

UNIVERSIDAD MIGUEL HERNÁNDEZ DE ELCHE

ESCUELA POLITÉCNICA SUPERIOR DE ELCHE

GRADO EN INGENIERÍA INFORMÁTICA EN  
TECNOLOGÍAS DE LA INFORMACIÓN



**UNIVERSITAS**  
*Miguel Hernández*



**Biblioteca Digital**  
UNIVERSITAS Miguel Hernández

"JIRA APP: PROYECTO DE  
MANIPULACIÓN DE JIRA A TRAVÉS DE  
APP CON DART-FLUTTER"

TRABAJO FIN DE GRADO

Junio -2022

AUTOR: Dmitry Romanyutenko

DIRECTOR/ES: Antonio Peñalver Benavent



# AGRADECIMIENTOS Y DEDICATORIAS

Me gustaría agradecer a mi tutor Antonio Peñalver Benavent por su ayuda ofrecida durante el desarrollo de este Trabajo de Fin de Grado, dedicándome el tiempo y los conejos muy útiles relacionados al trabajo y los estudios que surgieron durante el proceso.

Agradecer también al equipo de NTT Data que han estado pendiente de mi desarrollo de la aplicación en todo el momento. Me han proporcionado el material de manera gratuita (Por cuenta personal sale bastante caro ya que es la plataforma Udemy [1]) muy valioso para conocer y aprender nuevas tecnologías. Por ver cómo funciona la aplicación y ofrecer sus sugerencias y comentarios para mejorarla.

Especial agradecimiento quiero dedicar a un empleado de NTT Data José Francisco Vera Girona que me ayudó en cuanto al código de la aplicación, dando soluciones y consejos útiles durante el desarrollo de la misma.

Y por último agradecer a toda la comunidad de Flutter, ya que al ser un proyecto open source la realización de este proyecto habría sido imposible sin el trabajo de los desarrolladores que aportan su granito de arena para mejorar este framework.



# RESUMEN

Este documento contiene el desarrollo de un proyecto de software, desde su planteamiento inicial hasta su implementación final, pasando por las distintas fases del desarrollo de éste.

El propósito de este proyecto es el desarrollo de una aplicación multiplataforma para dispositivos móviles Android la cual ofrezca la posibilidad de realizar algunas operaciones de la página web de JIRA [2] sin entrar a la página web. Con tan sólo entrar a la aplicación con los datos de la empresa se evita entrar al navegador y manejarlo desde ahí.

Las funciones más comunes son:

- Ver los datos del empleado, ver todos los proyectos, ver todas las tareas de cada proyecto, ver tareas asignadas al empleado de un proyecto y ver detalles de cada subtarea.
- Modificar el estado de una subtarea.
- Añadir el tiempo dedicado a una subtarea.
- Crear una subtarea.

La aplicación necesita tener una conexión a internet para poder funcionar, ya que se realizan peticiones HTTP a la API de la plataforma. Por tanto, para previamente crear una operación, hay que probarla desde un programa que se verá más adelante.

El desarrollo completo estará expuesto en este documento, así como las herramientas y tecnologías utilizadas, haciendo hincapié en Flutter ya que este proyecto es la primera vez que se utiliza dicha tecnología.



# ÍNDICE

<b>INTRODUCCIÓN</b>	<b>17</b>
1.1 INTRODUCCIÓN E IMPORTANCIA	17
1.2 ELECCIÓN DEL PROYECTO	18
1.3 QUE SE PRETENDE CONSEGUIR	18
1.4 QUE NO SE PRETENDE CONSEGUIR	19
1.5 NECESIDAD	20
<b>ESTADO DE LA CUESTIÓN</b>	<b>22</b>
2.1 ANTECEDENTES	22
2.2 HERRAMIENTAS PARA LA GESTIÓN DE PROYECTOS	23
2.2.1 JIRA CLOUD BY ATLISSIAN	23
2.2.2 JIRA DATA CENTER AND SERVER	24
2.3 CONCLUSIÓN	24
<b>OBJETIVOS DEL PROYECTO</b>	<b>27</b>
3.1 OBJETIVOS GENERALES	27
3.2 OBJETIVOS TÉCNICOS	27
3.3 OBJETIVOS DIDÁCTICOS	28
3.4 OBJETIVOS PERSONALES	28
<b>HIPÓTESIS DE TRABAJO</b>	<b>30</b>
4.1 DART	31
4.2 FLUTTER	32
4.2.1 PAQUETES DE FLUTTER INTERNOS Y EXTERNOS	37
4.2.2 VENTAJAS DE FLUTTER	42

