AFANight Events

Autores:

Agustín Basilio, Fabricio Cruz y Antonio Segura

Tutor: Mario Marugán Cancio

CFGS Desarrollo de Aplicaciones Multiplataforma

Curso 2019-2020

IFP (Innovación en Formación Profesional)

Convocatoria de Presentación: Junio 2020

RESUMEN

En los tiempos que corren actualmente, por la gran utilización de las tecnologías y dispositivos móviles; hay ciertas cosas que no se han realizado correctamente en el ámbito del ocio nocturno. Es difícil saber qué fiestas o eventos se encuentran cerca o algo más retirado de la zona de residencia de una persona. Al no ser que sea a través de publicaciones vía redes sociales, anuncios en televisión...etc.

En el TFG se va a desarrollar una aplicación que permita la reserva de entradas a este tipo de eventos, ya sean espectáculos en pubs o en discotecas. Las aportaciones y funcionalidades más importantes serán la ubicación, que ofrecerá al usuario la posibilidad de ver todos los eventos disponibles en el momento y lugar, y también la reserva de entradas online (vía móvil); también podrán ver los eventos más importantes que habrá en todo el país. Otra novedad será la una funcionalidad que permitirá la incorporación a los dueños de estos establecimientos en modo administrador (web), donde ellos mismos se encargarán de colocar el número de entradas. Por este motivo se ha decidido desarrollar tanto una aplicación móvil como una página web. También se hablará sobre las tecnologías empleadas más adelante.

Tras realizar una búsqueda por internet, se llegó a la conclusión que ninguna app móvil y pagina web ofrecen estas funcionalidades dichas anteriormente.

La realización del TFG nos ha supuesto la motivación de haber puesto en práctica todos nuestros conocimientos vistos en estos dos años de curso y sobre todo a la hora de afrontar y saber solucionar los errores que han ido apareciendo durante el desarrollo del proyecto.

INDICE

Capítulo 1. INTRODUCCIÓN

El tiempo de ocio es fundamental en la vida de las personas, sobre todo en estos tiempos donde cada vez más variedades de locales, discotecas, pubs, conciertos…etc.

Ahora con el auge de la tecnología, las personas siempre llevan en su mano un teléfono móvil porque aparte de las llamadas, también lo utilizan para sus redes sociales, GPS, compras en plataformas de ventas online (tipo Amazon) o incluso realizar reservas de restaurantes y hoteles (ej: El Tenedor y Booking), como también eventos deportivos (Ej: Partido de futbol).

El problema es en el ámbito del ocio nocturno, que muchos locales y establecimientos no tienen página web y solamente anuncian sus espectáculos a través de sus redes sociales, por lo que al final, las personas deben seguir cada uno de estos locales en redes sociales, sin poder centralizar la información y sin enterarse de espectáculos fuera de aquellos lugares ya conocidos.

Por tanto, ante estas dificultades, el objetivo es desarrollar una app móvil y una página web, que permita informarse en un único lugar y poder acceder a los usuarios a reservar las entradas online siempre y cuando previamente se hayan registrado a la aplicación. Tanto la app como la web tendrán diferentes funcionalidades:

Web: Aparecerán los eventos más importantes y actualizados que haya en el país y donde los usuarios ya registrados, podrán reservar sus entradas.

Móvil: Los usuarios podrán consultar los eventos disponibles y filtrarlos en base a su ubicación para que así aparezcan con éxito los eventos disponibles cerca de su lugar de residencia y también en toda la comunidad autónoma. Esto es una de las funcionalidades más importantes de la aplicación.

Si se introduce en cualquier buscador de Internet “aplicaciones para eventos”, al instante nos aparecen numerosos portales web donde puedes hacer reservas o compra de entradas para eventos empresariales, musicales, y de entretenimiento en general.

Tras la búsqueda, se observa que ninguna aplicación ofrece las funcionalidades que incluirán en este TFG. Por lo que con las ideas claras y los conocimientos adquiridos durante estos dos años de curso, se toma la decisión de realizar el proyecto.

Una vez decidido el tema del proyecto se evalúan los diferentes lenguajes y metodologías para poder realizarlo y desarrollarlo, que se abordarán detalladamente en los siguientes capítulos (2 y 4).

1.1 Estructura del documento TFG

En este apartado se describe resumidamente la estructura de los apartados de este TFG.

- **Capítulo 1**. Introducción: En este primer capítulo se realiza una breve introducción al tema, se menciona el entorno y la justificación de la importancia del proyecto realizado.

- **Capítulo 2**. Objetivos TFG: En esta sección se detallan los objetivos con los que se pretenden alcanzar en este TFG.

- **Capítulo 3**. Estado del arte: En este apartado se detallan los resultados de la búsqueda bibliográfica realizada para establecer el marco teórico sobre el cual desarrollar el presente TFG.

- **Capítulo 4**. Método de trabajo: En este apartado se explican las metodologías realizadas durante el proyecto para la consecución de los objetivos del TFG.

- **Capítulo 5**. Resultados: En este capítulo se detalla el desarrollo por fases del contenido del proyecto llevado a cabo para la consecución de los objetivos del ciclo realizado

- **Capítulo 6**. Conclusiones: En esta sección se expondrán las ideas principales a las que se han llegado gracias a la consecución del proyecto.

- **Capítulo 7**. Bibliografía: Apartado donde se muestran las referencias escogidas y la bibliografía complementaria para la realización del TFG.

- **Anexos**: Lista de Acrónimos utilizados.

Capítulo 2. OBJETIVOS TFG

2.1 Objetivo principal

El objetivo principal de este TFG es el desarrollo de una aplicación que permita al usuario poder reservar de manera online su entrada para cualquier tipo de evento relacionado con discotecas o pubs que se celebren cerca de su ubicación (lugar de residencia) y en toda la comunidad. También ofrece permisos de administrador para que los dueños de los establecimientos puedan ocuparse de agregar eventos y modificar fechas, siempre y cuando se encuentre dentro del marco legal establecido.

De este objetivo principal se dividen los objetivos específicos, que se explicaran a continuación.

2.2 Objetivos específicos

Para conseguir el objetivo principal, es vital haber realizado previamente los objetivos específicos.

- **Objetivo 1**: realización de un plan de proyecto en el que se estimen los costes y el tiempo empleado.
- **Objetivo 2**: realización de un diseño sólido de la aplicación como base para su respectivo desarrollo y funcionamiento.
- **Objetivo 3**: creación de una base de datos mediante Firebase, para almacenar con éxito los datos pertenecientes a la aplicación.
- **Objetivo 4**: empleo de diferentes lenguajes de diseño y programación, HTML5, JAVASCRIPT, ANDROID (JAVA), y el uso de frameworks, Express.js (Node.js), Bootstrap.
- **Objetivo 5**: desarrollo de la aplicación comentada previamente, con sus respectivas tecnologías aplicadas.

2.3 Medios Utilizados

En este apartado se especificaran los diferentes medios empleados para la realización del TFG.

2.3.1 Medios Hardware

- **Antonio**: ordenador Portátil Asus F541U, 8GB de Memoria RAM, 1TB de disco duro y procesador Intel Core i7.
- **Fabricio**: ordenador Portátil PS63 Modern 8RC-013ES, 16GB de Memoria RAM, 512GB de disco duro y procesador Intel Core i7.
- **Agustín**: ordenador Portátil MacBook Pro (2017), 16GB PLDDR3 de Memoria RAM, 500GB APPLE SSD de disco duro y procesador Intel Core i7.

2.3.2 Medios Software

- **Node.js (v8.12.0)**: para el desarrollo de la aplicación del lado del servidor.
- **Express.js (v4.17.1)**: framework de Node.js empleado para el desarrollo de la web.
- **Firebase (v5.12.0)**: base de datos NoSQL, alojada en la nube.
- **HTML5**: lenguaje de marcas para el diseño web.
- **CSS3**: estilizar el diseño de la web.
- **Javascript**: lenguaje de programación para la implementación de la aplicación web.
- **Bootstrap (v3.4.1)**: framework para crear interfaces y diseños responsive. Versión 3.4.1 debido a la seguridad que ofrece en cuanto a la estabilidad de arranque y admisión de Internet 9 o alguna versión más antigua.
- **Android Studio (v3.6)**: entorno de desarrollo en Java para la aplicación móvil.
- **Trello**: planificador de tareas para la realización del proyecto.

Capítulo 3. ESTADO DEL ARTE

3.1 Evolución del desarrollo web y móvil

En cuanto a las aplicaciones web en la última década, ha experimentado un gran desarrollo. En la década de los noventa la web no era mucho más que folletos digitales, es decir una colección de páginas estáticas sencillas con documentos de texto, dónde solamente los usuarios podían consultarlas.

Pero con el paso del tiempo, esta evolución dio lugar a la creación y desarrollo de páginas web mucho más complejas con diferentes tipos de contenidos como redactar un email o publicar información restringiendo el acceso a ciertos usuarios, todos ellos provenientes de bases de datos originadas a través de una transformación de la red (internet), lo que permitió la creación de estas aplicaciones web centradas al usuario **[1]**.

Una aplicación web típica está organizada en tres capas o niveles (lo cual se adapta perfectamente al patrón de diseño MVC (**Vea imagen 1.1**)), donde el sistema recogerá datos del usuario, primer nivel (Vista), los enviará al servidor, que ejecutará la lógica del programa donde se cargarán los datos de la base de datos, segundo y tercer nivel (Controlador y Modelo respectivamente) **[2]**. Por lo tanto el MVC se divide en:

Modelo: Es la capa donde se trabaja con los datos, por tanto contendrá mecanismos para acceder a la información y también para actualizar su estado. Los datos los tendremos habitualmente en una base de datos, por lo que en los modelos tendremos todas las funciones que accederán a las tablas y harán los correspondientes selects, updates, inserts, etc. No obstante, cabe mencionar que cuando se trabaja con MCV lo habitual también es utilizar otras librerías como PDO o algún ORM como Doctrine, que nos permiten trabajar con abstracción de bases de datos y persistencia en objetos. Por ello, en vez de usar directamente sentencias SQL, que suelen depender del motor de base de datos con el que se esté trabajando, se utiliza un dialecto de acceso a datos basado en clases y objetos **[4]**.

Vista: Las vistas contienen el código de nuestra aplicación que va a producir la visualización de las interfaces de usuario, o sea, el código que nos permitirá renderizar los estados de nuestra aplicación en HTML. En las vistas nada más tenemos los códigos HTML y PHP que nos permite mostrar la salida. En la vista generalmente trabajamos con los datos, sin embargo, no se realiza un acceso directo a éstos. Las vistas requerirán los datos a los modelos y ellas se generarán la salida, tal como nuestra aplicación requiera **[4]**.

Controlador: Contiene el código necesario para responder a las acciones que se solicitan en la aplicación, como visualizar un elemento, realizar una compra, una búsqueda de información, etc. En realidad es una capa que sirve de enlace entre las vistas y los modelos, respondiendo a los mecanismos que puedan requerirse para implementar las necesidades de nuestra aplicación. Sin embargo, su responsabilidad no es manipular directamente datos, ni mostrar ningún tipo de salida, sino servir de enlace entre los modelos y las vistas para implementar las diversas necesidades del desarrollo **[4]**.

El **flujo de control [5]** que se sigue generalmente en el MVC es el siguiente:

1. El **usuario** interactúa con la interfaz de usuario de alguna forma (por ejemplo, el usuario pulsa un botón, enlace, etc.)

2. El **controlador** recibe (por parte de los objetos de la interfaz-vista) la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos.

3. El **controlador** accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario. Los controladores complejos están a menudo estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.

4. El **controlador** delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se reflejan los cambios en el modelo. El modelo no debe tener conocimiento directo sobre la vista.

5. **La interfaz** de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

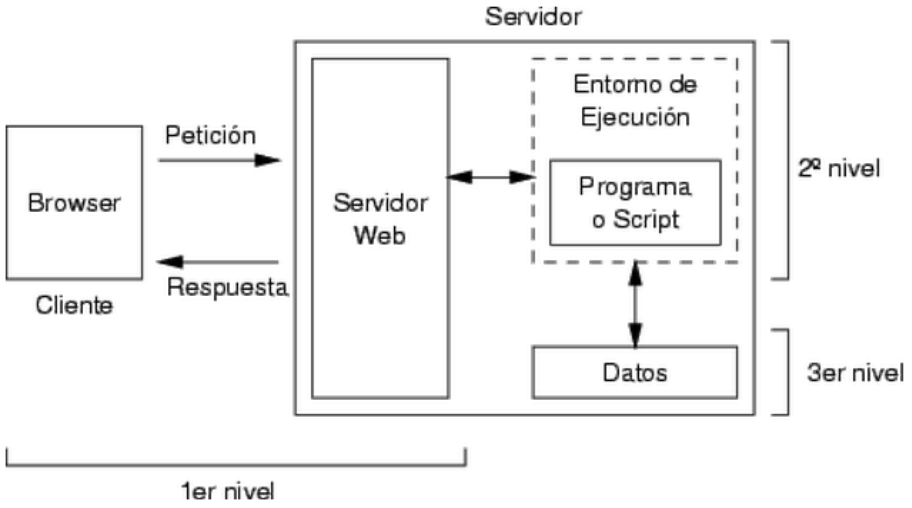


Imagen 1.1

Con respecto a las aplicaciones móviles, se empezaron a desarrollar a finales de la década de los 90, estas aplicaciones eran las que conocemos hoy en día como agenda de contactos, editores de texto, email y algún juego muy simple, como es el caso del mítico juego de "La Serpiente" **[1] [2]**.

La evolución de las apps se dio rápidamente gracias a las innovaciones en tecnología WAP y la transmisión de datos, todo esto vino acompañado de un desarrollo muy fuerte de los teléfonos móviles.

Finalmente la evolución de las aplicaciones junto con el internet, trajo consigo el nacimiento del iPhone de Apple y el desarrollo del sistema operativo para móviles Android. Junto a estos desarrollos llegan muchas más propuestas de desarrollo de Smartphones, y de esta forma empieza el boom de las apps más famosas, tales como WhatsApp, Twitter….y muchas otras más que han ido probando suerte.

De esta manera, la evolución que se espera a partir de ahora es ir hacia una mayor personalización y aumentar el feedback (retroalimentación), al igual que las páginas web, cuyo objetivo es promover una participación activa de los usuarios y una mejor adaptación a sus necesidades de manera más específica.

Al final, todo esto se traduce en una mayor atención al consumidor y una gran funcionalidad. Sin duda una evolución con muchas ventajas **[2] [3]**.

3.2 Comparativa de herramientas utilizadas por similares

En este apartado se detallan las ventajas de porque se han elegido estas herramientas y no otras similares.

- **Node.js:** es un entorno de ejecución para JavaScript **[6]**, que soporta hasta tres veces más peticiones que PHP. Utiliza un solo lenguaje de programación, mientras que en **PHP** se necesita programar en PHP en el back-end y JavaScript en el front-end (por lo que se tendría que aprender dos lenguajes de programación), en cambio con Node.js, tanto en el front-end como en el back-end sólo se necesitará programar en un lenguaje de programación **[7]**.

