UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

Flutter

Presentado por Samuel Casal Cantero
en Universidad de Burgos — 16 de julio de 2020
Tutor: César García Osorio y Francisco Javier Diez Pastor

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. José Francisco Diez Pastor y D. Cesár García Osorio. profesores del departamento de Ingeniería IJnformática. área de Lenguajes y Sistemas Informáticos. Expone:

Que el alumno D. Samuel Casal Cantero, con DNI 71301273p, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado Flutter.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 16 de julio de 2020

Vº. Bº. del Tutor:

D. nombre tutor

Vº. Bº. del co-tutor:

D. nombre co-tutor

Resumen

En este primer apartado se hace una **breve** presentación del tema que se aborda en el proyecto.

Descriptores

Android, Flutter, Dart, VisualStudio Code, Github, ...

Abstract

A **brief** presentation of the topic addressed in the project.

Keywords

Android, Flutter, Dart, VisualStudio Code, Github, ...

Índice general

Índice de figuras

Índice de tablas

Introducción

A principios de diciembre de 2019, fue la primera de las reuniones con los tutores, a fin de explicar las ideas propias o de barajar la opción de hacer el trabajo de fin de grado en la empresa, ITCL, donde estaba cursando las prácticas extracurriculares. Entre mis ideas estaba hacer un traductor de jeroglíficos o crear una base de datos de los grafittis de la calle. Al final me decanté por hacerlo en la empresa, ya que vi opciones para ello.

La prmera idea de proyecto era hacer un detector de moscas de la fruta. Este primer enfoque, no se llegó a iniciar, debido a que la empresa no estaba participado de una manera fluida y estaba en una fase muy temprana de desarrillo. El inicio de las conversaciones fue a mediados de enero del 2020, y a finales de febrero al no ver esperanzas, búsque otro trabajo entre los disponibles de la plataforma.

La segunda idea del proyecto, si que la comenzé, decicando numerosas horas al desarrollo e investigación, pero acabé dejandola de lado porque no me sentía motivado, ya que se reunierón muchos factores que me bloquearon.

Brevemente, consistia en la implementación de un formulario de login dentro de un nodo de Knime[**?**]. , con el fin de poder conectarse a la base de datos de Moodle, recogiendo los datos pertinentes, para hacer mineria de datos, con las herramientas de la aplicación.

Finalmente a un mes de la entrega para la segunda convocatoria, vi la necesidad de hacerlo, decantandome por una aplicación en Android. Una gran carga de trabajo, en caso de hacer algo que tenga entidad suficiente como para aprobar, pero la idea era intentarlo. Uno de los pasos inciales para comenzar a prograr para estos dispositivos, fue buscar cuéles de los lenguajes de programación actuales son más interesantes para el desarrollo.

Pensé en varios candidatos pero finalmente me decante por Flutter, ya que está desarrollado y respaldado por Google.

1.1. Estructura de la memoria

La memoria sigue la siguiente estructura:

- **Introducción:** breve descripción del problema a resolver y la solución propuesta. Estructura de la memoria y listado de materiales adjuntos.
- **Objetivos del proyecto:** exposición de los objetivos que persigue el proyecto.
- **Conceptos teóricos:** explicación de los conceptos teóricos clave para el entendimiento de la aplicación.
- **Técnicas y herramientas:** listado de técnicas metodológicas y herramientas utilizadas para gestión y desarrollo del proyecto.
- **Aspectos relevantes del desarrollo:** exposición de aspectos destacables que tuvieron lugar durante la realización del proyecto.
- **Conclusiones y líneas de trabajo futuras:** conclusiones obtenidas tras la realización del proyecto y posibilidades de mejora o expansión de la solución aportada.

Objetivos del proyecto

A continuación, se detallarán los objetivos que han motivado la realización de este proyecto así como los resultados que se desean conseguir.

2.1. Objetivos generales

- Desarrollar una aplicación para *smartphone*.
- Implemenación y despliegue de la app en la tienda de de Google.
- Hacer que los usuarios pasen un buen rato.
- Mostrar quien es el creador de la aplicación, con el fin de poder usar esta como herramienta o portfolio.
- Desarrollar juegos.

