+

ABSTRACT

Nowadays, **social relations has became an important task from our daily lives**. We are continoulsy interacting with everybody around us. Nevertheless, sometimes and in certain situations and contexts**, doing this type of tasks can be thought and difficult**. If you are new into a town or you want to do activitys you enjoy with someone when no one is avaliable, **meeting new people is not an easy task**. Unfortunatley, there is any type of solution to this type of problems, and, the existing ones, are used by big companys for Team-Building activitys. **What we need rigth now, is an app that can be used by particulars on their day a day activities.**

The present document has as main goal to **develop an app that allows users to create social events to meet people.** In order to achieve this, we studied the structure of existing social networks. Also, **we learnt about differente technologies and programming languages** that are typically used during the develop of this type of services, such as PHP, XAMPP, Amazon EC2 and different JARS that allows us to to introduce differente funcionalities. **We were also able to use different APIS and endpoints** just as we could have access to mobile services such as GPS and internal storage.

**The app developed during this document has received a good feedback from users that we used as beta-testers**. It have been demonstrated that the main goal we pursuit has been reached. This mobile app **can be used to create and join social events** using an intuitive interface and, more important, we have found a solution to the initial problem we disccused at the introduccition. **It can be used by anyone, no matter gender or color.**

**The development of this app demonstrante the potencial for technology to enchace our social lives** and build new connections with communities we enjoy. We have created a service that can be really used to enjoy new activityes with people you havent met before.

JUSTIFICACIÓN

El siguiente trabajo se presenta como parte del programa de materias del módulo de Desarrollo de Aplicaciones Multiplataforma.

**Las interacciones sociales y el ocio son una parte importante de nuestro día.** Estas interacciones ocurren prácticamente todos los días. Aún así, **a veces puede llegar a ser complicado el conocer a nuevas personas** o el realizar planes de ocio que disfrutamos con gente que conocemos.

**Hoy en día no hay muchas soluciones que ayuden a poner un parche a este problema** y, las que hay, están orientadas a realizar talleres de Team-Building en empresas más que estar orientadas al uso real de particulares.. Es debido a esto, que vemos necesario el entregar una solución que ayude a arreglar dicho problema de forma efectiva.

**Tenemos 4 objetivos principales** que se cumplirán durante el desarrollo de este proyecto:

1. Ayudar a personas con dificultades a la hora de conocer gente.
2. Comprender y estudiar el corrector funcionamiento de distintas
3. Analizar el funcionamineto interno de las redes sociales.
4. Crear una herramienta con una utilidad real y un funcionamiento útil.
5. Conocer y compender el uso de herramientas de control de versiones.
6. Comprender y analizar como distintos sistemas pueden interaccionar entre sí.

**Considermaos que la aplicación desarrollada y descrita en este TFG ayudará a resolver este problema,** proporcionando una solución profesional y, muchas más importante, funcional, que pueda ser utilizada por cualquier persona sin importar su lugar de procedencia o preferencias.

INTRODUCCIÓN

A lo largo de nuestra historia, **los humanos hemos vivido en sociedad**. **Somos seres sociables**, y, por lo tanto, **dichas interacciones representan una parte importante de nuestra vida**, tanto en la parte privada como en la parte pública.

Estas mismas, a pesar de su aparente sencillez, **pueden llegar a ser extremadamente complicadas para algunas personas,** sobre todo si eres nuevo en una ciudad o trabajo haciendo que conocer gente nueva puede ser una tarea muy complicada.

**La finalidad de este proyecto es precisamente facilitar esta tarea a todo tipo de personas que lo necesiten**, por medio del desarrollo de una red social que permita a los usuarios registrados en esta tanto **crear eventos sociales de cualquier tipo** y tópico como poder unirse a eventos de otros usuarios.

**Estos eventos los verá el usuario dependiendo de su ubicación** y de un radio en km delimitado por el mismo. Así mismo podrá buscar eventos en ubicaciones especificas así como eventos con nombres especificos.

Con ello, **conseguimos crear una herramienta que facilita a las personas el conocer a gente que comparten sus gusto y aficiones**, además de permitir realizar actividades en compañía. Así, **cualquier persona que tenga dificultades para socializar o encontrar nuevos amigos**, puede hacerlo, asegurandose por el camino que tienen gustos y aficiones en común, consiguiendo así evitar los típicos sentimientos de soledad acompañados a estos.