- **Express.js:** este framework para Node.js es minimalista y permite realizar aplicaciones, sitios web y API. Se ha desarrollado con Connect de modo que sólo se emplean los módulos que necesitas. Mientras que otros como **Compound.js** es un framework desarrollado sobre Node.js que permite crear solamente aplicaciones web de manera más sencilla. Posee una serie de módulos con los que se puede extender su funcionalidad. Funciona tanto en servidor como en navegador web **[8] [9]**.

- **Editor Visual Studio Code:** es gratuito, de código abierto, multiplataforma (Windows, Linux, macOS, ARM y Web), ágil, rápido y súper potente. Además tiene infinitas extensiones para poder ampliar su funcionalidad sin que pierda la ligereza que lo caracteriza, e integra soporte para las herramientas más habituales (desde Git o la línea de comandos integrada, hasta Docker).

  En comparación con otros **editores gratuitos**, **Sublime Text 3**, **Notepad ++**, **Atom**…etc. No solo se puede usar para programar HTML, CSS y JavaScript, sino que tiene un gran soporte (nativo o con extensiones) para infinidad de lenguajes: plataforma .NET y C#, Node.js, PHP, Java, Python, C++..., además de buen soporte para bibliotecas JavaScript como Angular, Vue.js o React y no solamente para la programación **[10]**.

- **Firebase:** es una plataforma para el desarrollo de aplicaciones web y móvil desarrolla por Google. Se encuentra ubicada en la nube Google Cloud Plataform (Cloud Firestore) con su base de datos NoSQL, que usa un conjunto de herramientas para la creación y sincronización de proyectos fáciles o más complejos. Es compatible con grandes plataformas, como IOS, Android, aplicaciones web, Unity y C++. Usa la infraestructura de Google y escala automáticamente para cualquier tipo de aplicación, desde las más pequeñas hasta las más potentes. Crea proyectos sin necesidad de un servidor **[11]**.

  Mientras que otras plataformas como **MongoDB**, que también se utiliza para el desarrollo de aplicaciones web y móvil. Es un sistema de base de datos NoSQL, orientado a documentos y de código abierto. Tiene ciertas limitaciones como problemas de consistencia, bloqueo a nivel de documento (bloquea la base de datos), problemas de rendimiento cuando el volumen de datos supera los 100GB…etc **[12]**.

- **Trello**: se trata de una herramienta de gestión de proyectos muy simple, principalmente para desarrolladores que empiezan a realizar proyectos, tiene un sistema de tarjetas muy intuitivo y sobre todo permite múltiples tableros y miembros en la versión gratuita. Y sobre todo que su interfaz permite utilizarse en español

mientras que otras herramientas como **Backlog**, **Asana**…etc, no lo permiten y además son más complejas **[13]**.

- **Bootstrap:** consta de una librería CSS, una librería JavaScript opcional (para algunos componentes que la requieren), también se le puede añadir un tema para proporcionarle una interfaz más personalizada o llamativa. Tiene una mayor comunidad de usuarios (lo que facilita cualquier tipo de consulta), más plugins desarrollados y una buena documentación **[14]**.

  En comparación con otros como **Foundation**, que es un framework más orientado a la estrategia Mobile First, donde primero se diseña la interfaz para los dispositivos móviles de una web, y a continuación se va adaptando a las versiones más grandes como tablets o escritorio y otros como Semantic UI cuyo desarrollo web es mucho más complejo**[14]**.

- **Android Studio**: es un entorno de desarrollo integrado (IDE) programado sobre todo en **Java**, ya que actualmente es el lenguaje de programación más utilizado y más popular del mundo. Mientras que en **IOS** el único IDE es el **Xcode** y utiliza su propio lenguaje de programación llamado **Swift**.

  Android incluye todas las máquinas virtuales de todas las versiones anteriores para que no se altere el funcionamiento de una app. Es un sistema más abierto, fácil y "libre" para el desarrollo de una app, aparte de que es más fácil, más barato y encima se cubren a más usuarios potenciales, ya que permite realizar casi cualquier tipo de app **[15] [16]**. En Android hay miles de fabricantes con cientos de terminales disponibles. A nivel mundial, Apple tiene una media 15% del mercado global, y Android en torno al 80%.(**Vea imagen 1.4**)

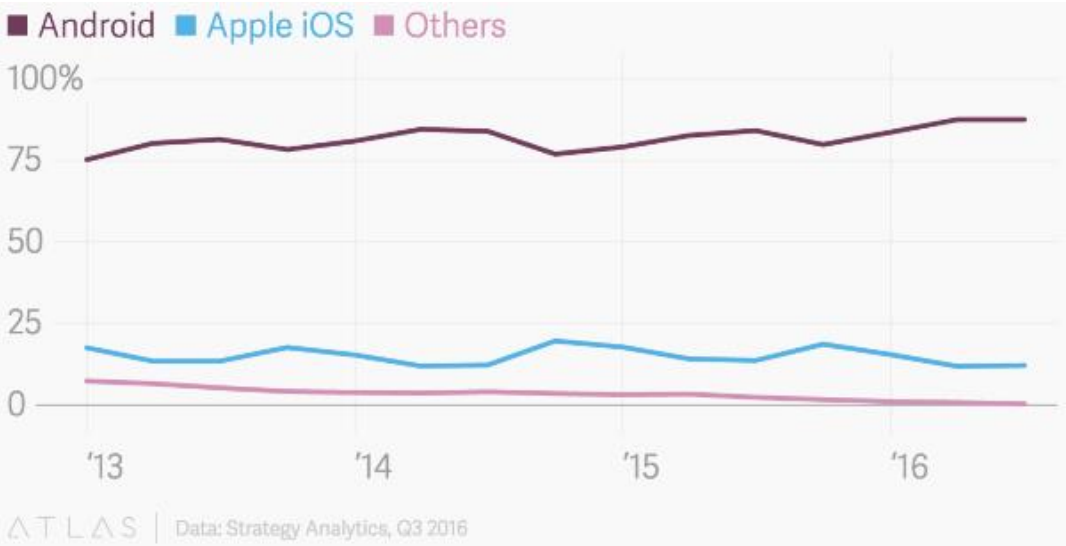


Imagen 1.4

Por todo esto, Android Studio es la herramienta empleada para el apartado de la aplicación móvil del TFG, aparte de que también es el único entorno de desarrollo móvil impartido durante estos dos cursos.

3.3 Aplicaciones similares

A continuación se realiza una comparación y crítica con varios portales de Internet **[17]**, todos ellos realizan funciones de venta de entradas en diferentes secciones y tienen registro de usuarios. Como en este TFG se puesto especial hincapié en la reserva de entradas para discotecas y pubs, y sobre todo en la implementación de la ubicación para saber que eventos de este tipo se encuentra cerca de donde la persona (usuario) se encuentra ubicado.

3.3.1 Feverup.com

Feverup (**Vea imagen 3.1 y 3.2**) es una página web y aplicación móvil que se centra en la venta de entradas en diferentes sectores de ocio, ya sea festivales, eventos de scaperoom, gastronomía, deporte y actividades, escapadas y viajes, cine y teatro o festivales.

- Una de las características principales a diferencia de la aplicación, es que esta no emplea la ubicación. Solamente permite ver los eventos que hay en una ciudad en concreto (Ej: Madrid).

- Otra característica que diferencia la aplicación realizada sobre esta, es que no tiene eventos de discotecas, ni de pubs.

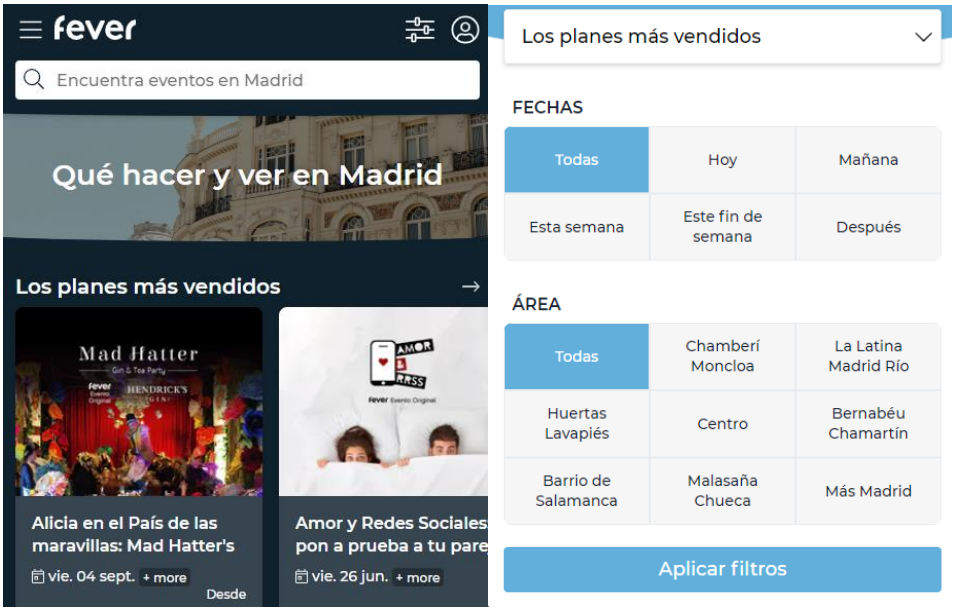


Imagen 3.1 y 3.2 (versión móvil desde la web)

3.3.2 Eventbrite.es

Eventbrite (**Vea imagen 3.3**) es otra página web y a la vez también una aplicación móvil, que ofrece diferentes tipos de eventos al igual que la anterior aplicación.

- ☐ En este caso esta aplicación, se centra sobre todo en eventos de música, fiestas y gastronomía. Al igual que la otra aplicación, esta tampoco está basada en la ubicación y en el apartado de búsqueda no se encuentran eventos en pub o en discotecas.
- ☐ Al igual que la aplicación realizada, esta también permite la creación y configuración de eventos.

Imagen 3.3 (app móvil)

Después de analizar y comparar estas aplicaciones, se ha visto que no ofrece ni contiene las características dichas en capítulos anteriores sobre la aplicación realizada.

Capítulo 4. MÉTODOS DE TRABAJO

A continuación se describen las metodologías a utilizadas para el desarrollo del proyecto.

Se ha decidido utilizar una metodología similar a **Scrum**, adaptando esta al tamaño y características del equipo: Ya que este actúa tanto como equipo de desarrollo y como cliente, además está compuesto por tan solo 3 personas y no se encuentran geográficamente cerca ni se reúnen habitualmente en ningún lugar común.

Teniendo en cuenta estas características del equipo de trabajo, crea un grupo de **WhatsApp [18]** integrado por los miembros del equipo donde se comparten todas las comunicaciones cotidianas del equipo, aparte de la realización de videollamadas.

4.1 Elección del software de gestión

Para la gestión de los diferentes apartados del proyecto y haciendo uso de esta metodología se ha decidido utilizar un software de administración de proyectos.

Se buscaron aplicaciones de gestión que cumplieran los criterios de:

1. Estar disponible online y poder crear, por la facilidad que esto proporciona para acceder desde cualquier equipo, y por la diversidad de herramientas gratuitas que podemos encontrar.

2. Poder crear tareas, una acción básica para poder organizar el trabajo.

3. Poder asignar tareas a usuarios, para tener claro si una tarea está siendo trabajada y por quién.

Se propusieron varias aplicaciones online que parecían satisfacer las necesidades, destacando entre ellos, **Backlog [19]**, **Asana [20]** y **Trello [21]**.

La mayoría de los cuales fueron descartados porque eran de pago u ofrecían un periodo de prueba muy limitado, habiendo alternativas gratuitas. De entre las restantes se eligieron Trello y Backlog y luego de algunas pruebas el equipo se decantó por Trello, ya que reunía características similares a Backlog pero su interfaz permite utilizarse en español.

4.2 Trello

En Trello se estableció un panel con diferentes columnas, donde se listan las tareas a realizar de la manera más anatómica posible. Las tareas no solo consisten en desarrollo de software, sino también otras como la documentación del proyecto (**Ver imagen 4.1 y 4.2**). Estas tareas irán progresando por las distintas columnas de izquierda a derecha hasta llegar a la última. Cuando una tarea llega a la columna más a la derecha significa que la tarea está concluida.

A continuación se listaran las columnas incorporadas al panel y se describirá que significa que una tarea se encuentre en dicha columna, como llego ahí y que hace falta para que se mueva.

El equipo ha intentado ser lo más riguroso posible a la hora de mover las tareas de columnas, pero cabe señalar que no se ha utilizado como una estructura rígida, si después de estar trabajando con una tarea el equipo se encuentra con que la tarea no está suficientemente bien definida o simplemente es incorrecta, se evalúa cada caso y se decide si se desdobla en más tareas, se saca del tablero, o lo que haga falta. La idea detrás de utilizar un tablero con tareas y una progresión de columnas es tener un control general por parte de todo el equipo de que está ocurriendo en el proyecto.

- **Backlog**: son las tareas a realizar durante toda la duración del proyecto. Corresponde a las tareas del **Product Backlog**.

- **Design**: algunas tareas, sobre todo las relacionadas a la capa de presentación, requieren que se realice un trabajo de diseño antes del de programación. Cuando formen parte del **Sprint Backlog** se posicionarán inicialmente en esta columna para diferenciarlas del resto.

- **To Do**: son las tareas planificadas a realizar durante el transcurso del **Sprint**. De aquí el de desarrollador elegirá la tarea que desea realizar y la pasará a la siguiente columna, asignándose la misma.

- **Doing**: son las tareas que están actualmente en la fase de desarrollo, significa que alguno de los desarrolladores ha decidido llevarla a cabo.

- **Code Review**: etapa pensada para cuando una tarea se ha realizado pero existen dudas sobre su resolución y es necesario que otro miembro del equipo revise el código. Si no fuera el caso la tarea pasa directamente al siguiente apartado.

- **Testing**: las tareas en columnas aquí serán revisadas funcionalmente por otro miembro del equipo para corroborar que se halla implementado adecuadamente. Si es positivo, pasa al siguiente estado, en caso contrario y dependiendo de la gravedad de la falla vuelve a alguno de los estados anteriores.

- **Done**: son las tareas desarrolladas que han superado la fase de Testing, estas tareas se encuentran terminadas y no se requiere más trabajo sobre ellas.

4.3 SCRUM (Adaptado)

Scrum es una metodología ágil de autogestión, pero no es en sí misma una metodología de desarrollo puesto que no indica de qué manera se debe trabajar sobre el código.

Existen varios roles y eventos destacables. Entre los roles se encuentran:

- El "**Product Owner**", que es quien está en contacto con el cliente, conoce sus necesidades y por tanto es el encargado de crear la lista de funcionalidades que implementa la aplicación, se conoce a esta lista de funcionalidades como "Product Backlog".

- El "**Development Team**", compuesto por todos los actores encargados de realizar las tareas correspondientes al progreso del desarrollo de la aplicación, este equipo debe auto-organizarse y tiene el control sobre las tareas que se realizan en cada iteración a fin de poder entregar un incremento en la cantidad de funcionalidades listas al final de cada ciclo.