2.2. Objetivos técnicos

- Aprender una alternativa moderna a javascript mediante Dart.
- Comprender el funcionamiento de Flutter.
- Control de versiones con la herramienta GitHub, mediante comandos a través de GitBash.
- Generar documentación de todo el proceso en LaTeX.

- Realizar una planificación mediante *Scrum* eficiente, a través de la herramienta ZenHub, integrada en GitHub.
- Comenzar a usar las herramientas telemáticas para las reuniones con los tutores del trabajo final de grado.
- Comunicación de la aplicación mediante WebServices.
- Conocer y aprender a usar las herramientas que proporciona el *Cloud Services* de Google.
- Diseñar la arquitectura de la aplicación.

2.3. Objetivos personales

- Adquirir el conocimiento necesario para desarrollar aplicaciones móviles y multiplataforma. Es decir, para tres entornos, Android, iOS y web.
- Aprobar el trabajo de fin de grado, ya que es un reto.
- Estudiar como generar documentación en LaTeX.
- Aprender que sin esfuerzo no hay recompensa.
- Comprender el funcionamiento de la comunicación entre la una aplicación y los servicios en la nube.
- Reforzar los conocimientos adquiridos durante la carrera.
- Investigar diferentes herramientas para solventar los problemas que salgan.
- Adquirir las nociones necesarias para poder llevar proyectos.
- Conseguir manejarme en entornos de incertidumbre.

Conceptos teóricos

La parte del proyecto más importante es el proceso de lanzar una aplicación Android desde cero, para ello se ha tenido que realizar una investigación para determinar que herramientas, lenguajes de programación, entornos de desarrollo son los más apropiados para un despliegue ágil.

Todo esto se sitúa en un mercado altamente competitivo, con una gran variabilidad (usuarios, empresas, desarrolladores ...) y con un constante cambio.

3.1. Framework

Un framework (de origen anglosajón, marco de trabajo), según lo que dice la wikipedia [**?**], es una estructura conceptual y tecnológica de soporte definido, normalmente por módulos de software concretos, que sirve de base para la organización y el desarrollo de software. Las ventajas que ofrece son varias, entre las que se puede destacar:

- **Evita repetición de código:** las partes más usadas, pasan a ser algo del *core* del framework.
- **Uso de buenas prácticas:** muchos de estos están basados en patrones de diseño que nos obligan a usar.
- **Elementos avanzados integrados:** cosas complejas y que implementarlas llevaría mucho tiempo, suelen venir integradas.
- **Desarrollo ágil:** por los factores anteriores, podemos centrarnos más en la lógica de negocio de la aplicación que se desea hacer, de una manera más rápida y segura.

Por lo tanto, es necesario trabajar mediante frameworks, ya que nos garantizar una aplicación de mayor calidad, que si la hacemos desde cero, en código nativo.

Opciones disponibles

En el mercado existen muchos frameworks disponibles para el desarrollo de aplicaciones móviles, por lo que decantarse por uno no es tarea sencilla, ya que como se aprecia cada uno tiene sus pros y contras. En una encuesta realizada en un foro [?], a mediados del 2019, la opinión de los más recomendados era 3.1:

Ionic	**26**	20,97%
React Native	**37**	29,84%
Flutter	**40**	32,26%
Xamarin	**10**	8,06%
Otro...	**11**	8,87%

Votantes: **124**. Tú ya has votado en esta encuesta

Figura 3.1: Encuesta

Algo ideal es que se intento buscar un framework *crossplataform*, con el que abarcar un mayor mercado, a nivel de usuarios y dispositivos.

Ionic

Ionic [?] se basa en lenguaje de programación javascript, html y css. Internamente esta basado en otro framework, como es Angularjs. La licencia Open source, fue lanzado en el 2013, permite el desarrollo para iOS y Android. A sufrido gran cantidad de cambios desde entonces.

Ventajas:

- Una amplia comunidad de usuarios.
- Documentación extensa y de calidad.

