañantes han participado en el juego SmartGame durante su estancia?
- ¿Le ha gustado participar en el juego SmartGame?
- ¿Viaja con niños de más de 10 años?
- En caso afirmativo, ¿cree que sus hijos han disfrutado de la estancia en el hotel?

Con estas preguntas en un cuestionario de satisfacción, podemos averiguar:

- El porcentaje de participación en el juego por parte de los clientes.
- Si la participación en el juego es satisfactoria.
- Si hay correlación entre la participación en el juego y el índice de satisfacción global.
- Si hay diferencias de satisfacción entre los clientes de segmento "familiar" y otros clientes.

Debido a la tipología del establecimiento para el que proponemos el despliegue (un hotel 4* destinado fundamentalmente a cliente de ocio), estimamos positivo recoger estos datos durante un periodo de un año completo antes de extraer conclusiones.

Esto abriría la puerta a otros estudios interesantes, aunque secundarios:

- ¿El juego contribuye especialmente a enriquecer la experiencia de los clientes en caso de estancias prolongadas?
- En un destino de turismo rural, el clima es un factor importante para los clientes, que puede determinar la cantidad de tiempo que los clientes pasan dentro del hotel. Cuando el clima no acompaña (lluvia, temperaturas extremas), ¿hay más participación de los usuarios en el juego? ¿contribuye el juego a mejorar la satisfacción de los clientes en caso de clima adverso, al ofrecerles una alternativa de entretenimiento?

8 Conclusiones

8.1 Logros alcanzados

La propuesta que nos ocupa interconecta varias disciplinas y tecnologías para intentar dar respuesta a la pregunta:

¿Cómo hacer más atractivo para sus visitantes un determinado espacio físico?

Durante el desarrollo de este trabajo hemos logrado:

- Describir el estado de la cuestión de la *gamificación*, así como su aplicación a diferentes ámbitos del conocimiento actual. A día de hoy, la *gamificación* no es un tema tan en boga como a principios y mediados de esta década. Sin embargo, en la introducción teórica comprobamos cómo hay evidencia científica de que contribuye a aumentar la motivación intrínseca del usuario participante en un buen número de actividades.

- Describir la Web Física y su principal tecnología subyacente: Bluetooth Low Energy. La Web Física es un estándar cuya implementación más extendida se basa en el protocolo abierto de balizas Bluetooth Google Eddystone-URL, que también hemos revisado en profundidad y comparado con sus alternativas.

- Idear una actividad de *gamificación* haciendo uso de la Web Física, incluyendo algunos de los elementos claves de una actividad gamificada, como un indicador de progreso o un ranking con los mejores jugadores.

- Diseñar un sistema de balizas bluetooth y aplicación web que implemente la actividad de *gamificación* propuesta.

- Introducir los elementos principales de la implementación de la solución diseñada:

    - Balizas comerciales: Hacemos un recorrido por los modelos comerciales.

    - Placas de desarrollo: Comparamos las distintas opciones para poner en funcionamiento una baliza piloto.

    - Frameworks de programación web: Revisamos *Django*, una librería de programación web para el backend basada en Python. También describimos y utilizamos *Bootstrap*, una librería frontend de programación web especializada en diseños adaptables.

    - Alojamiento web: Realizamos una aproximación a Microsoft Azure, el conjunto de servicios *cloud* de Microsoft.

- Implementar y publicar una plataforma software para la gamificación de espacios físicos, abriendo la puerta a implementaciones piloto.

- Proponer una implementación piloto del sistema, junto con varios experimentos para comprobar si la gamificación de dicho despliegue piloto aporta un valor a los usuarios del servicio a *gamificar*.