- Y el "**Scrum Master**", que es la persona encargada de promover y soportar el Scrum, maneja las interacciones entre quienes están fuera y dentro del equipo y se encarga de facilitar que todos los actores del equipo puedan realizar su trabajo sin problemas.

Entre los eventos encontramos:

- El **"Sprint"** (**Vea imagen 4.0**), es un intervalo prefijado durante el cual se crea un incremento de producto "Hecho o Terminado" utilizable, potencialmente entregable A lo largo del desarrollo hay Sprints consecutivos de duración constante **[22]**. Cada sprint consta de 5 etapas:

    - El **Sprint Planning**, el trabajo que se ha de realizar durante el Sprint es planificado en esta etapa. Todo el equipo colabora en la elaboración de este plan, formado por aquellos ítems del Product Backlog que se desean realizar en este periodo. Esta lista de ítems seleccionados recibe el nombre de Sprint Backlog.

    - **Daily Scrum**, es un evento de corta duración que se produce cada día del sprint. Aquí el Development Team sincroniza el trabajo a realizar en las próximas 24hs, se revisa el trabajo realizado desde el último Daily Scrum y se ajusta el mismo con vistas al objetivo al final del Sprint. En estas reuniones también se identifican y eliminan obstáculos para el desarrollo.

    - El trabajo de desarrollo en sí, esto es, las tareas que se realizan diariamente en pos del objetivo al final del Sprint.

    - El **Sprint Review**, una reunión que se realiza al final del Sprint donde se presenta el trabajo terminado durante el Sprint. También se evalúa el trabajo que no se pudo terminar y se ajusta el **Product Backlog**.

      El equipo discute qué se realizará a continuación, teniendo en cuenta que en el transcurso del Spring las prioridades del cliente o el mismo **Product Backlog** pueden haber cambiado. Es el momento para discutir qué cosas salieron bien durante el **Spring**, qué problemas surgieron y cómo se solucionaron. El objetivo final del **Sprint Review** es tener un **Product Backlog** revisado que define los potenciales ítems a incluir en el próximo **Sprint**.

    - Y el **Sprint Retrospective**, una reunión que ocurre luego del **Sprint Review** y antes del siguiente **Sprint Planning**. Es una oportunidad para que el equipo analice cómo fue el último Sprint, identifique aquellas cosas que salieron bien y como se pueden mejorar y cree un plan para implementar mejoras en la manera que el equipo realiza su trabajo **[23]**.

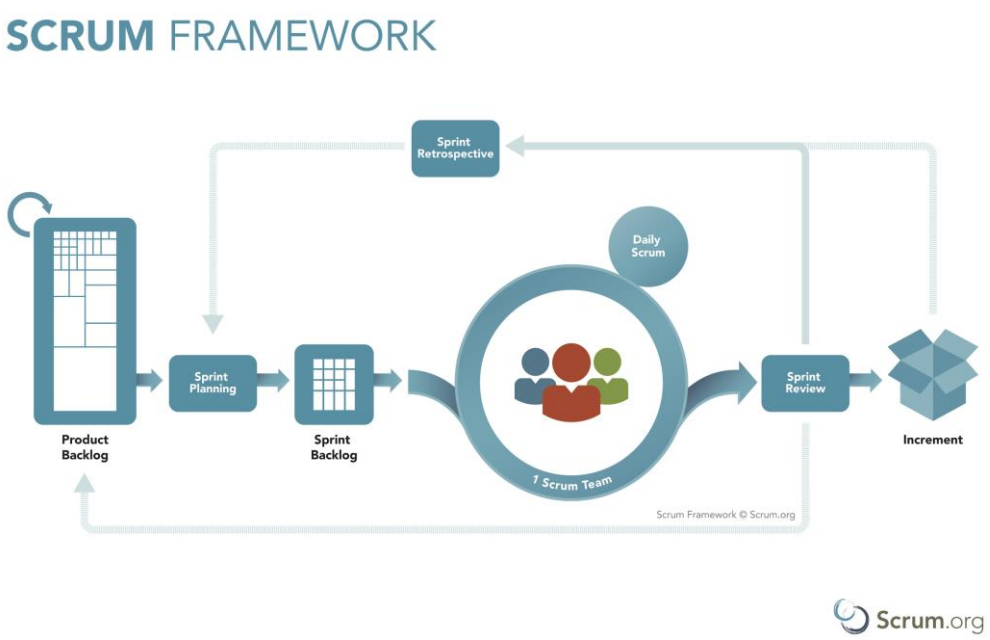


Imagen 4.0

Como ya se ha mencionado, para gestionar el proyecto se ha optado por adaptar SCRUM a las realidades del equipo.

En este caso, los tres miembros del equipo actúan como clientes proponiendo funcionalidades que deben tener las aplicaciones.

- Para el rol **Product Owner** se designa a uno de los integrantes del equipo y es este el que se encarga de cargar las tarjetas con tareas al **Backlog** del proyecto. Las tareas son discutidas y acordadas por todos los miembros del equipo, pero se asigna a una persona este rol para que en el caso de conflictos que no se puedan resolver o cambios de prioridades sea esta persona la responsable de velar por el **Backlog** y sus objetivos generales.

- El rol de **Development Team** es cumplido por todos los miembros del equipo en conjunto.

- El rol de **Scrum Master** es asignado a uno de los miembros del equipo para que sea este el encargado de que todo el equipo realice el **Scrum** adecuadamente.

- Se adaptaron las **Daily Scrums** a un mensaje en el mencionado grupo de WhatsApp al inicio de cada jornada de trabajo, mencionando ¿Qué se realizó en la jornada anterior? ¿Qué se pretende realizar en esta? Y si existe algún impedimento de cualquier tipo (**stopper**).

- Se establecen una lista de tareas, el Sprint **Backlog**, en **Trello** de donde cada desarrollador escoge aquella que le parece más adecuada y la realiza hasta el final.

- Se establece una duración de 4 semanas para cada Sprint.

El trabajo de desarrollo en sí, así como las distintas reuniones al inicio y fin de cada Sprint se realizan por videoconferencia.

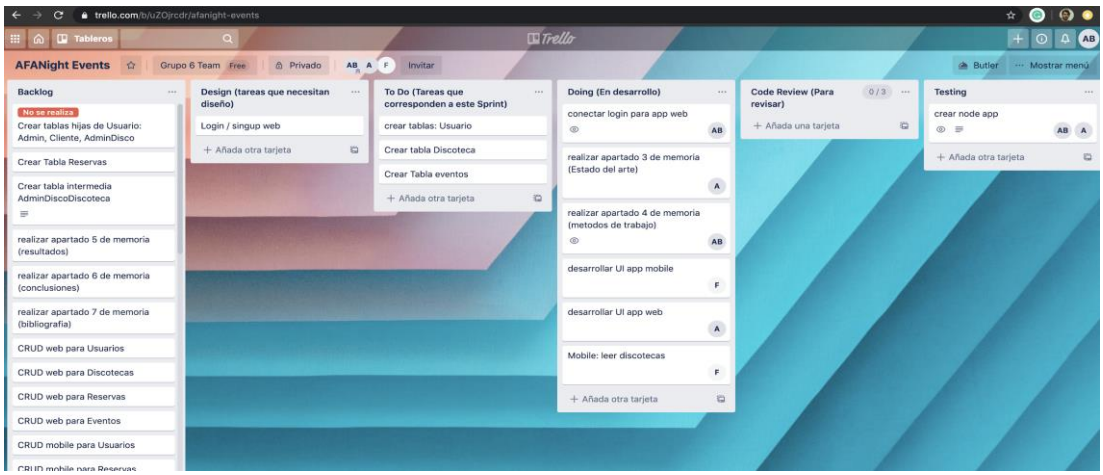


Imagen 4.1

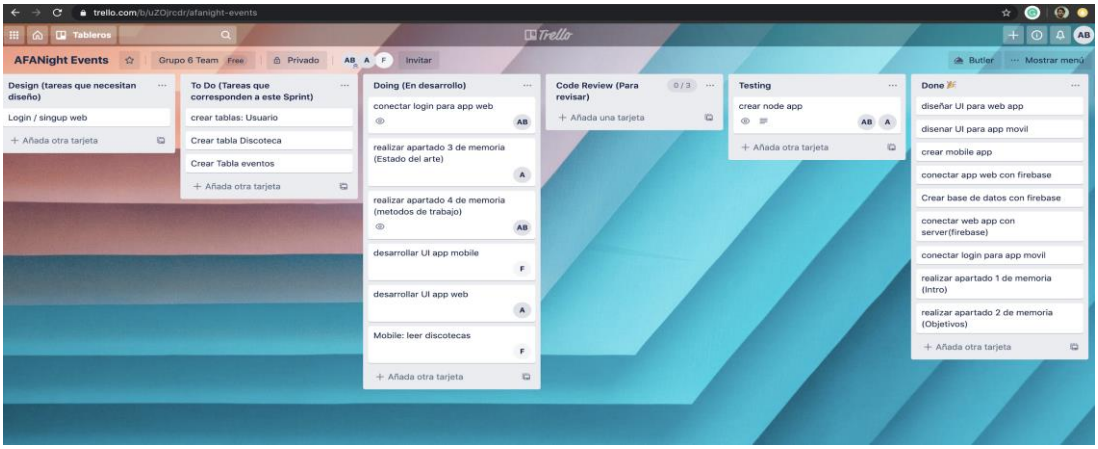


Imagen 4.2

4.4 Modelado

Para la realización de los distintos diagramas se utiliza diagrams.net en su versión online, por ser una aplicación ya utilizada por el equipo con anterioridad y con la cual está familiarizado.

También se utiliza “diagramuml” para extraer algunas de las clases de la aplicación android.

- **Diagrmauml**, es un plugin para Android Studio que permite extraer clases en formato UML.

- **Diagrams.net,** es una aplicación open source, que permite crear y editar una gran variedad de diagramas. Cuenta con una gran variedad de formas iconos, conectores y plantillas, lo que permite una gran agilidad a la hora de crear diagramas. Existe tanto en su versión online como distintas versiones descargables.

4.5 Desarrollo

Para realizar la aplicación web se utilizan:

- **Visual Studio Code** como editor de texto y su consola integrada tanto para para instalar paquetes de **Node.js** como para ejecutar el motor del mismo.
- **Node.js** para el desarrollo de la aplicación del lado del servidor.
- **Express Web Framework** que provee a **Node.js** un conjunto de características para aplicaciones web, en concreto para este TFC se lo utiliza como enrutador para los pedidos generados desde el cliente.
- **EJS** como motor de plantillas, para poder incorporar código **JavaScript** a la generación dinámica de las vistas devueltas al usuario.
- **Ejs-mate** incorpora de manera simple la funcionalidad de vistas parciales a utilizar junto con **EJS**.
- **CSS3** para proporcionar estilos a las vistas del lado del cliente.
- **Firebase Realtime Database**, base de datos NoSQL alojada en la nube que permite almacenar y sincronizar datos en tiempo real, como base de datos para almacenar la información.
- **HTML5**, lenguaje de marcas de hipertexto, para las vistas que se presentan al usuario.

Para la aplicación móvil se utiliza:

- **Android Studio**, el entorno de desarrollo integrado oficial para el desarrollo de apps para Android, como IDE para el desarrollo de la aplicación.
- **Java** como lenguaje de programación.
- **Firebase Realtime Database**, como base de datos para almacenar la información.

Capítulo 5. RESULTADOS

En esta sección se va a detallar las fases que se han seguido para la aplicación propuesta de este TFG. Se tratará de detallar la finalidad de cada fase y el resultado obtenido al final de la misma.

5.1 Fase 1-Comienzo

En esta **fase** se lleva a cabo la búsqueda de aplicaciones existentes o no, con el objetivo de tener una idea para realizar la aplicación, y también información acerca de las tecnologías a utilizar para la elaboración del **Anteproyecto**. También se llevan a cabo las primeras entrevistas con el cliente, Mario Marugán Cancio (Tutor del TFG), para coger ideas.

Una vez realizado estas **primeras tareas**, se procede a realizar un estudio de viabilidad del proyecto para saber si el trabajo a realizar resultará exitoso o se tienen que plantear soluciones alternativas. Además se realizará la primera planificación del proyecto donde se estimarán los costes en cuanto a tiempo y presupuesto.

5.1.1 Primeras entrevistas con el cliente

En **Marzo de 2020** se tiene una primera toma de contacto con el cliente para tratar la idea de la aplicación a desarrollar. Se ponen en común las ideas acerca del alcance que se debe tener de la aplicación y se entrega el Anteproyecto para que el cliente (tutor) dé el visto bueno.

Tras la aprobación del Anteproyecto, se comienza a desarrollar el TFG y a realizar las primeras versiones de un Diagrama E/R, que fue aceptado por el cliente, aunque se tenían que realizar bastantes mejoras sobre ello, para ofrecer un mayor entendimiento al cliente (tutor).

A continuación se exponen los diagramas dichos anteriormente, vea imagen 5.1.

Según se iba avanzando en el desarrollo de la aplicación, se han ido realizando ciertas modificaciones en los diagramas, que se explicará con más detalles en el plan de proyecto.

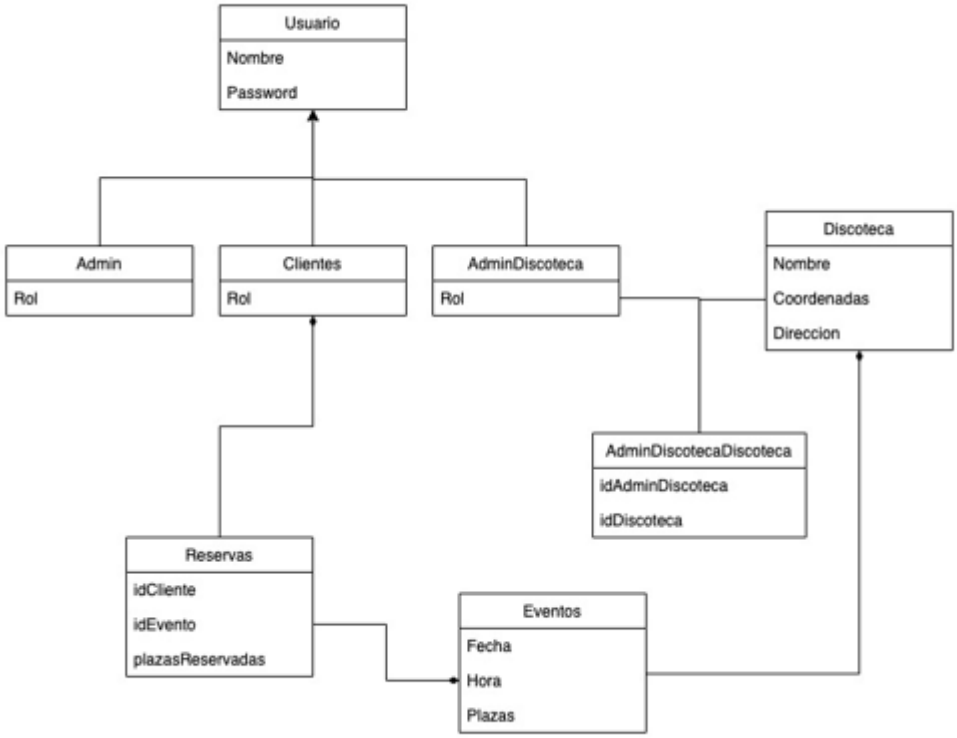


DIAGRAMA E/R ANTIGUO (Imagen 5.1)

5.1.2 Ámbito en el que se desarrolla la aplicación

La aplicación a desarrollar se llama AFANight Events, y se dividirá en AFANight Events (WEB) y AFANightApp (MÓVIL) donde cada una realizará las siguientes tareas:

AFANight Events (WEB):

- La principal característica será que los usuarios están obligados a registrarse a la aplicación si quieren poder reservar entradas y poder navegar por ella para ver otros eventos que están por venir.
- Los usuarios podrán ver su perfil y también editarlo.
- Los usuarios, tras realizar la reserva del evento podrán ver la reserva que han realizado previamente.