**Para la elaboración de dicho proyecto, usaremos distintas tecnologías**. Primero de todo, la plataforma para la que estará disponible serán los teléfonos Android. Usaremos **como lenguaje de programación Java**, acompañado de ciertas tareas que serán realizadas **usando PHP**. Contaremos con una base de datos, siendo esta el **servicio MySQL que proporciona el paquete de aplicaciones XAMPP**. Dicha base de datos se conectará al WorkBench de MySQL para poder permitir las conexiones remotas a la base de datos usando el teléfono móvil.

**El IDE utilizado será Android Studio**, pues tiene las funcionalidades necesarias para poder realizar aplicaciones móviles. Aún así, **para las tareas en las que necesitamos realizar el desarrollo utilizando PHP**, el Entorno de Desarrollo Integrado utilizado es Visual Studio

OBJETIVOS

Evidentemente, hay una serie de puntos que vamos a cumplir durante el desarrollo de este TFG. Estos puntos descritos más abajo ayudarán tanto a nuestro desarrollo profesional como a tener una mejor comprensión de los sitemas utilizados durante este tipo de desarrollos y a poder mejorar los servicios que tenemos como objetivo el proporcionar:

1. **Ayudar a personas con dificultades a la hora de conocer gente**, a que puedan realizar dicha tarea en un entorno seguro y cómodo para ellos. Queremos poder entregar una solución intuitiva, fácil de usar, y que de verdad suponga un cambio para cualquier individuo que tenga dificultades para relacionarse o conocer gente en un entorno desconocido para él.
2. **Comprender y estudiar el corrector funcionamiento de distintas tecnologías** como PHP o XAMPP para la realización de tareas de desarrollo. Con ello, podremos entender y analizar las ventajas y utilidades que tienen dichas herramientas a la hora de desarrollar nuevos proyectos.
3. **Analizar el funcionamineto interno de las redes sociales** y de la relación que guardan con las bases de datos, así como poder entender el funcionamiento de este tipo de sistema para tener una mejor comprensión de estas para futuros proyectos.
4. **Crear una herramienta con una utilidad real y un funcionamiento útil** que pueda ayudar a las personas en su día a día. Queremos poder crear una herramienta que proponga una solución real así como un impacto positivo en la sociedad.
5. **Conocer y compender el uso de herramientas de control de versiones**, así como de otras utilidades que permiten a los desarrolladores el poder compartir código entre los distintos integrantes de un equipo.
6. **Comprender y analizar como distintos sistemas pueden interaccionar entre sí** para poder proporcionar distintos servicios que pueden ser utilizados por los usuarios.

Al final de este documento, y del desarrollo de la aplicación aquí presentada, **esperamos el haber cumplido dichos objetivos de forma efectiva, aportando soluciones útiles a los problemas planteados anteriormete.**

**Nuestro mayor deseo es poder aportar un cambio real** a un problema al que muchas personas se tiene que enfrentar sin saber como lidiar exactamente con ellos.

ESTIMACIÓN DE COSTES

DESARROLLO DEL PROYECTO

TECNOLOGIA UTILIZADA

Antes de explicar este apartado, **hemos de entender cuales son las funcionalidades** que contendrá nuestra aplicación:

1. **Creación de cuentas** dentro del servicio ofrecido, enviando un correo de confirmación de cuenta.
2. **Creación de eventos sociales** en distintas ciudades, con distintas plazas y, así como nombre y descripción.
3. **Sistema de busqueda de eventos según la ubicación del usuario**. Posibildiad de bscar eventos según ubicación, usuario creador del evento así como nombre del evento.
4. **Links asociados a cada evento** a los que te puedes apuntar por medio de entrar a estos mismos.
5. **Creación de QR especficos para caada evento creado,** a los que te puedes unir por medio de escanear dichos QR

Así pues, teniendo en cuenta esta información, podemos entender y justificar el por que de los distintos elementos que utilizaremos para el desarrollo de este trabajo.