4.2.3	DESVENTAJAS DE FLUTTER	46
4.2.4	ALTERNATIVAS PARA EL DESARROLLO HÍBRIDO	47
4.2.4.1	XAMARIN [22]	47
4.2.4.2	APACHE CORDOVA (PHONEGAP) [23]	48
4.2.4.3	REACT NAVITE [24]	49
4.2.5	CONCLUSIÓN	49
4.3	LUCID	50
4.4	API & JSON	50
4.5	POSTMAN	52
4.6	QUICKTYPE.IO	53
4.7	VISUAL STUDIO CODE	54
4.8	GIT & GITHUB	55
4.9	ANDROID STUDIO	57
4.10	EQUIPOS FÍSICOS	59
4.10.1	EQUIPO DE SOBREMESA	59
4.10.2	EQUIPO PORTÁTIL [35]	60
4.10.3	DISPOSITIVO MÓVIL FÍSICO [36]	60
4.10.4	DISPOSITIVO MÓVIL EMULADO [37]	61
<b>METODOLOGÍA Y RESULTADOS</b>		<b>63</b>
5.1	CICLO DE VIDA DEL PROYECTO	63
5.1.1	REQUERIMIENTOS	64
5.1.2	ESPECIFICACIÓN Y DISEÑO	65
5.1.3	IMPLEMENTACIÓN Y PRUEBAS UNITARIAS	72
5.1.4	VALIDACIÓN Y PRUEBAS	73
5.1.5	IMPLANTACIÓN, USO Y MATENIMIENTO	74
5.2	DIAGRAMA DE GANTT	74
5.3	ACTORES Y CASOS DE USO	75
5.3.1	EMPLEADO	77
5.3.2	USUARIO NO IDENTIFICADO	94
5.4	PROTOTIPOS	96
5.4.1	WIREFRAME	96
5.4.2	RESULTADO FINAL	98
5.4.3	COMPARATIVA	98

5.5	PROBLEMAS	105
5.6	EJEMPLO FUNCIONAL	106
<b>CONCLUSIONES Y PROPUESTAS</b>		<b>112</b>
6.1	CONCLUSIONES	112
6.2	PROPUESTAS DE FUTURO	113
<b>BIBLIOGRAFÍA</b>		<b>115</b>
7.1	FUENTES BIBLIOGRÁFICAS	115
<b>BIBLIOGRAFÍA</b>		<b>115</b>





# ÍNDICE DE FIGURAS

Ilustración 1 - Interfaces en la aplicación móvil Jira Cloud by Atlassian mostrando sus pantallas .....	23
Ilustración 2 - Interfaces en la aplicación móvil Jira Cloud by Atlassian mostrando sus pantallas .....	24
Ilustración 3 - Logotipo de Dart.....	31
Ilustración 4 - Logotipo de Flutter .....	32
Ilustración 5 - Fragmento de código en Flutter que muestra en una fila dos textos, cada uno con diferente formato .....	33
Ilustración 6 - Demostración del ejemplo del código de la ilustración anterior .	34
Ilustración 7 - Arquitectura Flutter .....	34
Ilustración 8 - Menú de depuración en Flutter .....	35
Ilustración 9 - Esqueleto de proyecto .....	36
Ilustración 10 – Repositorio de Flutter - Importación de paquetes .....	38
Ilustración 11 - Archivo de las dependencias (Paquetes).....	39
Ilustración 12 - Iconos de paquete de Cupertino Icons.....	39
Ilustración 13 - Lista de llamas a provider .....	40
Ilustración 14 - Ejemplo de uso de Flutter Secure Storage - Lectura de variables .....	40
Ilustración 15 - Caso real de uso de paquete, en caso fallido de petición imprimir los datos de error .....	41
Ilustración 16 - Código ilustrativo que abre el navegador con el url especificado .....	42
Ilustración 17 - Elemento de carga "discreteCircular" .....	42
Ilustración 18 - Representación en árbol de los widgets entre otros .....	43
Ilustración 19 - Los frameworks más queridos de 2020, fuente: Stack Overflow 2020 Developer Survey .....	44
Ilustración 20 - Logotipo de Apache Cordova .....	47
Ilustración 21 - Logotipo de Apache Cordova .....	48