AFANightApp (MÓVIL):

- Los usuarios pueden editar su perfil, así como añadir una foto.
- Tienen la posibilidad de poder ver donde se encuentra el evento a donde van ir.
- Tienen también un apartado llamado “favoritos” donde aparecen los eventos seleccionados, incluso antes de hacer la reserva.

5.1.3 Visión general de la aplicación

Esta aplicación aporta un nuevo concepto en el ocio nocturno y también en cuanto a aplicaciones se refiere, que existen hoy en día. Esta aplicación tiene la posibilidad de realizar reservas y ver eventos próximos, como también saber dónde están ubicados los eventos en los cuales el usuario realiza la reserva (esto solamente se realiza por la aplicación móvil). Como también en permitir que los dueños (clientes) de los locales donde se celebre el evento puedan registrarse en modo administrador para poder publicar su respectivo evento y controlar el número de entradas. El objetivo es acercar de una manera distinta, a los usuarios y los dueños de los locales. Toda esta aplicación tiene la obligación de adaptarse a las normas estipuladas y requisitos, con el objetivo de ofrecer seguridad y adaptarse a las necesidades del usuario y el cliente (dueño del local).

5.1.4 Descripción general

La aplicación a desarrollar no necesitará de otros sistemas para su funcionamiento, simplemente la interacción entre la persona (usuario) que la utilice y el propio sistema de la aplicación. Para realizar dicha interacción, se va a diferenciar distintas interfaces según el tipo de rol que tenga el usuario en el sistema:

- **Interfaz de Usuario** (tanto para aplicación web como para la aplicación móvil).
- **Interfaz Usuario-Administrador** (solamente en la aplicación web).

5.1.5 Características de los usuarios

Usuario Web: Puede hacer reservas de diferentes eventos, ver otros próximos y configurar su perfil.

Usuario Móvil: Al igual que la web puede hacer las reservas de los eventos, también ver dónde se encuentran localizados esos eventos como guardarlos en el apartado de “favoritos”.

Usuario-Administrador (Sólo Web): Puede publicar su evento y estar pendiente de las reservas que han hecho los usuarios.

5.1.6 Restricciones

El sistema debe cumplir con la ley de protección de datos. Nunca podrán ser revelados los datos proporcionados de los **Usuarios** y los **Usuarios-Administrador**.

Para desarrollar la aplicación web se va a emplear **JavaScript** como lenguaje de programación bajo el framework **Node.JS** y **Express.JS**. Para la base de datos se va utilizar **Firebase**. Para las vistas y el diseño se utilizará **HTML5**, **CSS3** y **Bootstrap**.

Para conseguir un sistema seguro y de confianza, se debe proporcionar datos de acceso, es decir, usuario y contraseña tanto a los Usuarios como los Usuarios-Administradores.

El objetivo de la aplicación, es que contenga un sistema **intuitivo y cómodo** para los usuarios.

5.1.7 Estudio de viabilidad

Con este estudio se pretende determinar el éxito o el fracaso del proyecto a presentar, con el fin de decidir la viabilidad de la idea presentada y con la aspiración de ponerlo en marcha.

Para el desarrollo de este estudio se tendrán en cuenta las siguientes áreas:

- **Viabilidad económica:** Estudio de los costes del desarrollo.
- **Viabilidad técnica:** Estudio de los conocimientos y aptitudes adquiridas durante los dos años de curso del grado superior.
- **Viabilidad legal:** Determinar cualquier posibilidad de infracción o responsabilidad legal.

5.1.7.1 Viabilidad económica

Para realizar dicho estudio, se toma la decisión de separar los gastos, en gastos directos e indirectos, con el objetivo de evaluar la posibilidad de realizar este TFG.

Costes directos: Son aquellos gastos fijos que habrá que embolsar como inversión inicial o los costes fijos que se tendrán todos los meses (Vea tabla 5.1).

Gastos Fijos	**Coste**
Sueldo Desarrollador de Aplicaciones Multiplataforma	150€

Tabla 5.1

Al haber tres desarrolladores, se multiplica ese sueldo por 3, por lo que cada mes se estima un gasto de 450€.

Costes Indirectos: Son aquellos gastos que no están relacionados directamente con las actividades o resultados, sino con el conjunto de ellos (Vea tabla 5.2).

Gastos Indirectos	Coste
Luz	30€
Calefacción	45€
Internet	35€
Diferente material de oficina (bolígrafos, subrayadores, cartuchos impresora)	15€

Tabla 5.2

Costes totales: El TFG tiene una duración de 2 meses y medio por lo que el coste por el sueldo del desarrollador será 450€*2 meses y medio= 1.125€ totales. En cuanto al gasto indirecto total será 125 al mes, y si lo multiplicamos por los 2 meses y medio para realizar el proyecto sería de: 125€*2 meses y medio= 312,5. Más luego multiplicado por los tres desarrolladores el total de los costes indirectos será de unos 937,50€. Como se tiene en cuenta que los desarrolladores no percibirán ningún sueldo ya que lo hacen por cuenta propia, y que también los gastos indirectos se asumen por cuenta propia.

En conclusión, el proyecto solo tendrá los gastos indirectos, cuya responsabilidad será por parte del alumno, por lo que no afectará a la realización del proyecto.

5.1.7.2 Viabilidad técnica

El análisis técnico consiste en realizar una evaluación del software y personas disponibles y si tienen las capacidades técnicas requeridas.

Para la realización de este Trabajo Fin de Grado se cuentan con las siguientes herramientas de trabajo:

- **Antonio**:
  - Ordenador Portátil Asus F541U.
  - Navegador Google Chrome.
  - Visual Studio Code 2020.
  - Microsoft Windows 10.

- Microsoft Office 2013.
- JavaScript.
- Framework Node.JS.
- Framework Express.JS.
- Librería Muuri.
- Framework Boostrap.
- HML5 y CSS3.
- Android Studio 2020.
- Adobe Acrobat Reader (PDF)

Fabricio:

- Ordenador Portátil PS63 Modern 8RC-013ES
- Navegador Google Chrome.
- Visual Studio Code 2020.
- Microsoft Windows 10.
- Microsoft Office 2016.
- JavaScript.
- Framework Node.JS.
- Framework Express.JS.
- Librería Muuri.
- Android Studio 2020.
- Adobe Acrobat Reader (PDF)

Agustín:

- Ordenador Portátil MacBook Pro (2017)
- Navegador Google Chrome.
- Visual Studio Code 2020.
- Microsoft Windows 10.
- JavaScript.

- Framework Node.JS.
- Framework Express.JS.
- Framework Boostrap.
- HML5 y CSS3.
- Android Studio 2020.
- Adobe Acrobat Reader (PDF)

Como se puede ver, la mayoría del software es **gratuito**, es decir **Open Source**, con lo cual se puede decir que no existe ningún problema a la hora de desarrollar el TFG.

En cuanto a las capacidades técnicas, el equipo ha adquirido el conocimiento suficiente durante estos dos años de grado, para desarrollar el TFG.

Tras ver que los dos análisis son **favorables**, se puede asegurar que el estudio de viabilidad técnica es favorable para la realización del TFG.

5.1.7.3 Viabilidad legal

En este análisis, se asegura que el proyecto no infringe ninguna norma o ley establecida. Además se debe garantizar el respeto a los acuerdos relacionados con el ámbito del proyecto. También se deben tener en cuenta que las licencias para el software empleado debe ser auténticas para evitar inconvenientes legales futuros. Y es por estas razones por las que se ha decidido poner atención en la **Ley Orgánica 15/1999 de Protección de Datos de Carácter Personal [24]**.

Puesto que se almacenarán los datos personales de todos los usuarios registrados en la aplicación (**web y móvil**), se aplicará un nivel de seguridad básico de la **LOPD**, ya que la aplicación de la **LOPD** con carácter general es de aplicación a todo tratamiento de datos de carácter personal **[24]**.

A la hora de dar de alta un nuevo usuario, se recogerán los datos a un mayor de edad por lo que cumple todos los requisitos del **artículo 5 de la LOPD**. En todo momento se respetará el derecho de acceso, rectificación, cancelación u oposición **(ARCO)** garantizado que el usuario dado de alta es exclusivamente el interesado **[24]**.

5.1.8 Plan de proyecto

Una vez efectuado el Estudio de viabilidad y ver que es posible realizar el proyecto, se desarrolla una planificación del proyecto inicial. Dicha planificación se hace a través de la herramienta Trello, explicada anteriormente en el capítulo 4, como también de la

herramienta Diagrams.net y DiagramUML, para la realización de los diagramas, explicado en el capítulo 4.

Los diagramas son los siguientes (Ver imágenes 5.1, 5.2, 5.3, 5.4, 5.5 y 5.6):

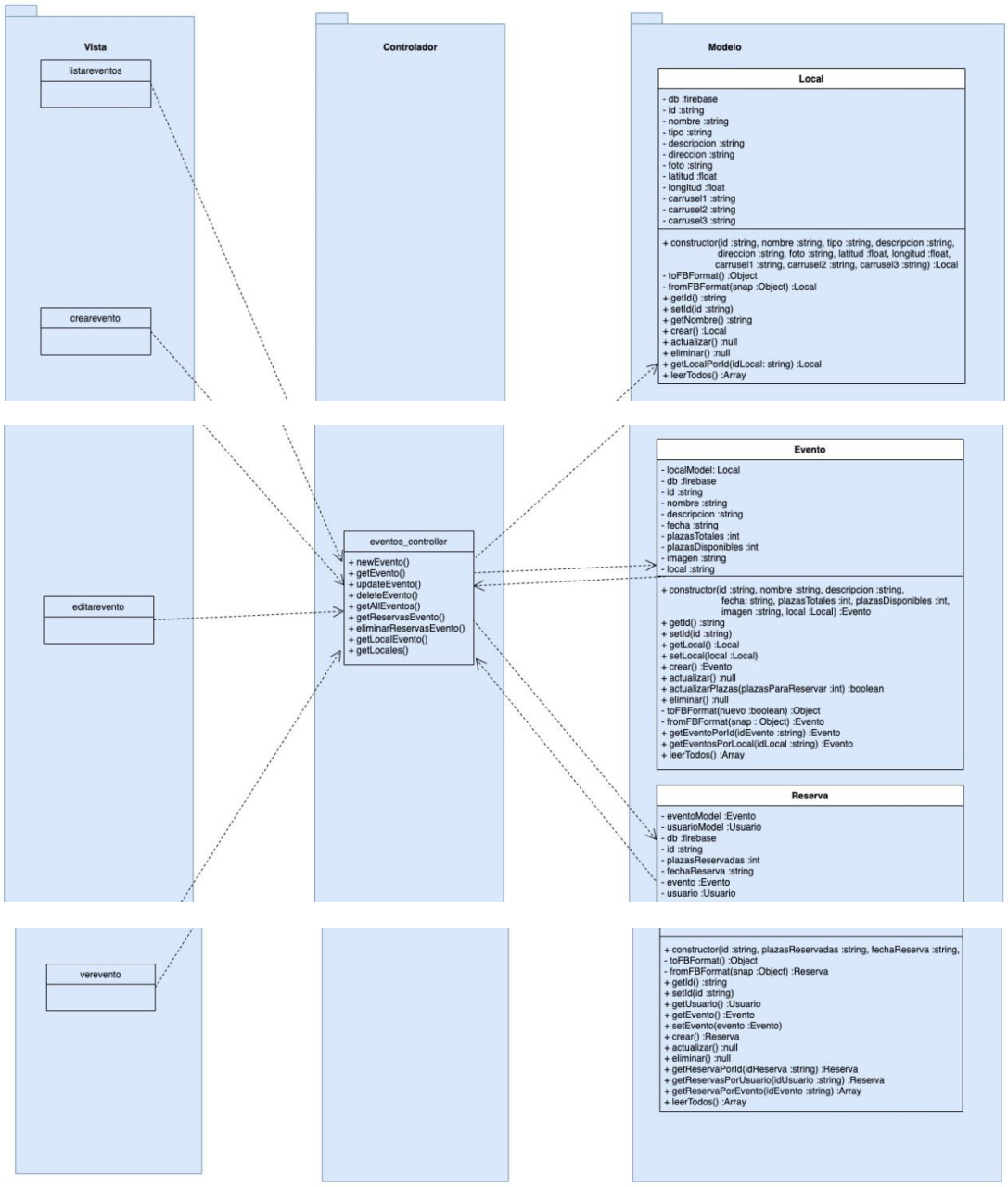


Imagen 5.1-DIAGRAMA DE CLASES (EVENTOS)