Algunas de las desventajas:

- Rendimiento algo menor, ya que no se desarrolla de forma nativa.

- Bibliotecas en constante cambio y evolución, más que ser algo bueno, puede que deje a la aplicación fuera de versión y se tenga que hacer desde cero.

React native

React native [**?**] fue creado en 2015, de la mano de Facebook. La licencia que tiene es MIT. La diferencia que tiene con React, es que no manipula el DOM, por lo que tampoco usa HTML o CSS. Se programa en javascript. Las aplicaciones más conocidas que usan este framework es instgram o facebook.

Ventajas:

- Gran comunidad de usuarios y herramientas.
- Mejora de las funcionalidades constantemente.

Algunas de las desventajas:

- Rendimiento algo menor.
- No es código nativo, pero casi, por lo que no tiene soporte oficial de Google y Apple.

Xamarin

Xamarin [**?**] creado por Microsoft en el 2011, por lo que está desarrollado en .NET, siendo propietario.

Ventajas:

- Permite el desarrollo de iOS, Android, web pero nativas de escritorio también.
- Puede llamar a fragmentos de código usados en otras plataformas.
- Soporte para wearables.

Algunas de las desventajas:

- Acceso limitado a las bibliotecas.
- Soporte y actualizaciones lento / tarde.

- Comunidad grande pero pequeña comparada con otras.
- Aplicaciones de mayor tamaño.
- Coste.

Otros

Hay muchos otros, como kotlin, Apache Cordoba, jQuery Mobile, Native script ... Pero no me convencieron por diversas razones.

3.2. Flutter

Flutter [**?**] es un SDK de código abierto creado por Google a finales del 2018. Permite que los desarrolladores puedan crear aplicaciones para iOS, Android y web.

Este se encuentra escrito en Dart, que es un lenguaje de programación creado por Google en 2011. Este lo que hace es una mejora del lenguaje javascript, pero sin pretender sustituirlo. Es decir, ofrecer mejores resultados y alternativas para algunos problemas, siendo una herramienta mejor, para proyectos más grandes, como es el caso de flutter.

Por lo tanto Flutter es una herramienta mi nueva, con la que se pueden hacer aplicaciones comerciales para varias plataformas sin tener que programar exclusivamente para cada una de ellas. Ya que nos ofrece la compilación nativa directamente, reduciendo los costes a la hora de tener que llevar proyectos en los que sea necesario estar en los dos mercados.

Las características más importantes son :

- Desarrollo rápido de las aplicaciones. Cuando se dice rápido es porque permite *hot reload*, esta es la carga en caliente durante la fase de desarrollo. Implica que nos es necesario tener que compilar todo el código, si no aquellas partes que fueron modificadas. Lo que permite en tiempo de ejecución ver los cambios.
- Está muy optimizado, con una evolución y soporte constante.
- Es un lenguaje de programación respaldado por Google, con lo que esto conlleva. Seguridad, confianza, fiabilidad, soporte, cursos y un montón de herramientas(más de 300 apis distintas: mapas, reconocimiento facial, traductor ...).

- La integración con el sistema operativo Android es mucho más eficaz, ya que este también pertenece a Google.
- Cuando se compila, lo hace a nativo, siendo una ganancia de rendimiento.
- La curva de aprendizaje es baja, en el caso de que sepamos javascript, typescript o ecmascript.
- La calidad de las animaciones.

Las desventajas que tiene son:

- La mayor parte de la documentación se encuentra en inglés, pero es más problema de desarrollador, que de la propia documentación del framework.
- Las aplicaciones ocupan más espacio que las nativas, ya que suelen incluir el SDK al completo.
- Al ser tan nuevo, tiene algunos problemas, no ofrece todas las funcionalidades, como las que ofrecen otras plataformas. Google lo sabe y está apostando mucho por ello.
- Es una comunidad pequeña, pero a crecido en dos años más que otras, en tiempos similares.