**Lo primero de todo y lo más importante es conocer cuales son los entornos de desarrollo que utilizaremos.** En este caso, contaremos con 2 IDE distintos:

* Android Studio:

**Es el IDE principal y donde estará desarrollada la mayoría de los servicios de nuestra aplicación.** La utilización de este IDE se debe a la comodidad que nos proporcioda y a la familiriadad que tenemos con este.

* Visual Studio:

**Se usa principalmente para poder desarrolar las 3 páginas web usando PHP** que necesitamos para poder enviar un correo de confirmación así como asociar una página web especifica a cada evento. Estas 3 páginas desarrolladas son:

* + Evento:

**Es la página web que tiene asociada un evento** para poder unirte a este automaticamente al acceder a dicha página.

* + Inicio de sesión:

**Permite al usuario logearse para poder confirmar asistencia** a los eventos de la página anterior.

* + Confirmación de cuenta:

**Cuando el usuario accede a esta, la cuenta creada por el usuario es confirmada automaticamente** y por tanto el usuario es capaz de acceder a ella.

**Otro aspecto importante es considerar cuales son los lenguajes de programación que vamos a utilizar para el desarrollo.** En este aspecto, contamos con 2 lenguajes de programación distintos:

* Java:

**Es el lenguaje de programación principal y en el que estará desarrollada la mayoría de la app.** Su uso se debe a las facilidades que nos proporciona y que es uno de los lenguajes principales utilizados para este tipo de desarrollos. Así mismo, **la gran comunidad de usuarios distintos así como a un compilado más rápido y el poder crear aplicaciones modulares, la hace una opción más interesante** que la alternativa de usar Kotlin. A tener en cuenta que Android Studio nos permite transformar el código de Java a Kotlin en caso de ser necesario.

* PHP:

**Lo utilizaremos como un lenguaje de apoyo para poder desarrollar las páginas web que necesitaremos descritos anteriormente**. Su uso frente a otros lenguajes backend como podria ser Node.js se resume a dos hechos principales: lo primero de todo que es **el lenguaje que hay que usar para poder usar el servicio MySQL del paquete de XAMPP**, y segundo, debido a la sencillez de la sintaxis utilizada para realizar la conexión a la base de datos. Al ser realmente 3 páginas web que no son el elemento principa de la aplicación y cuya unica tarea es realizar conexiones a la base de **datos, nos podemos permitir no utilizar otros lenguajes con mejores o más funcionalidades.**

En la aplicación utilizaremos distintas librerias para realizar distintas tareas. Estos JAR serán:

1. Java Mail:

**Lo usaremos para poder enviar mensajes desde un correo especificado por medio de nuestra aplicación Java**. Tenemos que crear una contraseña en nuestra cuenta de google para poder permitir que aplicaciones de terceros puedan acceder al correo para realizar tareas (en este caso, enviar mails con un link de confirmación de cuenta)

1. Zxing:

**Es la librería encargada de poder crear codigos QR.** **Lo usaremos para poder guardar información en una imagen** que estará asociada a cada evento creado. En este caso, guardara un link que servira para poder confirmar un evento a través de este.

1. JDBC:

**Necesaria para poder realizar la conexión a la base de datos desde el teléfono móvil.** Así, podemos realizar distintas consultas en la base de datos que estarán relacionadas con distintas tareas que se pueden hacer en la base de datos.

Así mismo, tenemos que tener en cuenta otras consideraciones:

1. XAMPP:

Para realizar ciertas tareas de la aplicación**, necesitamos utilizar páginas webs y urls,** como es el caso del envió del correo de confirmación, el cual envia al email del usuario una página web en PHP para poder confirmar las creación de la cuenta**. Es debido a esto que la base de datos que utilizamos esta desarrollada con phpMyAdmin,** pues así podemos acceder a ella por medio de PHP para realizar dichas funciones.

1. MySQL Workbecnh:

Evidentemente, **necesitaremos conectarnos a la base de datos que tenemos con el dispositivo móvil.** Es cierto que podriamos conectarnos directamente al MySQL de phpMyAdmin. Sin embargo, debido a las facilidades que proporciona MySQL Workbench para realizar tareas de desarrollo, hemos sincronizado dicha plataforma con phpMyAdmin por medio del puerto 3306.