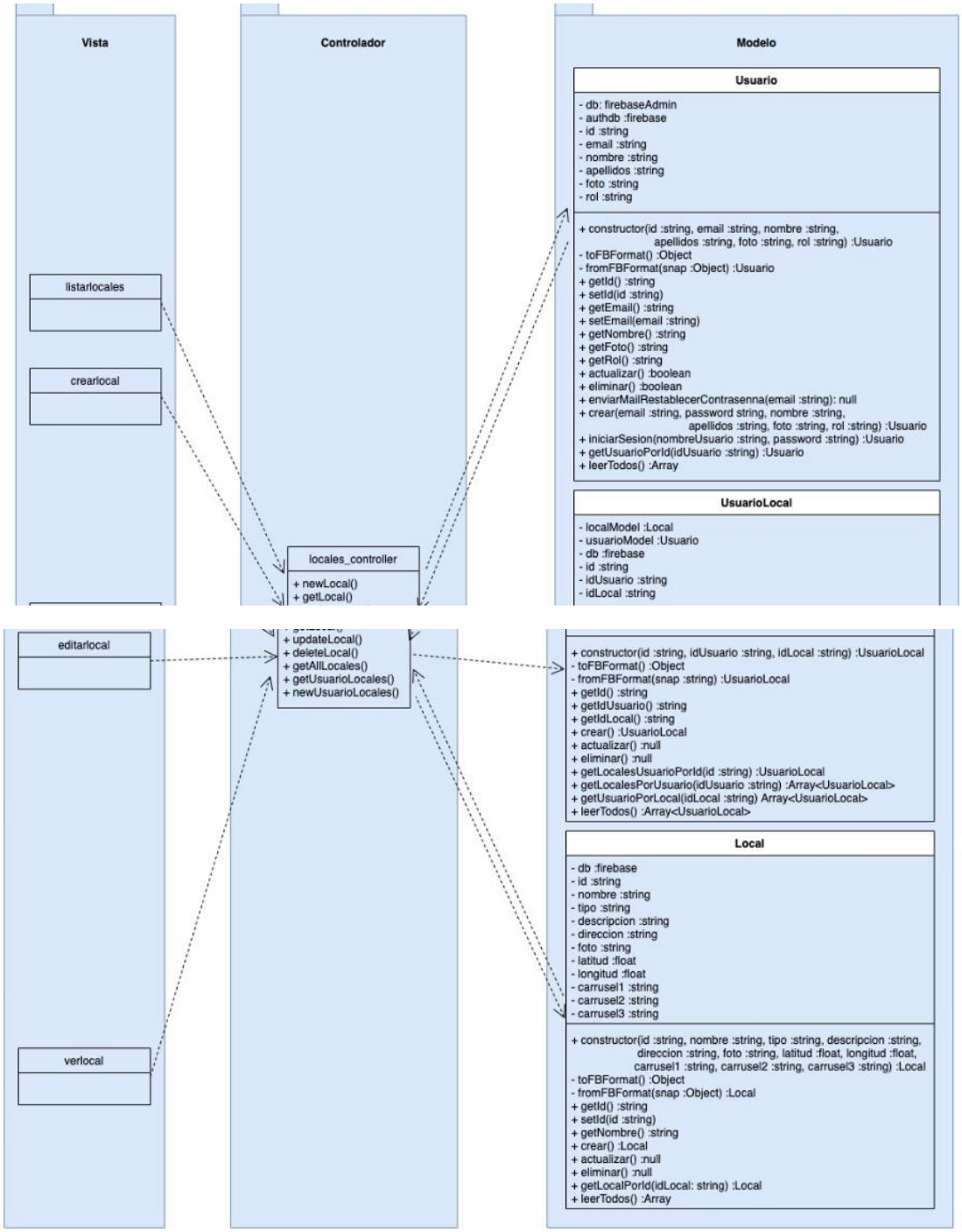
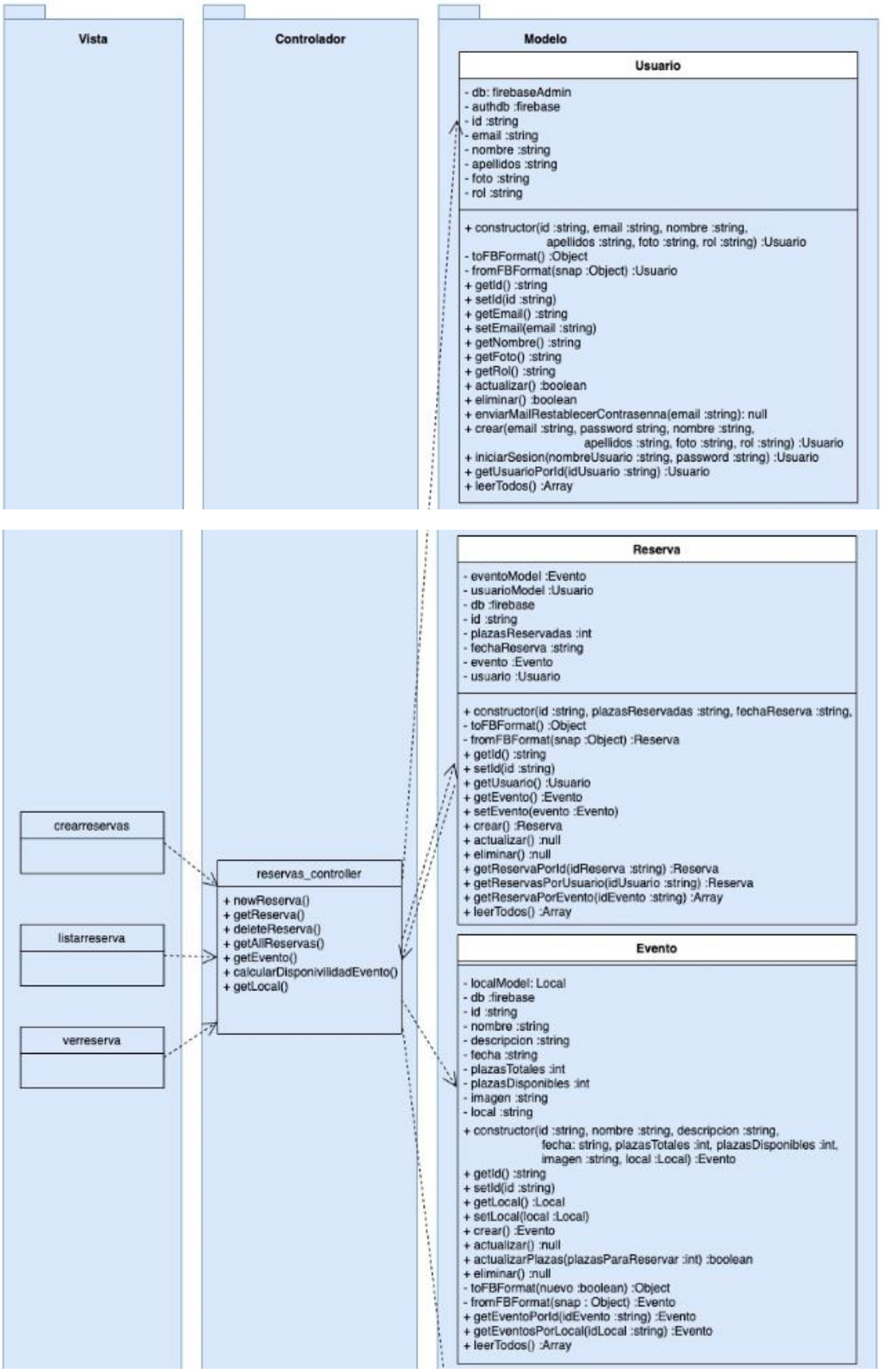


Imagen 5.2-DIAGRAMA DE CLASES (LOCALES)


Vista
Controlador
Modelo
Usuario
- db: firebaseAdmin
- authdb :firebase
- id :string
- email :string
- nombre :string
- apellidos :string
- foto :string
- rol :string
+ constructor(id :string, email :string, nombre :string, apellidos :string, foto :string, rol :string) :Usuario
- toFBFormat() :Object
- fromFBFormat(snap :Object) :Usuario
+ getId() :string
+ setId(id :string)
+ getEmail() :string
+ setEmail(email :string)
+ getNombre() :string
+ getFoto() :string
+ getRol() :string
+ actualizar() :boolean
+ eliminar() :boolean
+ enviarMailRestablecerContrasenna(email :string): null
+ crear(email :string, password string, nombre :string, apellidos :string, foto :string, rol :string) :Usuario
+ iniciarSesion(nombreUsuario :string, password :string) :Usuario
+ getUsuarioPorId(idUsuario :string) :Usuario
+ leerTodos() :Array
Reserva
- eventoModel :Evento
- usuarioModel :Usuario
- db :firebase
- id :string
- plazasReservadas :int
- fechaReserva :string
- evento :Evento
- usuario :Usuario
+ constructor(id :string, plazasReservadas :string, fechaReserva :string,
- toFBFormat() :Object
- fromFBFormat(snap :Object) :Reserva
+ getId() :string
+ setId(id :string)
+ getUsuario() :Usuario
+ getEvento() :Evento
+ setEvento(evento :Evento)
+ crear() :Reserva
+ actualizar() :null
+ eliminar() :null
+ getReservaPorId(idReserva :string) :Reserva
+ getReservasPorUsuario(idUsuario :string) :Reserva
+ getReservaPorEvento(idEvento :string) :Array
+ leerTodos() :Array
Evento
- localModel: Local
- db :firebase
- id :string
- nombre :string
- descripcion :string
- fecha :string
- plazasTotales :int
- plazasDisponibles :int
- imagen :string
- local :string
+ constructor(id :string, nombre :string, descripcion :string, fecha: string, plazasTotales :int, plazasDisponibles :int, imagen :string, local :Local) :Evento
+ getId() :string
+ setId(id :string)
+ getLocal() :Local
+ setLocal(local :Local)
+ crear() :Evento
+ actualizar() :null
+ actualizarPlazas(plazasParaReservar :int) :boolean
+ eliminar() :null
- toFBFormat(nuevo :boolean) :Object
- fromFBFormat(snap : Object) :Evento
+ getEventoPorId(idEvento :string) :Evento
+ getEventosPorLocal(idLocal :string) :Evento
crearreservas
listarreserva
verreserva
reservas_controller
+ newReserva()
+ getReserva()
+ deleteReserva()
+ getAllReservas()
+ getEvento()
+ calcularDisponivilidadEvento()
+ getLocal()

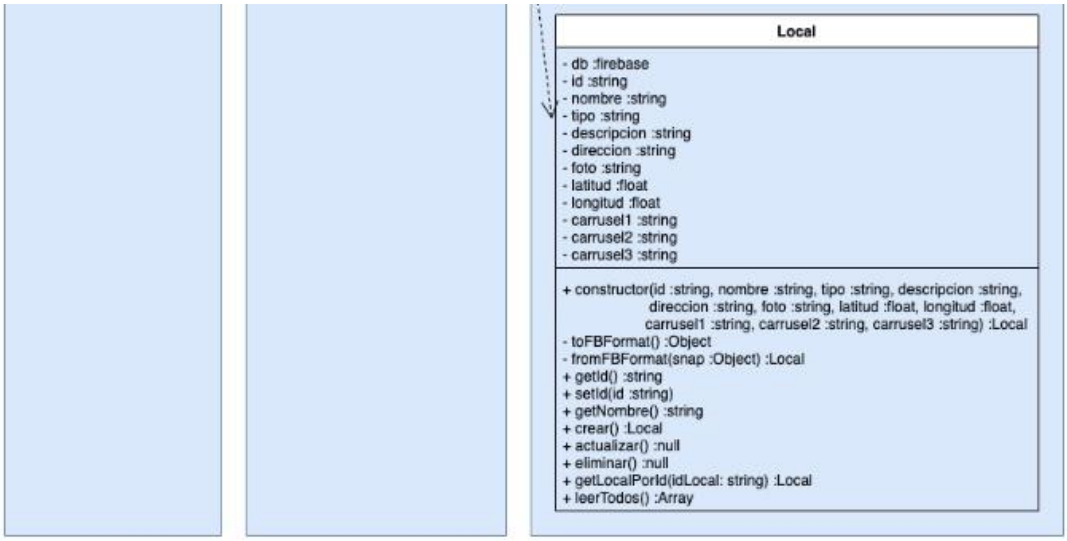


Imagen 5.3-DIAGRAMA DE CLASES (RESERVAS)

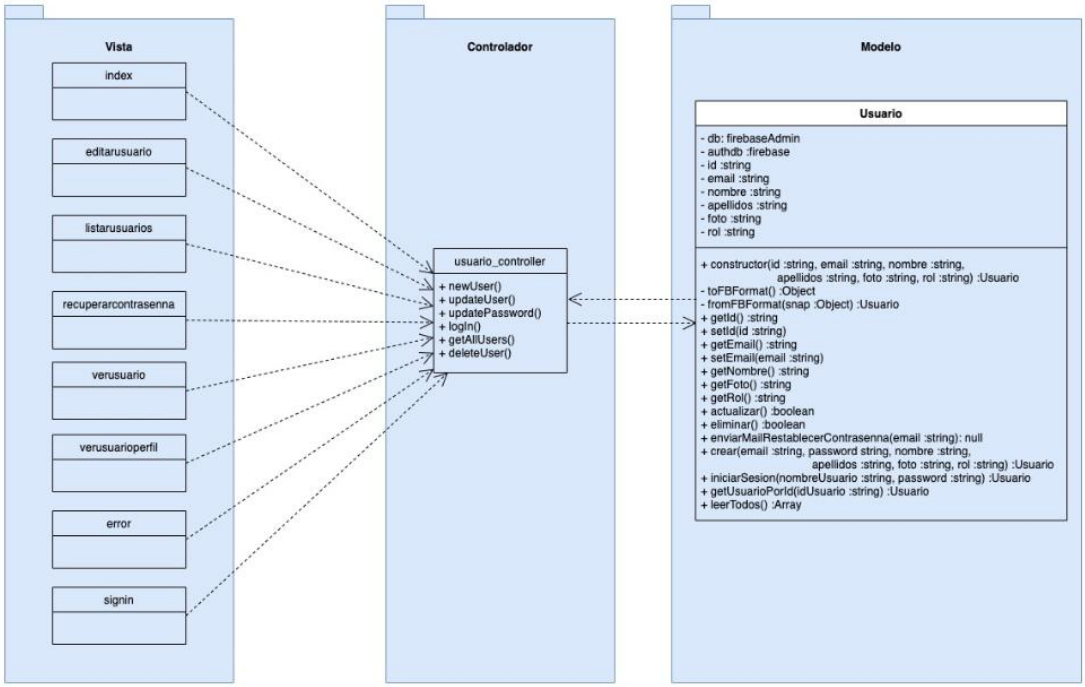


Imagen 5.4-DIAGRAMA DE CLASES (USUARIO)