Al final me decanto por este Framework básicamente porque es algo nuevo y disruptivo. El respaldo de Google se me presentaba como garantía, con la tranquilidad que eso conlleva. Es decir, todo el *cloud services de Google* se puede integrar con gran facilidad. Otra de las cosas que fueron de agrado es que la comunidad lo a recibido con los brazos abiertos, como se puede ver en la encuesta 3.1, se encuentra entre los favoritos de los desarolladores, por algo será.

Firebase

Firebase [**?**] es una plataforma para el desarollo de apliaciones web, Android e iOS, creada por Google en el 2014. Esta en la nube, formando el *Google Cloud Plataform*, que consta de un conjunto de herramientas para dotar de una calidad enorme a los proyectos. Ya que permite la integración del ecosistema de Google como un todo.

Fue integrada en el proyecto por el hecho de ser un servicio gratuito y que aportaba gran valor a la aplicación. Este pasa a ser de pago cuando la app tiene que escalarse a un nivel más grande, por el tema de usuarios o el número de peticiones que se realicen a la misma. Algunas de las herramientas que integra son de pago, otras no.

Por lo tanto al ser multiplatafomra es un backend con el que poder controlar todo, ganancias, gastos, escalabilidad, nuevos productos, herramientas ...

Entre los servicios que ofrece tenemos:

- **Real Time Data Base:** base de datos simples en tiempo real, si fuera necesario tener que guardar música, video o fotos, tiene otra parte que es la de Storage.
- **Crash reporting:** herramienta para el reporte de errores que se producen en la aplicación.
- **Autentication:** integración con la mayor cantidad de aplicaciones de redes sociales en las que su API permite esto. Obviamente la propia Google es una de ellas. Lo podemos ver en la siguiente imagen 3.2:

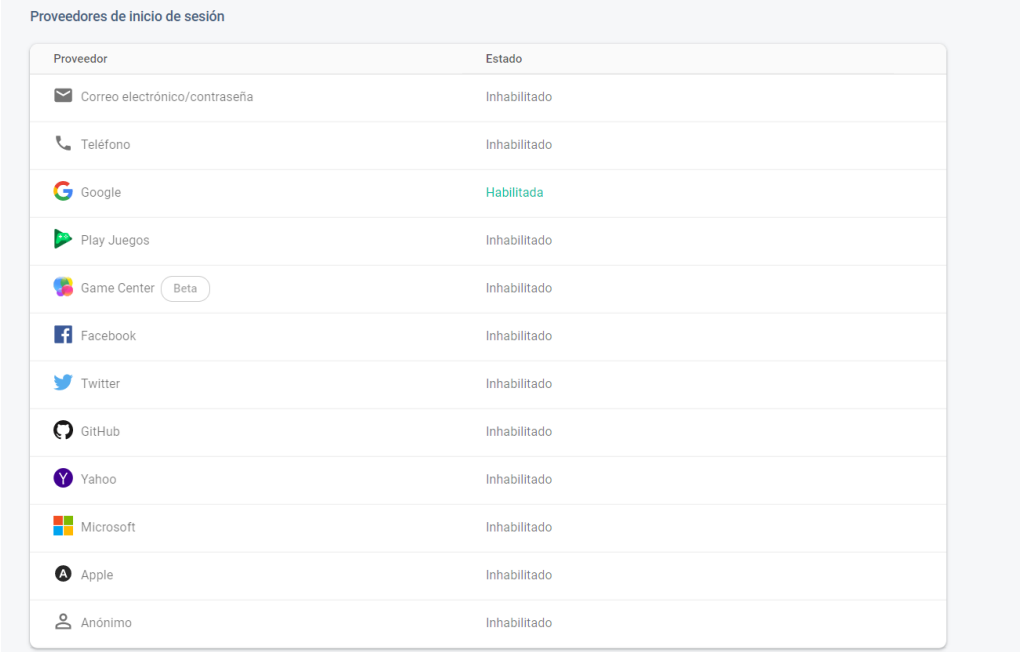

Figura 3.2: Inicio de sesión

- **Test lab:** probar la aplicación antes de realizar el despliegue de la misma.
- **Remote config:** hacer cambios internos del funcionamiento de la aplicación sin tener que recomplilar o actualizarla.
- **Hosting:** servidor donde podemos publicar una página web.
- **Análisis:** mediante las analíticas que ofrece 3.3, sirve como herramienta de toma de decisiones. Pudiendo optimizar a que mercados dirigirse mediante una estrategia de marketing.