1. SQLite:

**Es la base de datos propia con la que cuentan los dispositivos móviles Android.** Lo usaremos para poder guardar el nombre y la contraseña del usuario para poder automatizar el logeo a la aplicación. **Si el usuario se ha logueado con anterioridad, su cuenta se guardará en la dicha base de datos** y usaremos dichos datos para poder iniciar sesión de forma automática.

1. Endpoints/API’S:

**Dentro de nuestra aplicación, tambien utilizamos APIS que nos proporcionan distinta información útil.** Utilizamos la API de Nominatum, que nos proporciona dos servicios disintos:

* Comprobación de si una calle especificada realmente existe:

**Necesaria para comprobar si la ubicación del evento introducida por el usuario es una ubicación real.** Con esto, evitamos que personas no deseadas puedan acceder a nuestra página.

* Pasar a latitud y longitud una ubicación:

**Usada para poder obtener las coordenadas geográficas de una localidad.** Esta inforamción será utilizada para calcular la distancia entre la ubicación del usuario (por medio de su GPS) y la localidad de los distintos eventos guardados en la base datos. Si la distancia es mayor que la delimitada por el usuario, dicho evento no se muesta al usuario.

Evidentemente, necesitaremos permitir las conexiones remota para que los usuarios puedan acceder a la información almacenada en la base de datos. Para ello, usaremos 2 servicios distintos:

* Amazon EC2:

**Contaremos con una instancia de EC2 donde alojaremos la base de datos que usaremos para guardar la inforamción de las cuentas de los usuarios.** En este servidor, instalaremos XAMPP, y crearemos las regals de entrada necesarias para poder acceder remotamente a la base de datos y a las páginas diseñas con XAMPP. El puerto que abriremos para este caso será el 3306.

* Ngrok:

**Es una aplicación que usaremos para realizar pruebas y comprobar el correcto funcionamiento de nuestra pásginas creadas con PHP.** Además, un aspecto importante es que te asigna automatica una URL. **Funciona como un tunel,** que nos permite redireccionar el tráficoel puerto que especifiquemos para permitir accesos remotoros a loss recursos guardas en la carpeta htcdocs. **El puerto a abrir en cuestión sera el puerto 80.**

**Todas estas tecnologías descritas son las que tendremos que utilizar para poder poner en marcha los servicios ofrecidos en la aplicación.** Importante recalcar que todas y cada una de ellas son necesarias para permitir un uso óptimo y correcto de la aplicación.

PLANIFICACIÓN

ESPECIFICACIÓN REQUISITOS

**El siguiente apartado es uno de lo más importantes de todo este proyecto**. Es en este donde delimitaremos que puntos tenemos que cumplir para poder considerar que el proyecto desarrollado ha sido cumplido con éxito. **En este punto, vamos a describir tanto los requisitos funcionales como no funcionales de la aplicación a desarrollar.**

El objetivo principal de este apartado es **poder definir de forma clara y precisa los puntos que tenemos que cumplir**. Antes de empezar tenemos que distinguir entre dos tipos de requisitos distintos:

* Requisito no Funcionales:

Se refiere a los requisitos relacionados con conceptos como la seguridad, usabilidad así como con el rendimiento.

* Requisitos Funcionales:

**Se refiere a las funcionalidades que la aplicación** **ha de tener implementadas,** pudiendo ser utilizadas por los usuarios de la aplicación.

**Es la combinación de ambos tipos de requisitos lo que permitirá que la aplicación creada pueda ser considerada como un éxito.** En caso de que alguno de los puntos expuesto más adelantes no se pudieran cumplir exitosamente, podriamos considerar que, aunque no sea un fracaso, el nivel de calidad no sea el esperado.

**Es por ello que el delimitar estos requitios es tan importante**, pues nos permitirá analizar y comprender que elementos son los que marcarán la diferencia para poder desarrollar un proyecto funcional, óptimo y profesional.

REQUISITOS NO FUNCIONALES

Como ya hemos explicado en el apartado anterior, estos requisitos son los que tienen que ver con conceptos más relacionados con la seguridad y la usabilidad.