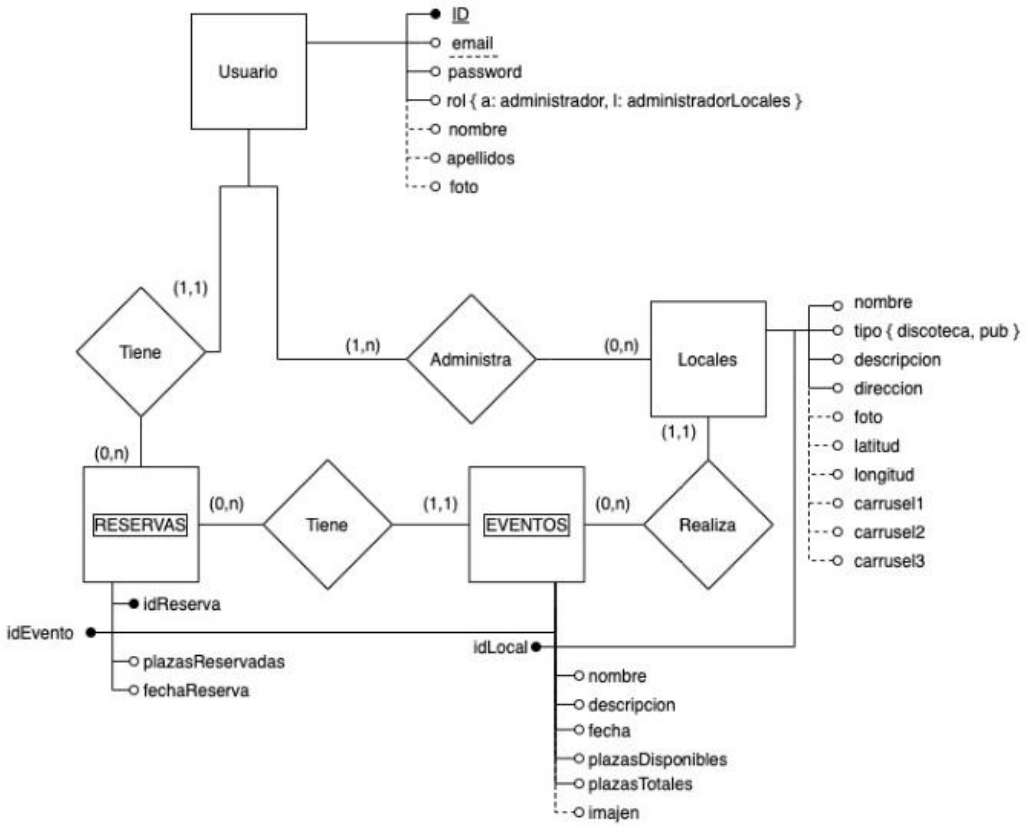


Imagen 5.5-DIAGRAMA E/R

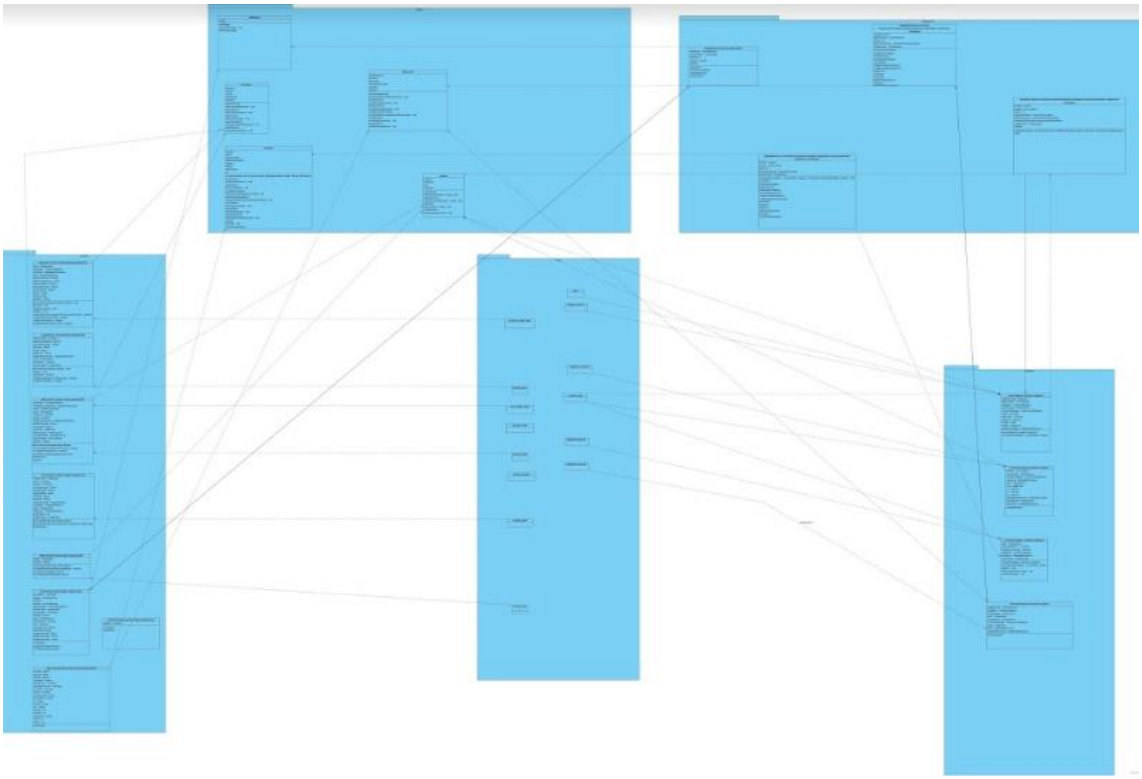


Imagen 5.6-DIAGRAMA UML

5.2 Fase 2 – Administración de Usuarios

En esta fase se comentará como el usuario puede registrarse o iniciar sesión.

5.2.1 AFANightApp

La descripción se va hacer desde la parte superior a la parte inferior**.**

Crear Usuarios - Registro (Ver imagen 5.2.1):

Crea correctamente los usuarios con un nombre y una url fija que referencia a una imagen. Estos datos se guardan en firebase realtime los datos de autentificación como el gmail y contraseña se guardan en firebase authentication, por lo tanto estos datos se guardan en la Base de Datos de la aplicación.

Imagen 5.2.1

Para registrarse, el usuario deberá rellenar un formulario que contiene ciertas restricciones:

-Ningún campo puede estar vacío.

-El email que se solicita debe tener el siguiente formato “Usuario@hotmail.com”.

-Se solicitan 2 contraseñas, deben coincidir las contraseñas.

-Debe aceptar las políticas de privacidad.

El registro también tiene otras funcionalidades, como dos botones deslizantes:

1. Un botón para poder visualizar la contraseña que el usuario escribe.

2. Un botón que le redirige al Login (activity login) cuando el usuario ya esté registrado podrá acceder directamente a la app desde ahí.

Si la autenticación ha sido un éxito, el usuario ya queda registrado, pudiendo acceder ya a todo el contenido de la app.

Login (Ver imagen 5.2.2):

Imagen 5.2.2

El usuario si ya tiene una cuenta de AFANight podrá iniciar sesión. Una vez que el usuario inicie sesión ya no le volverá a pedir la aplicación, los datos de autentificación.

Los datos requeridos son email y contraseña que se verificarán en firebase authentication

Esta ventana contiene el botón que le permite iniciar sesión, otro botón deslizante para poder visualizar la contraseña introducida y un texto en el que si hace click en él, podrá recuperar su contraseña introduciendo su correo electrónico, esto enviará un email a su correo, con la finalidad de poder cambiar la contraseña satisfactoriamente.

5.2.2 Web

De manera similar, en la web podrán darse de alta usuarios o hacer login si ya han creado una cuenta anteriormente, independientemente de si la cuenta se creó en la aplicación móvil o en la web.

Crear Usuarios - Registro (Ver imagen 5.2.3-5.2.4):

Para registrar un usuario utilizando la web es necesario dirigirse al apartado “iniciar sesión/registro.

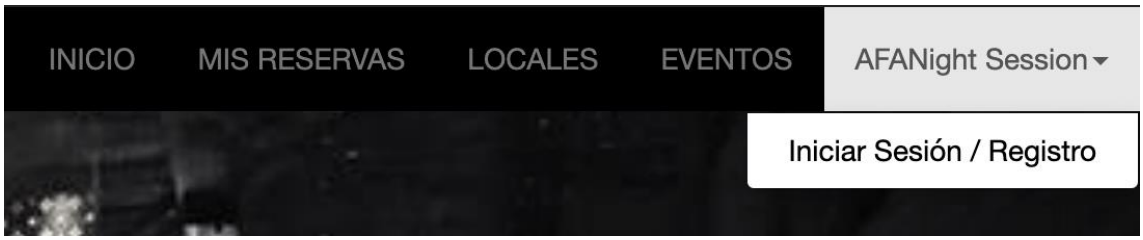


Imagen 5.2.3

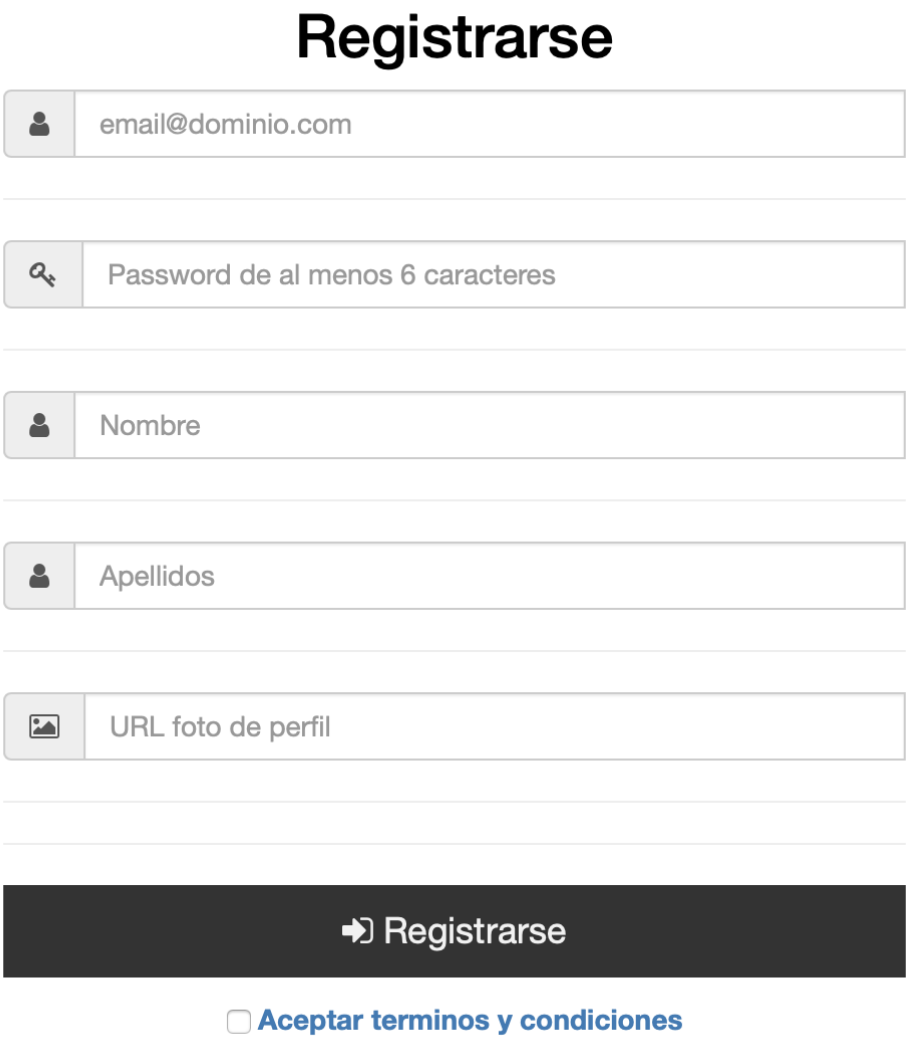


Imagen 5.2.4

Para completar el registro hace falta completar una serie de datos: email, password, nombre, apellidos y una url de una foto que se quiera usar como foto de perfil (Fig. 5.2.2-2). Siendo de estos el email y el password obligatorios y el resto opcionales. Para completar el registro es necesario leer y aceptar los términos y condiciones del servicio, para esto se dispone de un checkbox justo debajo del botón "Registrarse". Finalmente al presionar el botón "Registrarse" el usuario será dado de alta en el sistema, ganando acceso a ambas aplicaciones.

Login (Ver imagen 5.2.5):

El proceso de login es similar al de alta. Se debe contar con un usuario que haya pasado por el proceso de registro.

En esta ocasión deberá completar los datos en la sección "Iniciar Sesión". Debe ingresar su email y su contraseña y presionar el botón "Login". Existe la posibilidad de generar un mail de recuperación de contraseña siguiendo el enlace "¿Olvidaste tu Contraseña?"

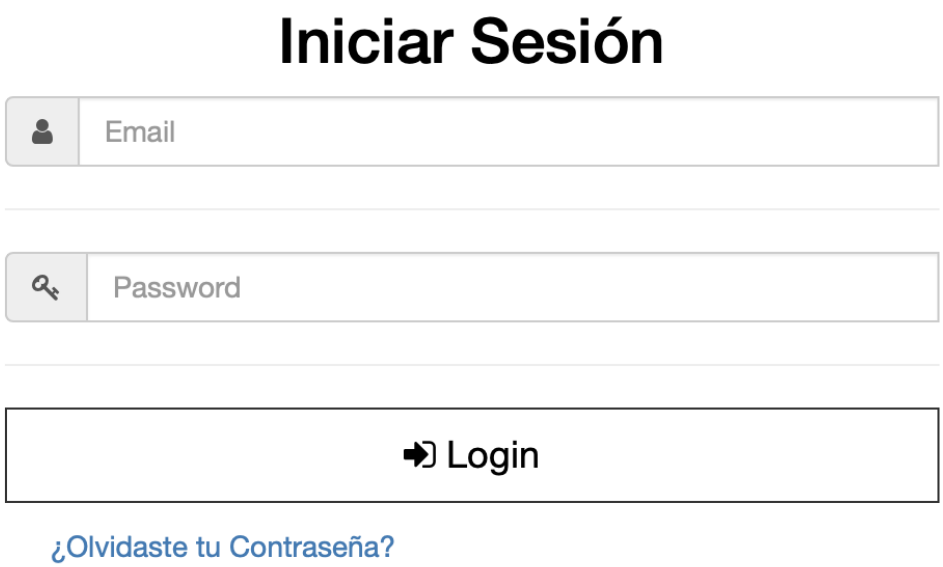


Imagen 5.2.5

5.3 Fase 3 - Contenido Principal

En esta fase se detallará el contenido tanto de la aplicación web, como la aplicación móvil.

5.3.1 Contenido AFANightApp (Móvil)

Barra deslizante:

-**Botón atrás**: oculta la barra deslizante (volviendo al mainActivity).

-**ImagenPerfil:** al hacer click sobre la imagen, se abre una pestaña con la información del usuario.

-**Texto Usuario**: se visualiza el nombre del usuario que ha iniciado sesión.

-**Menú compuesto por 5 elementos:**

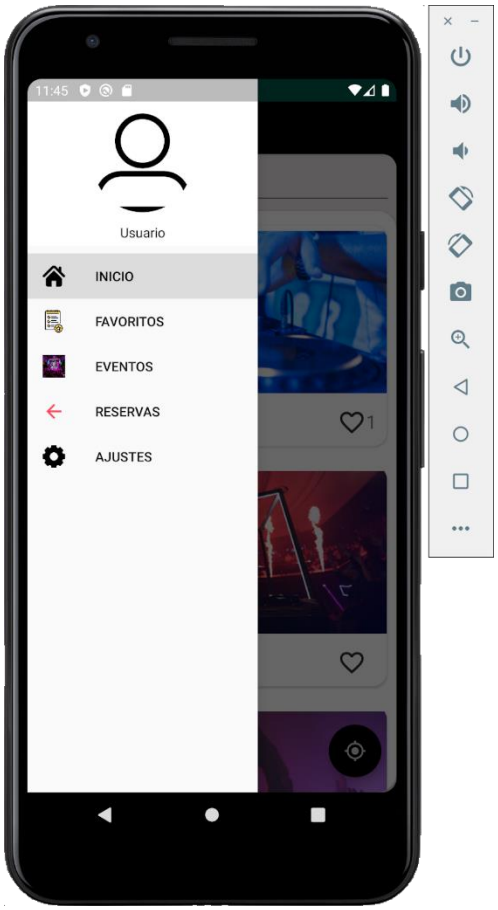
-Inicio.

-Favoritos.

-Eventos

-Reservas

-Ajustes.



Perfil:

-**Botón atrás:** (volviendo al mainActivity).