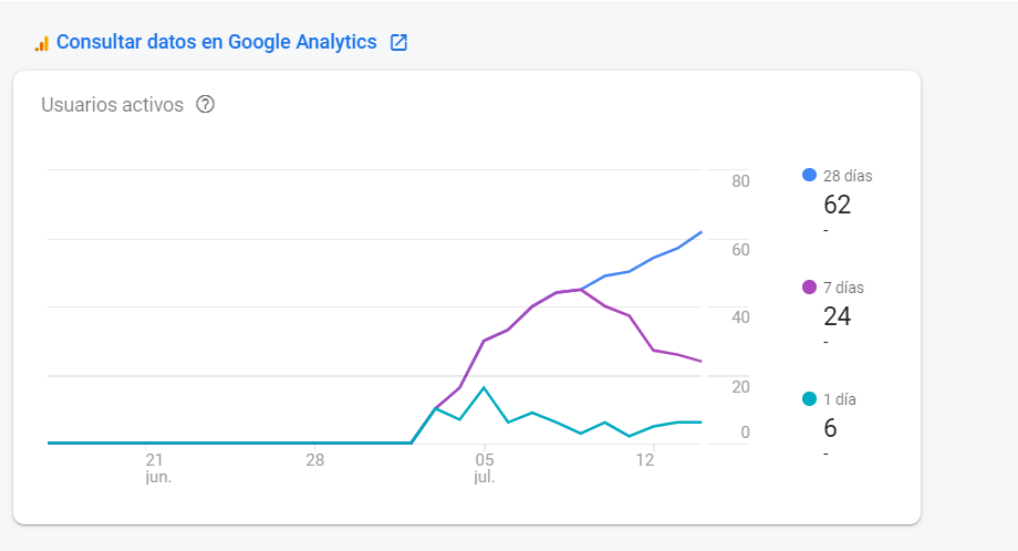

Figura 3.3: Ejemplo de Analytics

Técnicas y herramientas

Esta parte de la memoria tiene como objetivo presentar las técnicas metodológicas y las herramientas de desarrollo que se han utilizado para llevar a cabo el proyecto. Se comentará de manera breve, las diferentes opciones y la razón porque estás fueron descartadas.

4.1. Sistema Operativo

Opciones elegidas

Microsoft

Microsoft es el sistema operativo generalista. Es de sobra conocido, con sus pros y contras. Fue elegido por comodidad, ya que es el que tengo instalado en mis ordenadores. Además el IDE usado 16 para el desarrollo, será de la misma compañía, por lo que hay mayor optimización para este sistema.

Android

Andorid es un sistema operativo móvil desarrollado por Google. Está basado en el kernel de Linux. Tiene la casi toda la cuota de mercado de telefonía. Este se ha usado en mi *smartphone* personal con el fin de probar las aplicaciones en dispositivo físicos.

Alternativas descartadas

Ubuntu

Ubuntu es un sistema operativo de código abierto, distribuido bajo una licencia libre. Está basado en Debian, distribución de Linux. Se desarta porque no iba a instalar otro sistema operativo en mis ordenadores personales.

macOS X

macOs sistema operativo creado por Apple, basado en Unix. Se descarta por que se quiere trabajar con dispositivos Android. Pero en el caso de que se quiera desarrollar para iOS **??**, es indispensable usarlo aquí, porque a la hora de compilar tira de las librerías internas del sistema operativo.

iOS

macOs sistema operativo para *smartphones* creado por Apple, basado en Unix. Se descarta por que se quiere trabajar con dispositivos Android. Pero en el caso de que se quiera desarrollar para iOS **??**, es indispensable usarlo aquí, porque a la hora de compilar tira de las librerías internas del sistema operativo.