8.2 Relación con asignaturas cursadas en el Máster

Durante el desarrollo de este Máster, he tenido la oportunidad de cursar las siguientes asignaturas:

- Computación Ubicua
- Modelado y Simulación de Robots
- Sistemas de Percepción Visual
- Sistemas Difusos de Apoyo a la Toma de Decisiones
- Desarrollo de Software Seguro

A título personal, uno de mis objetivos en esta asignatura era el de crear un desarrollo que pudiese solucionar a corto plazo problemas que no estuviesen limitados al ámbito académico e investigador, a pesar de requerir una labor de investigación e interconexión de áreas del conocimiento para su diseño y ejecución.

El presente trabajo está especialmente relacionado con los contenidos de la Asignatura de Computación Ubicua, área de trabajo que considero apasionante por su capacidad de transformación social.

El trabajo supone un paso más en la digitalización de los espacios, elementos e interacciones de nuestro contexto cotidiano, de ahí su relación con los postulados de la Computación Ubicua.

También he aplicado conocimientos adquiridos en la asignatura de Desarrollo de Software Seguro, que podemos concretar en el estudio de los requisitos no funcionales, manejo de excepciones y técnicas específicas de desarrollo web, aplicadas en parte gracias a funcionalidades ya integradas en el framework de desarrollo elegido. Sin perjuicio de las demás, considero que los contenidos de esta asignatura deberían ser imprescindibles para cualquier estudiante de programación y disciplinas relacionadas.

Por último, comentar que el trabajo también tiene una fuerte relación con la asignatura de Psicología de la Motivación del Grado en Psicología, que tuve oportunidad de estudiar durante el curso 2011-2012 también en esta Universidad.

8.3 Líneas de trabajo futuro

El presente trabajo toma la senda de la investigación aplicada para unir tecnologías y técnicas, algunas de ellas incipientes, y proponer una solución al problema propuesto.

No obstante, hay algunos **retos** que la solución tendrá que enfrentar a futuro:

- ¿La Web Física seguirá teniendo el apoyo de empresas como Google? En caso contrario, ¿qué nuevas tecnologías vendrían a cubrir ese espacio? ¿podrían coexistir o sería mejor adaptarse a los protocolos y estándares por venir?

- Si la Web Física tiene éxito, es posible que los sistemas de filtrado sean más agresivos para proteger al usuario de contenido peligroso o irrelevante. ¿Cómo nos aseguramos el mantener la visibilidad a través de estos filtros?

- A medida que la *gamificación* se va integrando en más ámbitos de la vida cotidiana, ¿nuestra solución seguirá representando un valor añadido en los establecimientos que la implementen?

Más allá de estos retos futuros, hay **mejoras** que podemos implementar en nuestra aplicación de manera más inmediata, como pueden ser:

- **Cifrado ID**: Es posible cifrar los identificadores de las balizas, de manera que las URLs del juego serían menos previsibles y, por tanto, sería más complicado superar las distintas pruebas sin haber visitado realmente las balizas.

- **Más juegos**: el sistema está ideado de manera que es relativamente sencillo cambiar las pruebas actuales (una pregunta que hay que acertar) por otro tipo de pruebas, como pueden ser pequeños juegos de navegador implementados utilizando javascript. Otra posibilidad es cambiar totalmente la dinámica de juego, por ejemplo, exigiendo al usuario que visite las balizas por orden, con un límite de tiempo, etc.

- **Fomentar la involucración y competitividad**: En la introducción teórica vimos algunas técnicas que podrían incrementar el éxito de la aplicación, y que no han podido implementarse en esta prueba por falta de tiempo. Algunas de estas nuevas funcionalidades podrían ser:
    - Inclusión de avatares de usuario
    - Creación de equipos de jugadores
    - Compartir el progreso y el resultado final en redes sociales
    - Mejorar el ranking para que las puntuaciones máximas sean más dinámicas (por ejemplo, listando sólo las mayores puntuaciones del último mes)