Ilustración 22 - Logotipo de React Native.....	49
Ilustración 23 - Funcionamiento de la api. Fuente: <a href="https://appmaster.io/es">https://appmaster.io/es</a> .....	51
Ilustración 24 - Ejemplo de un .json. Fuente: <a href="https://www.hostinger.es/">https://www.hostinger.es/</a> .....	52
Ilustración 25 - Entorno de Postman .....	52
Ilustración 26 - Conversión de JSON (Izquierda) a una clase (Derecha)).....	54
Ilustración 27 - Interfaz de usuario de Visual Studio Code.....	55
Ilustración 28 - Interfaz del sitio web de Github.....	56
Ilustración 29 - Interfaz de Sourcetree.....	57
Ilustración 30 - Lista de dispositivos que se puede crear para emular. Fuente: <a href="https://developer.android.com/">https://developer.android.com/</a> .....	58
Ilustración 31 - Emulando el dispositivo Pixel 5.....	59
Ilustración 32 - Ciclo de vida en cascada mejorado .....	64
Ilustración 33 - El proceso de desarrollo del proyecto utilizando diagrama de Gantt .....	75
Ilustración 34 - Casos de uso del empleado.....	78
Ilustración 35 - Diagrama de secuencia C-01: Carga de la pantalla "Perfil".....	80
Ilustración 36 - Diagrama de secuencia C-02: Carga de la pantalla "Proyecto" .....	82
Ilustración 37 - Diagrama de secuencia C-03: Cambio de la lista de tareas.....	84
Ilustración 38 - Diagrama de secuencia C-04: Carga de la pantalla "Tarea" .....	86
Ilustración 39 - Diagrama de secuencia C-05: Crear nueva subtarea .....	88
Ilustración 40 - Diagrama de secuencia C-06: Carga de la pantalla "Subtarea" .....	90
Ilustración 41 - Diagrama de secuencia C-07: Añadir tiempo .....	92
Ilustración 42 - Diagrama de secuencia C-08: Cambiar estado.....	94
Ilustración 43 - Caso de uso del usuario no identificado.....	95
Ilustración 44 - Diagrama de secuencia C-09: Acceso web .....	96
Ilustración 45 - Pantallas Wireframe.....	97
Ilustración 46 - Pantalla de login.....	98
Ilustración 47 - Pantalla de carga.....	99
Ilustración 48 - Pantalla "Perfil" .....	100
Ilustración 49 - Pantalla "Proyecto" .....	101
Ilustración 50 - Pantalla "Tarea" .....	102

Ilustración 51 - Pantalla de crear subtarea .....	103
Ilustración 52 - Pantalla "Subtarea" .....	104
Ilustración 53 - Pantalla "Añadir Tiempo" .....	104
Ilustración 54 - Pantalla "Cambiar Estado" .....	105
Ilustración 55 - Captura de pantalla de la pantalla de inicio de sesión .....	107
Ilustración 56 - Captura de pantalla "Proyecto" .....	108
Ilustración 57 - Captura de la pantalla de nueva subtarea .....	109
Ilustración 58 - Captura de la pantalla de agregar el tiempo .....	110





# ÍNDICE DE TABLAS

Tabla 1 - Plantilla de actores.....	76
Tabla 2 - Plantilla de casos de uso.....	76
Tabla 3 - Actor empleado .....	77
Tabla 4 - Caso de uso C-01: Carga de la pantalla "Perfil" .....	79
Tabla 5 - Caso de uso C-02: Carga de la pantalla "Proyecto" .....	81
Tabla 6 - Caso de uso C-03: Cambio de la lista de tareas.....	83
Tabla 7 - Caso de uso C-04: Carga de la pantalla "Tarea" .....	85
Tabla 8 - Caso de uso C-05: Crear nueva subtarea.....	87
Tabla 9 - Caso de uso C-06: Carga de la pantalla "Subtarea" .....	89
Tabla 10 - Caso de uso C-07: Añadir tiempo.....	91
Tabla 11 - Caso de uso C-08: Cambiar estado.....	93
Tabla 12 - Actor usuario no identificado .....	94
Tabla 13 - Caso de uso C-09: Acceso web.....	95