En este sentido, los objetivos principales tendrán que ver con el Sistema Operativo para el que se desarrollará la aplicación, versión del lenguaje de programación… En general, con elementos no relaciodos con los servcios que ofrecemos en la app. Dichos requisitos son los siguientes:

* Seguridad:

Vamos a manejar datos privados del usuario, como podría ser su ubicación, así como imágenes y hobbys. Es por esto que la seguridad es uno de los elementos fundamentales que tendremos que cubrir, pues no podemos permitir accesos de terceros a este tipo de información sensible. Para ello, habremos de:

* Usar métodos de encriptamiendo como Hashes para poder cifrar la contraseña del usuario cuando esta sea guardada dentro de la base de datos.
* Usaremos PreparedStatements para impedir que ataques por medio del sistema SQLInjection, para que no se pueda extraer toda la información alojada en la base de datos.
* Escalabilidad:

Hemos de crear un sistema que sea capaz de crear y manejar un gran cantidad de usuarios simultáneos distintos. Debe de poder ser capaz de crecer de forma paralela con la cantidad de usuarios. Para con ello:

* Utilizaremos distintos eventos para realizar tareas de mantenimiento en la base de datos para así no sobresaturar de información desfasada y poder alojar nuevos datos.
* La arquitectura manejará un gran número de datos y usuarios simulatenaos sin afectar al rendimiento gracias a la utilización de consultas avanzadas.
* Monitoreamos MySQL para conocer el estado del servidor y poder identidificar y erradicar posibles cuellos de botella que se generen en la aplicación.
* Usabilidad:

Todo tipo de usuario ha de ser capaz de poder utilizar Jogo sin importar el nivel de familiaridad que tenga con este tipo de tecnologias, pudiendo entender todo de manera sencilla.

* Nos aseguraremos de tener elementos gráficos debidamente diferencieados, con estos mismos correctamente etiquetados para poder diferenciar entre las distinas partes de una pantalla.
* Usaremos códigos de colores así como apartados para que el usuario de un simple vistazo sepa a que zona quiere ir para utilizar las distinas funcionalidades que estarán a disposición del usuario.
* Disponibilidad:

Jogo a de poder ser utilizado en cualquier momento del día. Para ello, tanto el servidor como la base de datos que utilizamos a de estar levantada de forma constante para permitir conexiones remotas a esta. Además, Tomcat y Apache han de estar iniciados para poder enviar correos de confirmación de cuenta en caso de ser necesario.

* Para ello, el servidor tendrá que estar constamente levantado, con las reglas de entrada configuradas correctamente para permitir conexiones remotas por medio del puerto 3306.
* Rendimiento:

Vamos a usar varios hilos de ejecución para poder optimizar el tiempo de ejecución que se tarda en realizar ciertas tareas como inicializar distintas pantallas que cuentan con varios elementos gráficos así como para acelerar ciertos procesos como la inserción de los eventos creados en la base de datos.ç

* Usaremos sentencias avanzadas para poder optimizar el rendimiento de las consultasa que se hagan a la base de datos.
* Tendremos varios hilos de ejecución que se encargarán de realizar distintas tareas, permitiendo menores tiempos de carga entre las distintas pantallas.

Todo estos elementos permitirán que Jogo una usabilidad óptima y segura para el usuario, que estará disponible en cualquier momento sin importar el día o la hora.

REQUISITOS FUNCIONALES

Aquí detallaremos las funcionalidades que la aplicación tendrá que tener implementadas. Todas y cada una de las funcionalidades aquí descritas son esenciales para poder asegurar que los servicios que tenemos como objetivo el ofrecer puedan ser utilizados efectivamente por cualquier tipo de persona que quiera utilizar la aplicación.

DISEÑO DE LA APLICACIÓN

PROTOTIPOS

La aplicación consta de 11 pantallas distintas:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Logotipo, nombre de la empresa

Descripción generada automáticamente

REGISTRO CORREO

PANTALLA INICIAL

INICIO DE SESIÓN

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente



VER EVENTOS

REGISTRO USUARIO

REGISTRO USUARIO

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

ASISTENCIAS

BUSCAR EVENTOS

CREAR EVENTOS

Interfaz de usuario gráfica, Texto, Aplicación, Sitio web

Descripción generada automáticamente

PERFIL SEGUIDOR

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Sitio web

Descripción generada automáticamente

PERFIL

SEGUIDORES

Las pantallas son las siguientes:

* Pantalla Inicial:

**Es la pantalla inicial de nuestra aplicación.** Muestra el logo de la aplicación. Por debajo, busca en la base de datos de SQLlite la existencia de algún nombre de usuario o contraseña introducidas con anterioridad. Si existe, salta a la pantalla de Ver Eventos. En caso contrario, salta a la de inicio de sesión.