4.2. Control de versiones

Opciones elegidas

Git

Git es un software de control de versiones, pensado para trabajar con gran cantidad de archivos, con el fin de llevar el registro de los cambios y coordinar a las personas que los comparten. Es gratuito y de código abierto.

Me he decantado por él, porque lo usé durante las prácticas curriculares y extracurriculares de forma intensa, mediante la herramienta Git Bash, ya que mediante comando me siento más cómodo que con una interfaz. Aunque también la dispone mediante el comando *gitk*.

GitHub

Github es una plataforma web, recientemente comprada por Microsoft, usada para el control de versiones con las funciones de Git. Entre las

diferentes herramientas a destacar: wiki para cada uno de los proyectos, gráficos, funcionalidades de red social, gestor de proyectos, entre otras.

Se escogió esta porque la hemos usado durante el grado y es de sobra conocida. Además ofrece la posibilidad de integración con la herramienta Zenhub 15, para la gestión del proyecto, teniendo las dos cosas centralizadas en el mismo lugar, lo que facilita el proceso de desarrollo.

Alternativas descartadas

Gitlab

Gitlabes igual que Github pero de código abierto ya que tiene licencia MIT. También lo usé en la empresa durante las prácticas pero fue descartado porque me parece de menor calidad. En cuanto a espacio este tiene 10 GB a favor, en contra del 1 GB de Github.

Bitbucket

Bitbucket es otra web más para el control de versiones. Esta enfocado más a la empresa privada, ya que se suele integrar muy bien con otras herramientas de gestión de proyectos. Se descarto por el poco uso que he tenido con este.

Extensiones Visual Studio Code

Hay una gran cantidad de herramientas para el control de versiones en la tienda de este editor de código, puede ser práctico, pero las interfaces pueden ser liosas. Por eso me gusta más mediante comando, descartando rápidamente esta opción.

4.3. Gestión de proyecto

Opción elegida

Zenhub

Zenhub es una herramienta de gestión de proyectos, que viene por defecto integrada en Github 14, lo que implica a usarla sin pensarlo mucho. En mi caso lo que más usé fue el tablero de *kanban* donde poder ver las *issues* que me planificaba para cada *sprint*. Ofrece diferentes tipos de gráficos, el que mejor se encajaba a la planificación fue de *burndown*, ya que me permite ver

lo ideal del proyecto y la progresión que llevo. No me gustó que las tareas las cierra por días en vez de por horas.

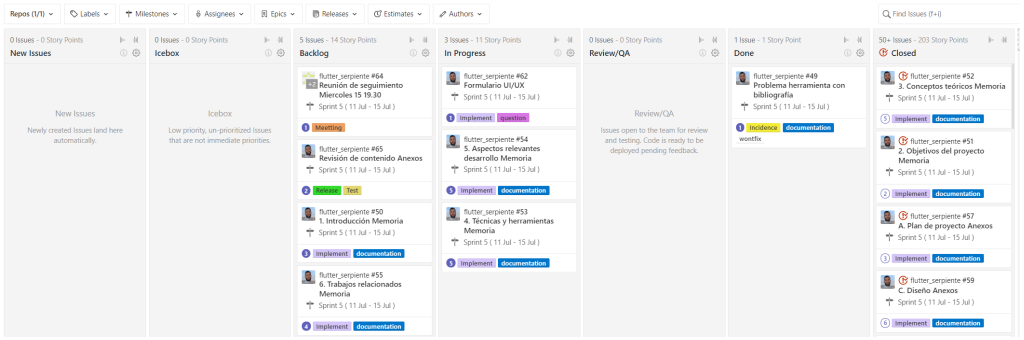

Figura 4.4: Herramienta Zenhub

Alternativas descartadas

Jira

Jira es una herramienta web propietaria, para el control, seguimiento de errores e incidencias, dentro de la gestión de proyectos. La conozco porque la usé durante las prácticas curriculares, me gustaba mucho, pero no me parece cómoda para lo que quería hacer. Ya que esta herramienta esta orientada en la mejora de procesos en la empresa, que al desarrollo de software.