# CAPÍTULO 1

## INTRODUCCIÓN

---

### 1.1 INTRODUCCIÓN E IMPORTANCIA

En la empresa tan grande como NTT Data [3] con más de 140 mil de empleados en más de 50 países, hay que dividir el trabajo en muchas partes para que con cada una de ellas trabaje una o varias personas. Para poder seguir de alguna manera los proyectos, tareas y subtareas tiene que haber alguna herramienta. En la herramienta cada empleado tiene que poder leer, cambiar, crear y eliminar datos de un proyecto, tarea o subtarea. Durante el desarrollo del proyecto hemos estado utilizando la herramienta web para registrar los datos relacionados con el desarrollo.

Es una herramienta interna de la empresa la cual solamente puede usar un empleado que tenga una cuenta creada en la página web. La aplicación viene a resolver el problema de sólo trabajar con la web desde un navegador, de aquí ya vienen dos opciones ya que no solamente desde la web se podrá dar el uso a la herramienta.

Aunque casi todos los empleados trabajan en un ordenador y no hay una excesiva necesidad de realizar el proceso desde el teléfono móvil, el objetivo de este proyecto es ofrecer la posibilidad de sí poder hacerlo desde un dispositivo Android, de la manera que se va a ahorrar el tiempo en encender el ordenador, entrar a la página web y registrar las horas trabajadas y además poder hacerlo desde cualquier lugar.

## **1.2 ELECCIÓN DEL PROYECTO**

La motivación de este proyecto surge desde mi gran interés a las tecnologías como Java y programación orientada a objetos (POO). Aunque no trabaje en Java, Dart [4] (Con framework Flutter) es un lenguaje muy similar que permite disfrutar del desarrollo de igual manera como si estuviera desarrollando en Java con POO.

Además, cuando se ha conocido el framework Flutter, éste ha mostrado sus fuertes características frente a otros frameworks de libre licencia en el mercado laboral. Es un framework que puede competir con los mejores del mundo del desarrollo del software para dispositivos móviles.

Además, se suma el reto motivador de aprender a desarrollar con una tecnología que no se había estudiado con anterioridad y que puede abrir las puertas a un futuro empleo posible.

Por último, debido a la primera experiencia en una empresa informática, más motiva obtener una experiencia muy valiosa tanto personalmente como para un puntazo en la experiencia laboral en un CV.

## **1.3 QUE SE PRETENDE CONSEGUIR**

Se pretende desarrollar una aplicación híbrida utilizando lenguaje de programación Dart y framework Flutter que permitan la realización de algunas operaciones de Jira a través de un dispositivo móvil con sistema operativo Android.

El usuario que pertenece al grupo de empleados de la empresa puede entrar a la aplicación para ver los proyectos asignados a él, dentro de cada proyecto puede observar las tareas que tenga asignadas a él y dentro de cada tarea puede ver, editar y crear las subtareas.

El usuario lo va a hacer mediante una interfaz sencilla que permita ser utilizada de forma ágil durante la instancia en la aplicación.

Como objetivo secundario se incluye el aprendizaje de distintas herramientas para el desarrollo de aplicaciones, tanto aquellas utilizadas para la codificación de la aplicación como a otro tipo de herramientas utilizadas durante el planteamiento del proyecto.

Por último, también se contempla la posibilidad de que este trabajo sirva como marco de referencia para trabajos de futuros empleados de la empresa.

## **1.4 QUE NO SE PRETENDE CONSEGUIR**

A pesar de que esta aplicación trabaja solamente con la sub-web de Jira de Atlassian, se puede modificarla y poder usar para cualquier web de Jira y eso no se busca debido a que la aplicación pertenece 100% a la empresa. Por tanto, el código y las ideas no pueden salir fuera del ámbito de la empresa, excepto para redacción del TFG.

Tampoco se pretende probar la aplicación final en dispositivos iOS, Windows y más SO por la falta del tiempo y porque no es el objetivo propuesto por la empresa.