* Inicio de sesión:

**Es la pantalla de inicio de sesión de nuestra aplicación.** Loguea al usuario en la aplicación y, en caso de ser necesario, introduce el nombre y la contraseña introducidas en la base de datos de SQLite para poder loguearse automaticamente.

* Registro de Correo:

**Es la pantalla encargada de registrar un correo en la aplicación.** Lo utilizamos para poder enviar un correo de confirmación de creación de cuenta para la aplicación. Comprueba que el correo introducido no exista con anterioridad en la base de datos.

* Registro de Usuario:

**Es la pantalla encargada de registrar un nombre de un usuario en nuestra aplicación.** Comprueba que el nombre del usuario no exista con anterioridad.

* Registro de Contraseña:

**Es la pantalla encarga de registrar la contraseña del usuario en la cuenta.** Así mismo, desde aquí se envia un email de confirmación al correo del usuario desde el correo de la aplicación.

* Ver evento:

**Permite ver los eventos cercanos en nuestra aplicación.** Podemos escoger tanto el radio de lejania de los eventos, como buscar eventos por nombre o ubicación asi como encontrar usuarios.

* Crear Eventos:

**Permite crear eventos.** Los usuarios se podrán apuntar a este evento una vez haya sido creada.

* Asistencias:

**Permite comprobar a que eventos y a cuantos eventos hemos confirmado la asistencia.** Estan ordenados según la fecha de confirmación de asistencia. Desde aquí, podemos cancelar la asistencia del evento.

* Perfil:

**Es la pantalla de perfil del usuario de la aplicación.** Permite ver tanto los seguidores como a quien seguimos así como comprobar los eventos que hemos creado, y si han finalizado o no. Tambien desde aquí podemos editar el perfil.

* Perfil persona:

**Es la pantalla de perfil de los usuarios.** Sirve para poder comprobar la información del perfil de un usuario que nos siga. Podemos ver eventos creados y confirmar asistencias.

* Seguidores:

Es la pantalla para ver los seguidores que tiene un usuario especifico. Podemos acceder al perfil de estos seguidores pulsando sobre ellos, saltando así a la pantalla de seguidores.

* Editar Perfil:

**Es la pantalla que permite al usuario cambiar la información de su perfil.** Permite cambiar el nombre de usuario, la descripción así como la foto de perfil que se le mostrará al usuario que acceda al perfil.

Hay algunas pantallas que están contenidos dentro de otras, pues a estas se acceden por medio de distintos fragmentos. Esto se hace para dinamizar la interacción entre usuario, aplicación e interfaz, pues limitamos así la cantidad de pantallas de carga con las que hay que interactuar para poder realizar distintas operaciones.

BASE DE DATOS

Primero de todo. La base de datos que vamos a utilizar es una base de datos relacional. Hay 4 razones principales para ello:

1. Estructura de datos coherentes:

**Con las bases de datos relacionales, nos podemos asegurar de que todos los datos guardados están debidamente organizados** además de que son fácilmente accesibles por los distintos usuarios. Esto es un aspecto importante, pues nos podemos cerciorar de que nuestros clientes puedan visualizar información detallada de los dinstintos eventos creados.

1. Complejidad de las consultas:

**Podemos elaborar consultas realmente complejas, juntando distintas tablas entre sí,** así como utilizando distintas funciones, que serán utilizadas por los usuarios para poder encontrar eventos, ya sea delimitados por la distancia respecto a su ubicación o bien por el nombre, ubicación o creador del evento.

1. Escalabilidad:

**Este punto se centra en que es capaz de manejar unas grandes volúmenes de datos y un gran volumen de usuarios.** Es realmente importante pues es una aplicación que puede crecer rápidamente con el paso del tiempo-

1. Restricciones:

**Podemos establecer restricciones de integridad** para asegurarnos de que los datos introducidos en la base de datos cumplen con unos límites que hayamos constituido con anterioridad.