Trello

Trello es una herramienta de gestión de proyectos, con interfaz móvil y web. Usa el sistema kanban como Zenhub 15 y tiene integración con Github 14, pero me parece que no encaja en proyectos unipersonales, como era mi caso. Ya que el enfoque es más colaborativo, con grandes grupos de trabajo, donde es necesario compartir documentos con los requisitos, etc ...

4.4. Entorno de desarrollo integrado (IDE)

Opción elegida

Visual Studio Code

Visual Studio Code es un editor de código fuente desarrollado por Microsoft, cuya licencia es MIT. Esta es totalmente gratuita, con una

cantidad enorme de extensiones o *pluggins*, que minimizan esfuerzos, además de la integración con git, que permite ver los cambios en tiempo real.

Esta facilidad de uso hicieron que me decantase por él.

Alternativas descartadas

Android Studio

Android Studio es la herramienta oficial de desarrollo de aplicaciones móviles para el sistema operativo Android. La licencia es Apache 2.0. y sustituye a Eclipse 17 como el entorno de desarrollo preferido por Google.

Este IDE se descarta para la creación de código, pero no para las máquinas virtuales.

Eclipse

Eclipse es la plataforma software compuesta de varias herramientas de programación de código abierto. Se descartó por ser un IDE no recomendado por parte de Google. Además que a mi parecer es algo tosco.

4.5. Entorno de virtualización

Opción elegida

Visual Studio Code

Android Studio 17 fue elegido para el despliegue de la aplicación en las máquinas virtuales que ofrece este IDE. Es eficiente y simple, centrado en Android. La creación de las VM es muy cómoda, ya que tiene todo integrado, (con cuatro clicks de ratón se puede lanzar una).

Sumado a que algunas herramientas de Visual Studio Code tiran de estas máquinas, hace que sea la opción más recomendable.

Alternativas descartadas

Virtual Box

Virtual Box es un software de virtualización desarrollado por Oracle. Es de sobra conocido, ya que lo hemos usado durante el grado. Puede desplegar Android, pero al no estar integrado en el desarrollo de aplicaciones, se descató.

Aspectos relevantes del desarrollo del proyecto

Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, comentados por los autores del mismo. Debe incluir desde la exposición del ciclo de vida utilizado, hasta los detalles de mayor relevancia de las fases de análisis, diseño e implementación. Se busca que no sea una mera operación de copiar y pegar diagramas y extractos del código fuente, sino que realmente se justifiquen los caminos de solución que se han tomado, especialmente aquellos que no sean triviales. Puede ser el lugar más adecuado para documentar los aspectos más interesantes del diseño y de la implementación, con un mayor hincapié en aspectos tales como el tipo de arquitectura elegido, los índices de las tablas de la base de datos, normalización y desnormalización, distribución en ficheros3, reglas de negocio dentro de las bases de datos (EDVHV GH GDWRV DFWLYDV), aspectos de desarrollo relacionados con el WWW... Este apartado, debe convertirse en el resumen de la experiencia práctica del proyecto, y por sí mismo justifica que la memoria se convierta en un documento útil, fuente de referencia para los autores, los tutores y futuros alumnos.

Trabajos relacionados

Este apartado sería parecido a un estado del arte de una tesis o tesina. En un trabajo final grado no parece obligada su presencia, aunque se puede dejar a juicio del tutor el incluir un pequeño resumen comentado de los trabajos y proyectos ya realizados en el campo del proyecto en curso.

Conclusiones y Líneas de trabajo futuras

Todo proyecto debe incluir las conclusiones que se derivan de su desarrollo. Éstas pueden ser de diferente índole, dependiendo de la tipología del proyecto, pero normalmente van a estar presentes un conjunto de conclusiones relacionadas con los resultados del proyecto y un conjunto de conclusiones técnicas. Además, resulta muy útil realizar un informe crítico indicando cómo se puede mejorar el proyecto, o cómo se puede continuar trabajando en la línea del proyecto realizado.

Bibliografía