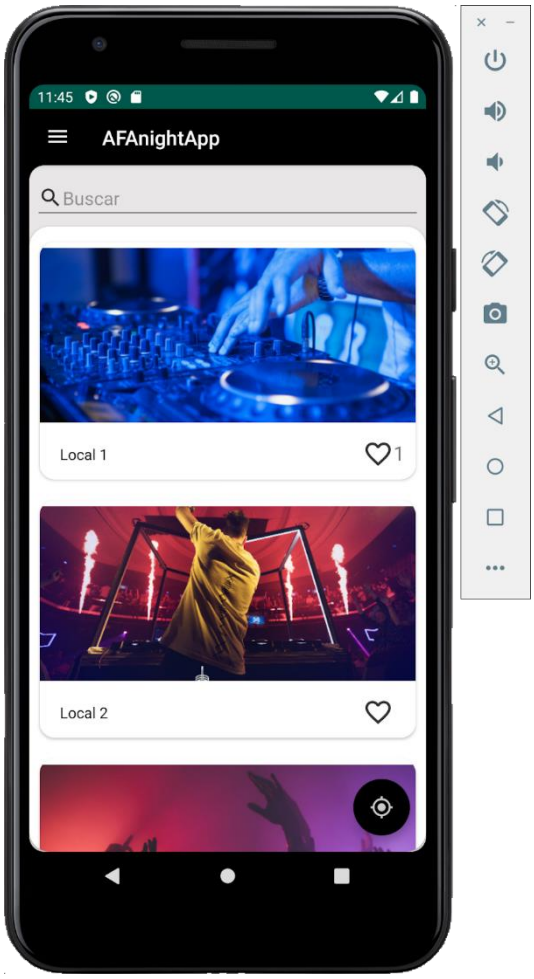
-**Imagen de perfil:** se visualiza la imagen del perfil del usuario. y pinchando sobre la imagen, se sube al storage y si todo ha salido correcto, envía la URL a la foto url de la base de datos.

-**Texto Usuario:** se visualiza el nombre del usuario.

-**Texto Email:** se visualiza el correo del usuario.

-**Botón Contador:** contador de interacciones con los locales.

-Inicio



Este fragmento contiene un buscador formado por un editex y un logo, en este caso el icono de la lupa.

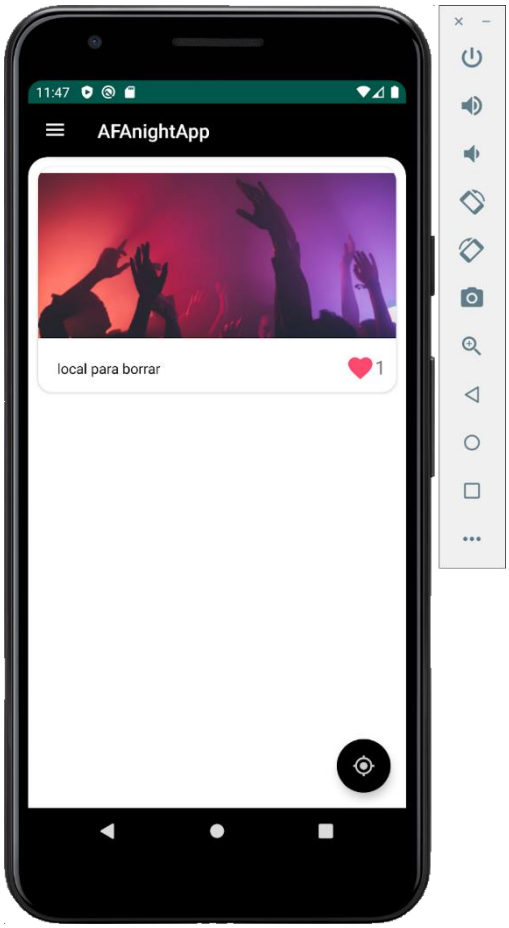
Debajo tiene un RecyclerView que mediante un adaptador y la referencia a la base de datos se cargarán los datos. Este recyclerview tiene un modelo que está formado por un carview con una imagen, el nombre del local, con la posibilidad de añadirlos a favoritos

Si hace click al CardView, le llevará a una nueva ventana con más información acerca del local.

Dentro del inicio (mainActivity) también se encuentra un **botón flotante**, que es botón del **mapa**, el mapa contiene botones de encontrar "**mi localización**", como también de aumentar y disminuir el zoom al mapa y las marcas de los locales que tenemos en el fragmento de inicio.

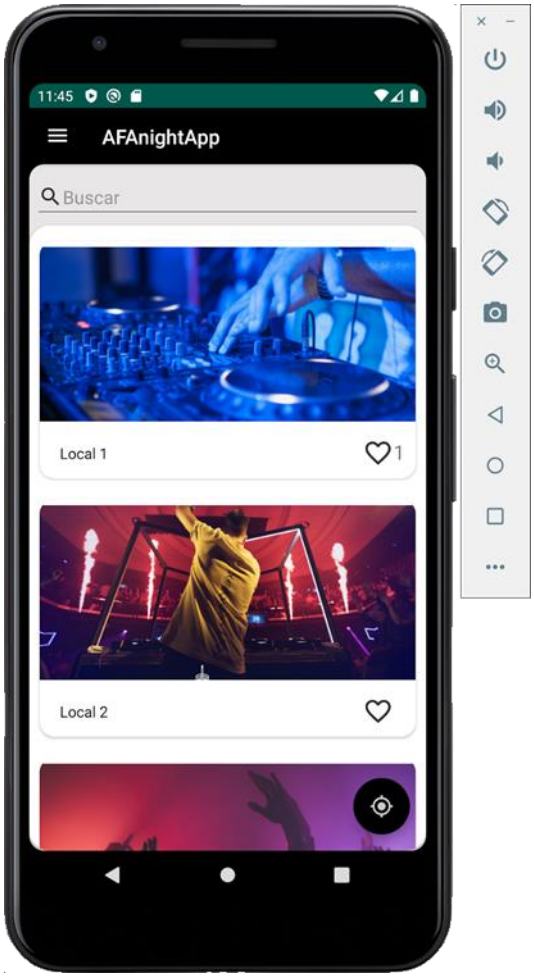
-Favoritos

En este fragmento se muestra una información animando a utilizar la opción de guardar en favoritos con una breve explicación, una vez que detecte un objeto añadido al RecyclerView, esta información se ocultará.



-Eventos

En este fragmento, se cargará un CardView sobre eventos en Madrid con una imagen, una fecha y el número de plazas del evento, si hacemos click en el CardView, podrá acceder a una ventana con más información donde podrá reservar las entradas del evento, esta acción crea un evento. Para realizar la reserva se utiliza **Transition**.



-Reservas

Este fragmento recibirá las reservas creadas una vez se creen se filtraran por el idUsuario y se cargaran solo las reservas del usuario estas reservas se pueden eliminar mediante un botón.

-Ajustes

Este fragmento se encuentran dos secciones: eliminar cuenta y cerrar sesión

5.3.2 Contenido AFANight Events (Web)

La web emplea una barra de navegación en la parte superior que permite a los usuarios dependiendo de su nivel de acceso ingresar en diferentes secciones.

Usuario invitado

Es el usuario con el mínimo nivel de acceso, este usuario no requiere de hacer login en la web para consultar estas secciones. Este usuario ver una barra de navegación como la mostrada en la **Fig. 5.2.2-4**. Desde aquí este usuario podrá consultar locales y eventos. Si encontrara un evento para el que quisiera reservar entradas, al presionar el botón reservar (**Fig. 5.2.2-5**) será redirigido a la página de login/registro, para que primeramente acceda como un usuario registrado.

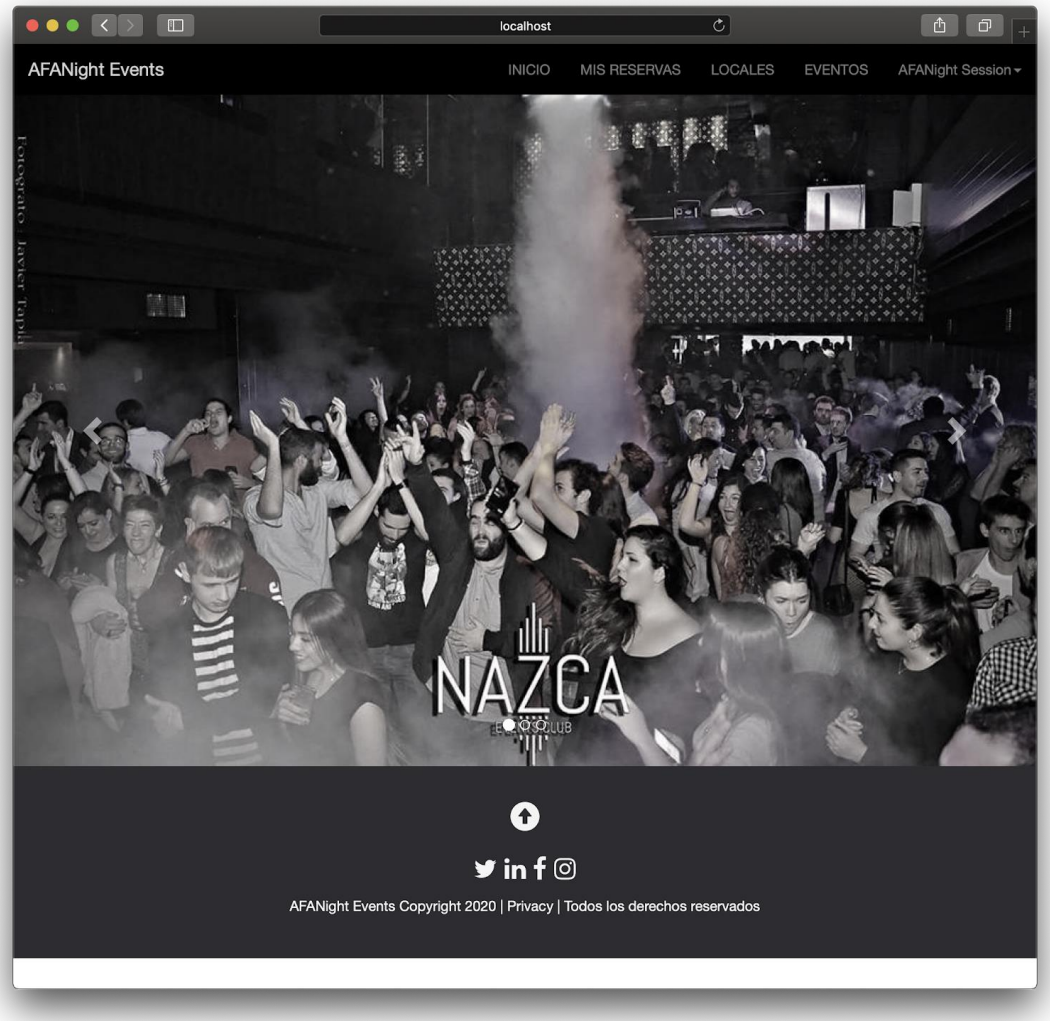


Fig. 5.2.2-4

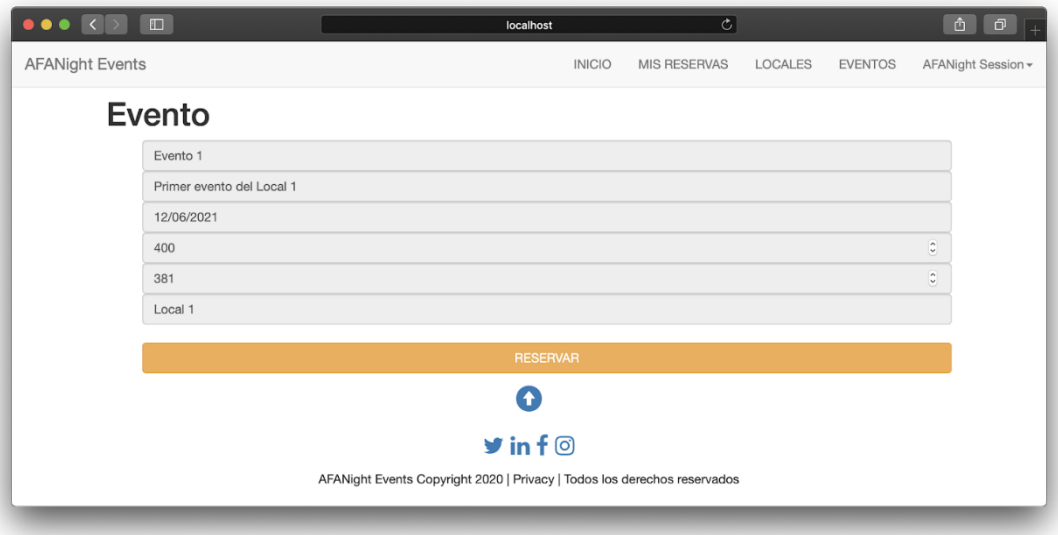


Fig. 5.2.2-5

Usuario

Este usuario además de lo explicado para el anterior, tiene la habilidad de hacer reservas y ver una lista de las reservas realizadas. En cuanto al menú de navegación cuenta con los mismos ítems, pero sin requerir un paso de registro. En el momento de realizar una reserva, se le solicitará al usuario que ingrese la cantidad de plazas que desea reservar (**Fig. 5.2.2-6)**. El sistema comprueba que hay suficientes plazas disponibles y en caso afirmativo genera la reserva. A partir de este nivel de usuario todos cuentan con una página de perfil de usuario donde podrán modificar sus datos personales, cambiar contraseña o incluso dar de baja la cuenta (**Fig. 5.2.2-7)**.