Todas estas ventajas hacen que la opción correcta para el desarrollo, consistan en pasar por medio de un modelo relacional en vez de un modelo NoSQL.

**Así pues, la base de datos está construida usando el servicio de MySQL, por medio de phpMyAdmin, proporcionada por el paquete de XAMPP.** Esto debido a que usamos PHP para realizar 2 operaciones distintas:

* Enviar un correo de confirmación de creación de cuenta al momento de crear una cuenta en la aplicación.
* **Permitir a un usuario confirmar asistencia a un evento** al momento de escanear el código QR de un evento por medio de la aplicación de Google Lens

Para poder realizar la conexión a la base de datos por medio de Java, hemos usado el Driver JDBC de MySQL. Hemos conectado la base de datos que esta guardada en phpMyAdmin a la aplicación de MySQL Worckbench por medio del puerto 3306. Así, hemos podido acceder a la base de datos usando Java sin afectar el acceso usando PHP.

**Así mismo, para poder permitir conexiones remotas, hemos añadido la línea *bind-addres: 0.0.0.0*** para poder hacer que dispositivos que se encuentras en nuestra misma red se puedan conectar a la base de datos y puedan realizar cambios por medio de un usuario creado previamente.

El modelo Entidad/Relación de la base de datos es el siguiente:

Diagrama

Descripción generada automáticamente

Podemos encontrar 4 tablas distintas:

**Contienen toda la información referente a la cuenta del usuario**. Guarda datos tales como el nombre del usuario, la contraseña, el email, la descripción de la cuenta del usuario y un booleano para conocer si la persona a validado su cuenta. Un usuario puede crear y asistir a muchos eventos.

Así mismo, un usuario puede seguir a otros muchos usuarios.

USUARIOS

**Contiene toda la información necesaria para crear eventos.** Contiene el identificador del evento, la información de la ubicación del evento (calle, ciudad, localidad), la hora y día del evento, el nombre junto con la descripción y las plazas, así como el nombre del creador del evento, el código QR del evento, la latitud y la longitud.

Un evento puede ser creado por un único y a un evento acuden varios.

EVENTO

SEGUIDORES

**Es la tabla que se encarga de relacionar a los usuarios entre sí.** Un usuario puede seguir a otro, para poder ver los eventos de este o bien poder ver la información de perfil de la otra persona, ya sea él número de seguidores o a los usuarios a los que sigue.

**Es la tabla encargada de guardar los asistentes de un evento.** Contiene el id del evento y el nombre de usuario del asistente. Realmente es una tabla intermedia resultante de una relación N:M entre la tabla de Evento y Usuario.

ASISTIR EVENTO

La base de datos cuenta con eventos que se encargarán de realizar ciertas tareas de mantenimiento para eliminar los datos que no son utilizados en ciertos momentos.

aplicación

DIAGRAMAS UML

IMPLEMENTACIÓN Y DOCUMENTACIÓN

ARQUITECTURA/ESTRUCTURA

La aplicación desarrollada en este documento es una red social. Por lo tanto, el tipo de arquitectura utilizada para este proyecto consiste en una arquitectura cliente-servidor.

Este tipo de arquitectura se divide en dos partes. La primera, el cliente, interactúa directamente con el usuario final y la que pide una serie de recursos al servidor, y la segunda, el servidor, , que se encarga de cumplir con dichos estos pedidos y devolverle el resultado de dichos servicios al cliente.

Todo este sistema descrito, se compone de 3 capas distintas que se encuentran en niveles diferentes. Generalmente, estas capas son:

* Capa de presentación o interfaz de usuario:

**Es la capa encargada de comunicarse con el usuario final.** Es la parte correspondiente al ‘front-end’ de una aplicación. Esta permite que los usuario de una aplicación puedan interaccionar con el sistema.

* Capa lógica o de negocios:

**Es la capa encargada de procesar y manejar los distintos datos de la aplicación.** Se encarga de validad, procesar y manipular distinta información. Realmente, es una capa intermedia entre la capa de presentación y la capa de datos.