- **Mejoras en la administración:** Nuestro desarrollo cuenta con una interfaz de administración suficiente para poner en marcha la prueba piloto. No obstante, sería interesante incluir las siguientes funcionalidades:
    - Configurar automáticamente las balizas Bluetooth: Utilizando balizas basadas en nuestro prototipo piloto con la placa RaspberryPI, sería posible desarrollar un software específico que conectase con nuestro gestor de contenidos a través de internet en un proceso de configuración inicial. En este proceso, la baliza comunicaría su dirección MAC a la aplicación, y ésta le devolvería la URL que tiene que emitir.
    - Generar automáticamente los carteles para cada una de las balizas a través del gestor de contenidos. Django ofrece la posibilidad de generar contenido no HTML.
    - Sistema de permisos granular: Actualmente, el administrador puede gestionar todas las sedes creadas. Podríamos extender el sistema de permisos para tener un mayor control de lo que puede hacer cada usuario.

- **Funcionalidades adicionales de marketing:** En el proceso de registro, recogemos la dirección de correo electrónico del usuario. Sería posible cruzar estas direcciones con los datos de participación del juego para ofrecer mensajes personalizados a los participantes. También sería posible mostrar contenido adicional en las URL de juego, e incluso contenido diferente para los usuarios que completen la actividad.

Por último, un despliegue como el que nos ocupa puede abrir las puertas a profundizar en la **investigación** sobre la *gamificación*, trabajando con variables como:

- Tipo de espacio *gamificado*.
- Tipo de actividad que desarrollan habitualmente los usuarios de dicho espacio.
- Diferentes mecánicas de juegos.
- Actividades individuales o grupales, colaborativas o competitivas.

Todo el conocimiento que pueda generarse en base a estos estudios abrirá las puertas a implementar mejoras en servicios de educación, entretenimiento, cultura, turismo, servicios públicos y multitud de sectores que pueden ser más agradables para las personas gracias a la aplicación de la tecnología.

9 Referencias bibliográficas

[1] M. Weiser, «Hot topics-ubiquitous computing», *Computer*, vol. 26, n.º 10, pp. 71-72, oct. 1993.

[2] «The Physical Web». [En línea]. Disponible en: https://google.github.io/physical-web/. [Accedido: 28-dic-2016].

[3] «UVIC-CGS-GAMIFICATION-2S2012-13.pdf». .

[4] S. Deterding, D. Dixon, R. Khaled, y L. Nacke, «From Game Design Elements to Gamefulness: Defining "Gamification"», en *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, New York, NY, USA, 2011, pp. 9–15.

[5] «Classifying Serious Games: the G/P/S model (PDF Download Available)». [En línea]. Disponible en: https://www.researchgate.net/publication/266462473_Classifying_Serious_Games_the_GPS_model. [Accedido: 29-dic-2016].

[6] B. Reeves y J. L. Read, *Total engagement: How games and virtual worlds are changing the way people work and businesses compete*. Harvard Business Press, 2013.

[7] E. H. Calvillo-Gámez, P. Cairns, y A. L. Cox, «Assessing the core elements of the gaming experience», en *Game User Experience Evaluation*, Springer, 2015, pp. 37–62.

[8] James Neill, «Introduction to motivation and emotion», 07:51:29 UTC.

[9] J. B. Rotter, «Generalized expectancies for internal versus external control of reinforcement», *Psychol. Monogr. Gen. Appl.*, vol. 80, n.º 1, pp. 1-28, 1966.

[10] R. M. Ryan y E. L. Deci, «Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions», *Contemp. Educ. Psychol.*, vol. 25, n.º 1, pp. 54-67, ene. 2000.

[11] M. R. Lepper, D. Greene, y R. E. Nisbett, «Undermining children's intrinsic interest with extrinsic reward: A test of the "overjustification" hypothesis», *J. Pers. Soc. Psychol.*, vol. 28, n.º 1, pp. 129-137, oct. 1973.

[12] F. Groh, «Gamification: State of the art definition and utilization», *Inst. Media Inform. Ulm Univ.*, vol. 39, 2012.