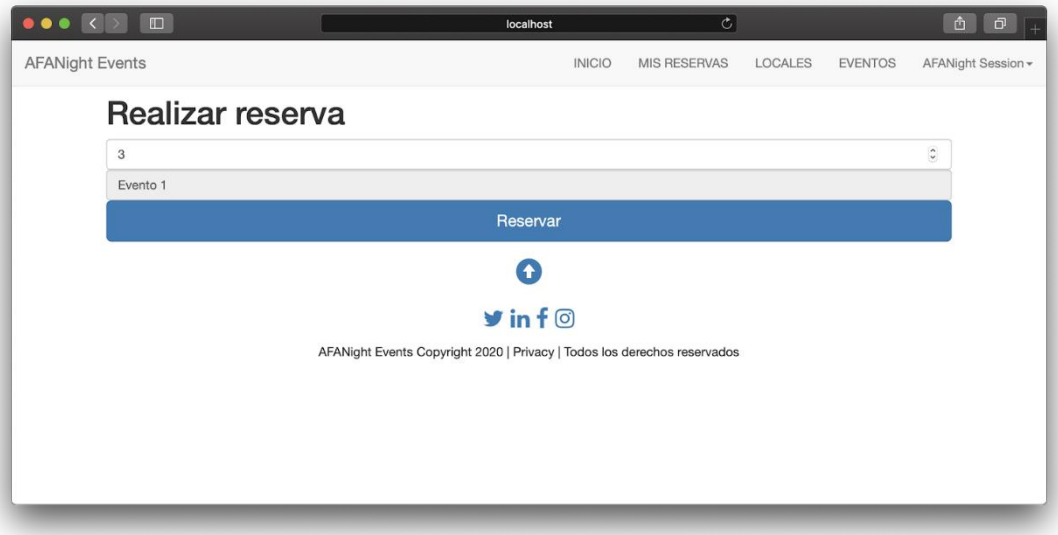


Fig. 5.2.2-6

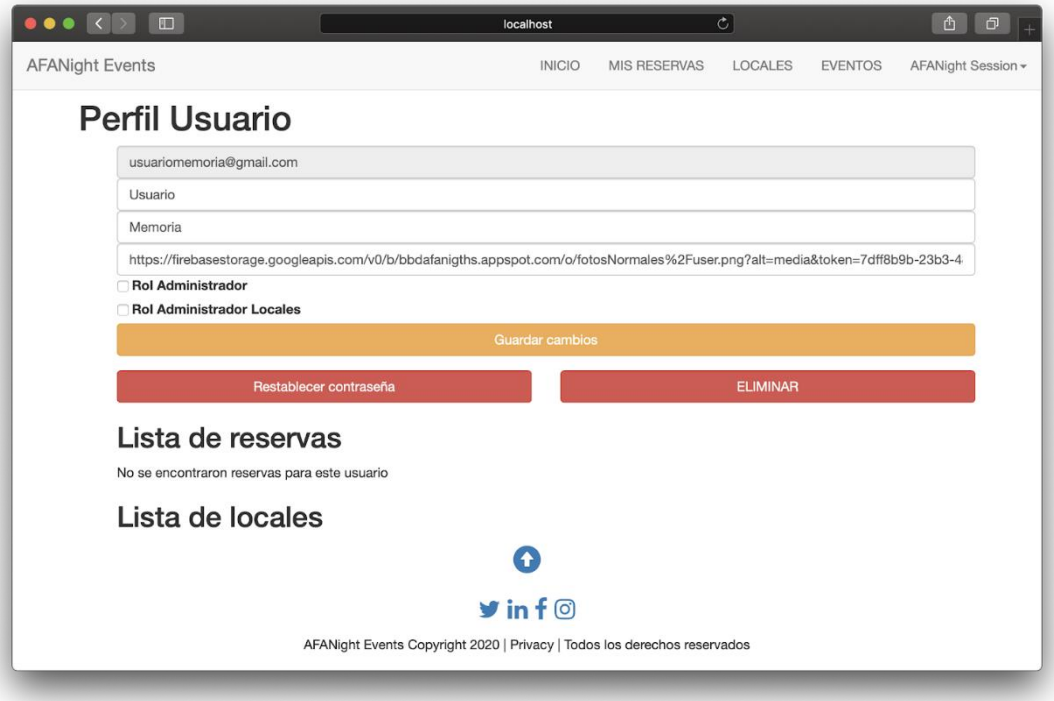


Fig. 5.2.2-7

El siguiente nivel de acceso es el usuario "administrador de locales": este usuario podrá gestionar locales y eventos, contando con items en la barra de navegación que le facilitarán esta tarea (**Fig. 5.2.2-8)**.

Finalmente el último perfil de usuario es el de administrador, este perfil suma la gestión de usuarios (**Fig. 5.2.2-9)**.

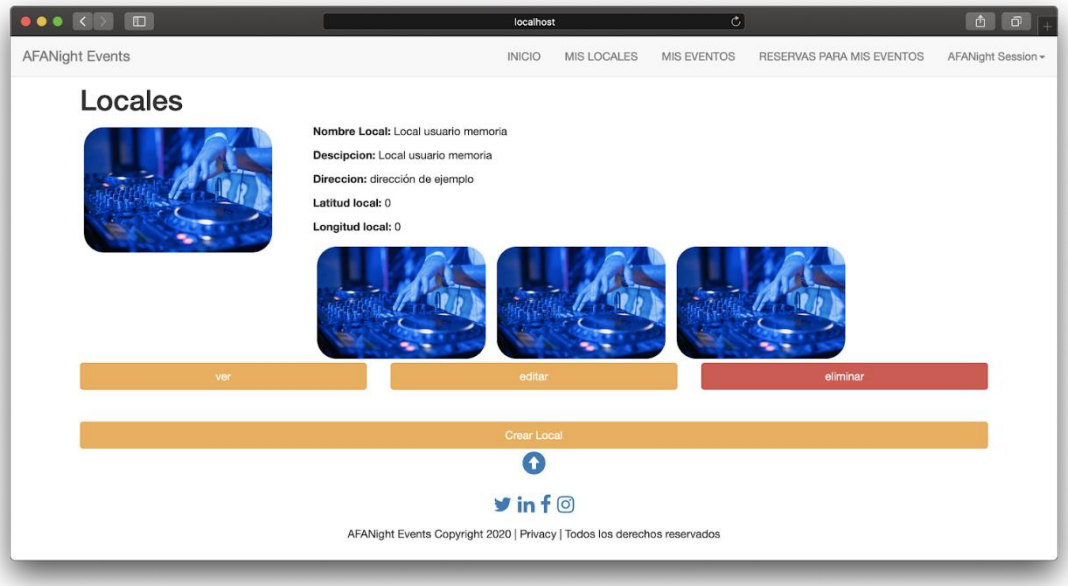


Fig. 5.2.2-8

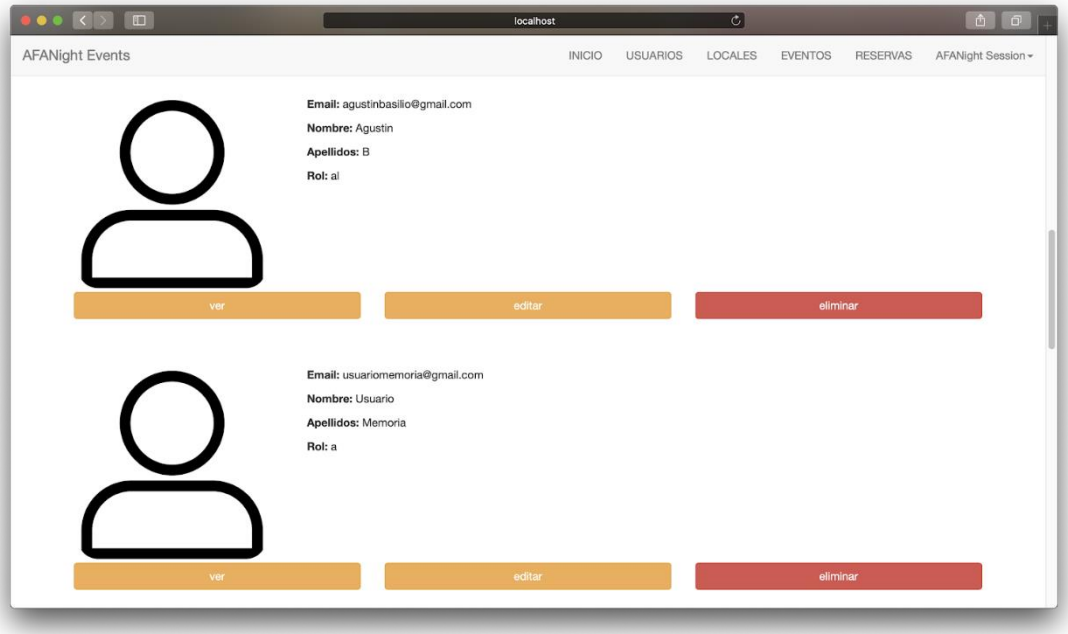


Fig. 5.2.2-9

Capítulo 6. CONCLUSIONES

En este capítulo se muestran las conclusiones de este TFG y además se proponen propuestas de trabajos futuros, en cuanto a las posibles mejoras, tanto de la aplicación web como de la aplicación móvil.

6.1 Conclusiones

En este apartado se discutirán los objetivos específicos logrados y dando una justificación de porqué motivo se han conseguido superar. Cumpliendo con los objetivos específicos, se ha logrado desarrollar una aplicación web y móvil que servirá para que los usuarios puedan realizar reservas de diferentes tipos de eventos de manera fácil y sencilla, siempre y cuando los usuarios se hayan registrado, así como la posibilidad de favorecer a los clientes (dueños) de los locales y establecimientos, registrándose en modo administrador para poder añadir el número de entradas que habrá en el evento.

A continuación se definen los objetivos específicos y se comprueba si se han conseguido de manera satisfactoria.

OBJETIVO	JUSTIFACIÓN	ÉXITO

O1 Plan de proyecto	Se ha desarrollado un plan de proyecto y un estudio de viabilidad (sección) del proyecto donde se estiman costes y tiempo de realización.	SI
O2 Diseño sólido de la aplicación	Se ha realizado un diseño sólido con todas sus funcionalidades.	SI
O3 BBDD con Firebase	Se ha creado correctamente la conexión con la base de datos.	SI
O4 Aprendizaje Node.js, Express.js, Java, JavaScript, Bootstrap, HTML5 y CSS	Herramientas aprendidas en estos dos años, puestas en práctica.	SI
O5 Desarrollo de la aplicación con sus respectivas tecnologías	Se han aplicado las tecnologías más comunes en la actualidad para realizar estas dos aplicaciones.	SI

Tabla 6.1: Tabla resumen de objetivos

Como se ven efectuados todos los objetivos específicos, el objetivo principal de este TFG, no es otro, que el desarrollo de una aplicación web y una aplicación móvil, que permita al usuario poder reservar y ver las reservas realizadas en diferentes eventos, y permitir a los dueños de los locales registrarse en modo administrador para poder publicar el evento.

6.2 Propuestas futuras-Mejoras

En este apartado del capítulo se van a proponer algunas mejoras para una posible comercialización.

1- La aplicación web, se basa en la gestión de reservas de eventos, dónde los usuarios se deben registrar para poder acceder y realizar dichas reservas con éxito, también la aplicación ofrece la posibilidad de que los dueños de locales puedan registrarse en modo administrador para publicar sus eventos en la página.

En este apartado se pueden realizar un par de mejoras que podrían estar interesantes:

- Un buzón de sugerencias, en el cual los usuarios podrían realizar consultas a los dueños que están registrados en la web, para aclarar dudas acerca del aforo, horarios…etc.

- Añadir la ubicación de los eventos, para que los usuarios puedan saber si les viene bien ir a ese evento, debido a su ubicación y distancia.

- Añadir los eventos mensuales más importantes a la página principal.

2- En cuanto a la aplicación móvil, que también se basa en la gestión de reservas de eventos, tiene como característica principal, añadir a "eventos favoritos" los eventos seleccionados, como si fuera Instagram, aparte que también ofrece poder ver la ubicación donde se encuentra o se va a realizar el evento.

En este apartado también se pueden realizar un par de mejoras que podrían estar interesantes:

- Mejorar la interfaz de usuario.

- Añadir otra clase de eventos como por ejemplo, conciertos.

En conclusión, son aplicaciones que ofrecen ciertas características diferentes que a las del resto. Puede ser una aplicación importante, si realizamos las mejoras explicadas anteriormente e incluso realizar otras mejoras de cara al futuro.

Capítulo 7. BIBLIOGRAFÍA

[1] **Evolución aplicaciones web**. Disponible en: https://unidad1programacionweb.wordpress.com/2014/03/04/1-1-evolucion-de-las-aplicaciones-web/, publicado el 04 Marzo, 2014.

[2] **Evolución aplicaciones web**. Disponible en: https://prezi.com/l4ttjs1jd_hs/11-evolucion-de-las-aplicaciones-web/, publicada y actualizada por Jaime Ávila el 15 Agosto, 2019.

[3] **Evolución aplicaciones móviles**. Disponible en: https://prezi.com/odplkcum7c1l/evolucion-de-las-aplicaciones-para-dispositivos-moviles/, publicada y actualizada por Fer García el 19 Diciembre, 2016.

[4] **MVC**. Disponible en: https://desarrolloweb.com/articulos/que-es-mvc.html, publicado por Miguel Ángel Álvarez el 02 Enero, 2014.

[5] **MVC**. Disponible en: https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador

[6] **Node.js**. Disponible en: https://nodejs.org/es/

[7] **L. Peris (10 marzo, 2019) Plataformas web: PHP vs Node.js**. Disponible en: https://luisperis.com/plataformas-web-php-vs-nodejs/

[8] **Herramientas similares a node.js**. Disponible en: https://blog.aulaformativa.com/alternativas-mvc-framework-para-node/, publicado el 23 agosto, 2019

[9] **Herramientas similares a node.js**. Disponible en: https://blog.aulaformativa.com/utiles-herramientas-para-el-entorno-de-programacion-node-js/, publicado el 23 julio, 2019

[10] **Los mejores editores gratuitos html, javascript y css.** Disponible en: https://www.campusmvp.es/recursos/post/Los-10-mejores-editores-gratuitos-de-HTML-CSS-y-JavaScript.aspx, publicado por José Manuel Alarcón el 28 Mayo, 2019.

[11] **Firebase**. Disponible en: https://es.wikipedia.org/wiki/Firebase

[12] **MongoDB**. Disponible en: https://es.wikipedia.org/wiki/MongoDB

[13] **Las mejores aplicaciones para la gestión de proyectos**. Disponible en: https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/las-mejores-aplicaciones-de-gestion-de-proyectos/, publicado el 17 Julio, 2019.

[14] **Herramientas similares a Bootstrap**. Disponible en: https://www.emezeta.com/articulos/alternativas-a-bootstrap

[15] **Diferencias desarrollo entre Android e IOS**. Disponible en: https://pickaso.com/2019/desarrollo-apps-diferencias-android-vs-ios, publicado por Georgina Palau el 8 Enero, 2019.

[16] **Diferencias desarrollo entre Android e IOS**. Disponible en: https://www.paradigmadigital.com/dev/versus-desarrollo-ios-vs-desarrollo-android/, publicado por Héctor Rubial y Jorge Calleja, 2017.

[17] **"Aplicaciones para eventos de ocio nocturno"**. Disponible en: https://www.lowi.es/blog/las-mejores-apps-fiesteras/, publicado el 27 Junio, 2018.

[18] **Whatsapp**. Disponible en https://whatsapp.com

[19] **Backlog**. Disponible en https://backlog.com

[20] **Asana.** Disponible en: https://asana.com

[21] **Trello.** Disponible en: https://trello.com/es

[22] **Etapas de un sprint desarrollo del Scrum**. Disponible en: https://obsbusiness.school/es/blog-investigacion/project-management/las-5-etapas-en-los-sprints-de-un-desarrollo-scrum

[23] **Guía Scrum**. Disponible en: https://www.scrumguides.org/scrum-guide.html#artifacts-productbacklog

[24] **Mario Marugán Cancio**, "PHELT: Planificador de Horarios en Entornos Laborales a Turnos", Trabajo Fin de Grado, Grado en ingeniería en informática, Universidad de Castilla-la Mancha, CR, España, 2015.

7.1 Acrónimos. Anexo A

En este apartado se mostrará un listado con los acrónimos encontrados en la memoria del TFG.

- **TFG**: Trabajo Fin de Grado.
- **MVC**: Modelo-Vista-Controlador.
- **HTML**: HyperText Markup Language.
- **CSS3**: Cascading Style Sheets 3.
- **PHP**: Hypertext Preprocessor.
- **IDE**: Integrated Development Environment.