* Capa de datos:

**Es la capa que se encarga de guardar los datos de la base de datos.** Su función principal es la de dar persistencia y permitir la recuperación de los distintos datos que necesita una aplicación para poder cumplir con los servicios ofrecidos correctamente.

Estas capas, generalmente, interactúan entre sí de la siguiente manera:

CAPA

LÓGICA

CAPA DE

DATOS

CAPA

PRESENTACIÓN

1. **El usuario interacciona con un elemento de la capa de presentación**. Dicha interacción puede darse de cualquier manera y forma, ya sea pulsando un botón, deslizando la pantalla…
2. **Dicha interacción desencadena una comunicación con la capa lógica o de negocio**, haciendo que un recurso sea pedido a la capa de datos, encargada de almacenar distinta información utilizada por la aplicación.
3. **La capa de datos recoge dicho recurso de la base de datos y lo reenvía a la capa lógica.** Es esta capa quien lo vuelve a enviar, esta vez a la capa de presentación para poder mostrar al usuario la información que se había pedido por medio de dicha interacción.

En el caso concreto de nuestra aplicación de creación de eventos, dicho sistema se sigue al pie de la letra. Dentro de esta, tenemos estas 3 capas. La primera capa, que se representa por medio de las propias pantallas de nuestra aplicación, interacciona con la capa lógica, que es representada por medio de dos clases distintas:

* Conector:

Es la clase donde gestionamos las conexiones a la base de datos remota donde alojamos tanto los usuarios como los eventos creados en la aplicación.

* SQLite:

Es la clase donde gestionamos la conexión con la base de datos interna del móvil para almacenar tanto el ultimo nombre de usuario como la contraseña que se utilizó para iniciar sesión en el servicio.

Como ya hemos explicado, son estas dos clases distintas las que son utilizadas para almacenar toda la lógica de negocio, y, por tanto, son las que se utilizan para realizar cambios y pedir recursos a las bases de datos utilizadas en nuestro servicio.

Dicha arquitectura es la que es necesaria utilizar pues permite distribuir de manera eficiente la carga de trabajo. Así mismo, esto nos permite realizar escalados tanto horizontales como verticales. Más importante aún, esta estructura es la correcta pues en este caso concreto vamos a tener que gestionar continuamente interacciones con los usuarios.

CONTROL DE VERSIONES

El control de versiones se refiere a la práctica de llevar un registro y rastreo de los cambios producidos en el código fuente de un sistema, servicio o producto. Así, si los desarrolladores han cometido un error que ha dejado al producto sin funcionar, pueden volver a una versión anterior estable mientras encuentran y solucionan dicho error.

Para realizar dicha tarea de control de versiones, hemos usado la aplicación de GitHub, donde hemos subido los cambios más significativos realizados en el código a un repositorio. En este repositorio está subidos tanto las distintas versiones de la aplicación como los distintos os que hemos utilizado durante el desarrollo.

La rama principal es la siguiente:

* Master:

Es la rama principal de la aplicación y la que tiene la última versión estable de la aplicación. Justo después de finalizar el desarrollo de una funcionalidad nueva para la aplicación, se ha subido la rama de la funcionalidad y se ha realizado un merge request a esta rama principal.

Para separar las distintas funcionalidades que hemos ido implementando al código, hemos ido añadiendo mensajes específicos comentando los cambios realizados en el momento de hacer los *commits*.

Las distintas versiones de las aplicaciones han ido variando dependiendo de que funcionalidades se han ido introduciendo a los servicios de la aplicación (ver figura …).

Es importante tener en cuenta que, como muchas de las funcionalidades desarrolladas dependían del desarrollo de otras, algunos de los *commits* realizados han contado con más de una funcionalidad nueva implementada. Esto es, por ejemplo, lo que ha sucedido en el caso de creación e implementación de los código QR.

MAPA NAVEGACIÓN

ASISTENCIAS

CREAR

PERFIL

REGISTRO CORREO

REGISTRO NOMBRE.U

REGISTRO CONTRASEÑA

INICIO DE SESIÓN

PANTALLA INICIAL

SEGUIDORES

VER

BUSCAR

ESTÁNDARES/VALIDACIONES

PLAN DE PRUEBAS

DESPLIEGUE Y MANTENIMIENTO