[13] R. M. Ryan y E. L. Deci, «Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being», *Am. Psychol.*, vol. 55, n.º 1, pp. 68-78, ene. 2000.

[14] M. Csikszentmihalyi, *Beyond boredom and anxiety*, vol. xxx. San Francisco, CA, US: Jossey-Bass, 2000.

[15] V. Marín, «Can Gamification Be Introduced within Primary Classes?», *Digit. Educ. Rev.*, jun. 2015.

[16] L. da Rocha Seixas, A. S. Gomes, y I. J. de Melo Filho, «Effectiveness of gamification in the engagement of students», *Comput. Hum. Behav.*, vol. 58, pp. 48-63, may 2016.

[17] L. Sera y E. Wheeler, «Game on: The gamification of the pharmacy classroom», *Curr. Pharm. Teach. Learn.*, vol. 9, n.º 1, pp. 155-159, ene. 2017.

[18] K. E. Rouse, «Gamification in Science Education: The Relationship of Educational Games to Motivation and Achievement», ProQuest LLC, 2013.

[19] J. F. Figueroa, «Using Gamification to Enhance Second Language Learning», *Digit. Educ. Rev.*, jun. 2015.

[20] W. Hsin-Yuan Huang y D. Soman, *A practitioner's guide to gamification of education. Toronto, ON, Canada: Rotman school of management; 2013*. .

[21] M. Urh, G. Vukovic, E. Jereb, y R. Pintar, «The Model for Introduction of Gamification into E-learning in Higher Education», *Procedia - Soc. Behav. Sci.*, vol. 197, pp. 388-397, jul. 2015.

[22] M. Qian y K. R. Clark, «Game-based Learning and 21st century skills: A review of recent research», *Comput. Hum. Behav.*, vol. 63, pp. 50-58, oct. 2016.
[23] T. Alahäivälä y H. Oinas-Kukkonen, «Understanding persuasion contexts in health gamification: A systematic analysis of gamified health behavior change support systems literature», *Int. J. Med. Inf.*, vol. 96, pp. 62-70, dic. 2016.
[24] D. Johnson, S. Deterding, K.-A. Kuhn, A. Staneva, S. Stoyanov, y L. Hides, «Gamification for health and wellbeing: A systematic review of the literature», *Internet Interv.*, vol. 6, pp. 89-106, nov. 2016.
[25] P. Barratt, «Healthy competition: A qualitative study investigating persuasive technologies and the gamification of cycling», *Health Place*.
[26] A. Miloff, A. Marklund, y P. Carlbring, «The challenger app for social anxiety disorder: New advances in mobile psychological treatment», *Internet Interv.*, vol. 2, n.º 4, pp. 382-391, nov. 2015.
[27] O. Korn y A. Schmidt, «Gamification of Business Processes: Re-designing Work in Production and Service Industry», *Procedia Manuf.*, vol. 3, pp. 3424-3431, ene. 2015.
[28] M. T. Cardador, G. B. Northcraft, y J. Whicker, «A theory of work gamification: Something old, something new, something borrowed, something cool?», *Hum. Resour. Manag. Rev.*
[29] C. F. Hofacker, K. de Ruyter, N. H. Lurie, P. Manchanda, y J. Donaldson, «Gamification and Mobile Marketing Effectiveness», *J. Interact. Mark.*, vol. 34, pp. 25-36, may 2016.
[30] J. Schell, *The Art of Game Design: A Book of Lenses, Second Edition*. CRC Press, 2014.
[31] L. F. Rodrigues, A. Oliveira, y C. J. Costa, «Playing seriously – How gamification and social cues influence bank customers to use gamified e-business applications», *Comput. Hum. Behav.*, vol. 63, pp. 392-407, oct. 2016.
[32] C. Kiourt, A. Koutsoudis, y G. Pavlidis, «DynaMus: A fully dynamic 3D virtual museum framework», *J. Cult. Herit.*, vol. 22, pp. 984-991, nov. 2016.
[33] «A realistic Gamification attempt for the Ancient Agora of Athens - IEEE Xplore Document». [En línea]. Disponible en: http://ieeexplore.ieee.org.ezproxy.uned.es/document/7413907/. [Accedido: 05-ene-2017].
[34] R. Confalonieri *et al.*, «Engineering multiuser museum interactives for shared cultural experiences», *Eng. Appl. Artif. Intell.*, vol. 46, Part A, pp. 180-195, nov. 2015.
[35] R. Frasca, A. Mazzeo, D. Pantile, M. Ventrella, y G. Verreschi, «Innovative systems for the enjoyment of pictorial works the experience of Gallerie dell'Accademia Museum in Venice», en *2015 Digital Heritage*, 2015, vol. 1, pp. 349-352.
[36] «Bluetooth Core Specification | Bluetooth Technology Website». [En línea]. Disponible en: https://www.bluetooth.com/specifications/bluetooth-core-specification. [Accedido: 10-ene-2017].
[37] «Introduction | Introduction to Bluetooth Low Energy | Adafruit Learning System». [En línea]. Disponible en: https://learn.adafruit.com/introduction-to-bluetooth-low-energy?view=all. [Accedido: 11-ene-2017].
[38] «Wibree forum merges with Bluetooth SIG», *Nokia*. [En línea]. Disponible en: http://www.nokia.com/en_int/news/releases/2007/06/12/wibree-forum-merges-with-bluetooth-sig. [Accedido: 11-ene-2017].
[39] «Platform Overview | Beacons», *Google Developers*. [En línea]. Disponible en: https://developers.google.com/beacons/overview. [Accedido: 17-ene-2017].
[40] «WorkshopBeacons.pdf». [En línea]. Disponible en: https://www.blueupbeacons.com/docs/WorkshopBeacons.pdf. [Accedido: 17-ene-2017].
[41] B. Cook, G. Buckberry, I. Scowcroft, J. Mitchell, y T. Allen, «Location by scene analysis of wi-fi characteristics», *Relation*, vol. 10, n.º 1.119, p. 6216, 2009.
[42] J. Zhu, K. Zeng, K. H. Kim, y P. Mohapatra, «Improving crowd-sourced Wi-Fi localization systems using Bluetooth beacons», en *2012 9th Annual IEEE Communications Society*

Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), 2012, pp. 290-298.
[43] C. C. Chiu, J. C. Hsu, y J. S. Leu, «Implementation and analysis of Hybrid Wireless Indoor Positioning with iBeacon and Wi-Fi», en *2016 8th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*, 2016, pp. 80-84.
[44] Z. Chen, Q. Zhu, y Y. C. Soh, «Smartphone Inertial Sensor-Based Indoor Localization and Tracking With iBeacon Corrections», *IEEE Trans. Ind. Inform.*, vol. 12, n.º 4, pp. 1540-1549, ago. 2016.
[45] P. M. Varela y T. O. Ohtsuki, «Discovering Co-Located Walking Groups of People Using iBeacon Technology», *IEEE Access*, vol. 4, pp. 6591-6601, 2016.
[46] K. Bouchard, R. Ramezani, Arjun, y A. Naeim, «Evaluation of Bluetooth beacons behavior», en *2016 IEEE 7th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, 2016, pp. 1-3.
[47] A. Akinsiku y D. Jadav, «BeaSmart: A beacon enabled smarter workplace», en *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, 2016, pp. 1269-1272.
[48] Z. Zhao, J. Fang, G. Q. Huang, y M. Zhang, «iBeacon enabled indoor positioning for warehouse management», en *2016 4th International Symposium on Computational and Business Intelligence (ISCBI)*, 2016, pp. 21-26.
[49] G. Saraswat y V. Garg, «Beacon controlled campus surveillance», en *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2016, pp. 2582-2586.
[50] M. Wang y J. Brassil, «Managing large scale, ultra-dense beacon deployments in smart campuses», en *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2015, pp. 606-611.
[51] A. Wakao, K. Matsumura, M. Suzuki, y H. Noma, «Treasure hunt game to persuade visitors to walk around a shopping mall», en *2015 IEEE 4th Global Conference on Consumer Electronics (GCCE)*, 2015, pp. 527-530.
[52] Z. He, B. Cui, W. Zhou, y S. Yokoi, «A proposal of interaction system between visitor and collection in museum hall by iBeacon», en *2015 10th International Conference on Computer Science Education (ICCSE)*, 2015, pp. 427-430.
[53] B. u Lee, S. y Im, S. w Lee, B. Kim, B. h Roh, y Y. B. Ko, «The beacon identification using low pass filter for Physical Web based IoT services», en *2015 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, 2015, pp. 354-358.
[54] M. S. Merkow y L. Raghavan, *Secure and Resilient Software Development*, 1st ed. Boston, MA, USA: Auerbach Publications, 2010.
[55] Yan, «Your AprilBeacon can be either iBeacon or Eddystone now!», 四月兄弟科技. .
[56] «Raspberry Pi 3 Model B», *Raspberry Pi*. .
[57] «BeagleBoard.org - black». [En línea]. Disponible en: https://beagleboard.org/black. [Accedido: 27-mar-2017].
[58] «ODROID | Hardkernel». [En línea]. Disponible en: http://www.hardkernel.com/main/products/prdt_info.php. [Accedido: 27-mar-2017].
[59] «GitHub - don/node-eddystone-beacon: Create an Eddystone Beacon using Node.js https://github.com/google/eddystone». [En línea]. Disponible en: https://github.com/don/node-eddystone-beacon. [Accedido: 22-feb-2017].
[60] «GitHub - sandeepmistry/bleno: A Node.js module for implementing BLE (Bluetooth Low Energy) peripherals». [En línea]. Disponible en: https://github.com/sandeepmistry/bleno. [Accedido: 22-feb-2017].
[61] «Node.js». [En línea]. Disponible en: https://nodejs.org/es/. [Accedido: 22-feb-2017].
[62] «Explore the Physical Web with Chrome - Android - Chrome Help». [En línea]. Disponible en: https://support.google.com/chrome/answer/6239299?co=GENIE.Platform%3DAndroid&hl=en&oco=0. [Accedido: 14-mar-2017].

[63] «The Web framework for perfectionists with deadlines | Django». [En línea]. Disponible en: https://www.djangoproject.com/. [Accedido: 06-mar-2017].
[64] «Getting started · Bootstrap». [En línea]. Disponible en: http://getbootstrap.com/getting-started/. [Accedido: 15-mar-2017].
[65] cephalin, «Introducción a Web Apps». [En línea]. Disponible en: https://docs.microsoft.com/es-es/azure/app-service-web/app-service-web-overview. [Accedido: 15-mar-2017].
[66] L. Webster, «Python Development Tools», *Visual Studio*, 13-sep-2016. .
[67] A. Holovaty y J. Kaplan-Moss, *The Definitive Guide to Django: Web Development Done Right*. Apress, 2009.
[68] «Django Builder». [En línea]. Disponible en: http://mmcardle.github.io/django_builder/#!/home. [Accedido: 26-abr-2017].
[69] «How to Create User Sign Up View», *Simple is Better Than Complex*, 18-feb-2017. [En línea]. Disponible en: https://simpleisbetterthancomplex.com/tutorial/2017/02/18/how-to-create-user-sign-up-view.html. [Accedido: 17-may-2017].
[70] «Security in Django | Django documentation | Django». [En línea]. Disponible en: https://docs.djangoproject.com/en/1.11/topics/security/. [Accedido: 21-may-2017].
[71] A. E. Abujeta Martín, «Intervención en el patrimonio arquitectónico extremeño: la red de hospederías», http://purl.org/dc/dcmitype/Text, Universidad de Extremadura, 2016.