## 1.5 NECESIDAD

Jira de Atlassian es usada a nivel mundial. Muchísimas empresas están dando uso a dicha herramienta para agilizar y facilitar el desarrollo de un proyecto. La gran ventaja es que puede dividir las tareas en subtareas y asimismo asignar a cada tarea o subtarea a una o varias personas. Cada persona asignada a un proyecto puede ver todas las tareas y asignaciones por lo que provoca una gran visibilidad del proyecto. El flujo de trabajo se acelera gracias a que no hay que estar preguntando el rol o qué tareas tienes que realizar el empleado, es altamente visible, y en caso de cualquier duda, se puede ver el nombre del responsable de una tarea.

Dicha herramienta puede usar tanto dos personas para un micro desarrollo de un proyecto como una grande empresa con miles de empleados y cientos de proyectos. Y cuando hay tanta demanda de una herramienta a este nivel, surge la necesidad de agilizar aún más el tiempo, en este caso, poder acceder a dicha herramienta con un dispositivo móvil. Ahora mismo el 68% [5] de población mundial cuenta con un dispositivo móvil.



# CAPÍTULO 2

## ESTADO DE LA CUESTIÓN

---

### 2.1 ANTECEDENTES

El trabajo de fin de grado se realiza en la empresa cuyo nombre es NTT DATA [3]. Es una compañía que proporciona una gama amplia de servicios en Europa y América. Incluye consultoría, integración de sistema y subcontratación.

Es una colaboración entre la universidad de Miguel Hernández y la empresa para realizar el trabajo de fin de grado y una práctica externa al mismo tiempo. Y en esta ocasión, el objetivo de desarrollo es pasar algunas de las funcionalidades de la página web de Jira de Atlassian a una aplicación para dispositivos móviles con sistema operativo Android.

NTT DATA ha creado su propio entorno de Jira, con el diseño algo modificado que difiere del Jira Software. Algunas de las funcionalidades han sido eliminadas, y algunas nuevas han sido añadidas. Además, han agregado plugins para ciertos fines concretos.

Jira es una herramienta online para administrar las tareas de un proyecto de cualquier especialidad. En nuestro caso se usa dicha herramienta para planificar el desarrollo de la aplicación y registrar las horas que se han invertido en el proyecto.

## 2.2 HERRAMIENTAS PARA LA GESTIÓN DE PROYECTOS

Existe una herramienta propia de Atlassian [6] que permite directamente manejar proyectos creados en Jira. La diferencia consiste en que el Jira que usa la empresa tiene un formato diferente, no contempla de todas las funciones de Jira y además tiene más funciones añadidas. Existe otra aplicación [7] también de Atlassian algo diferente en cuanto al diseño, a continuación, vienen los ejemplos.

### 2.2.1 JIRA CLOUD BY ATLASSIAN

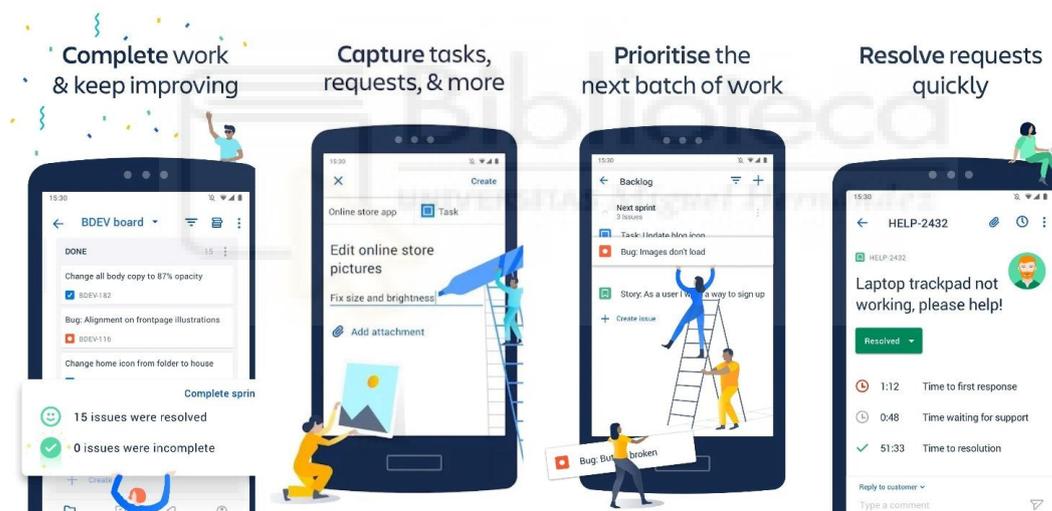


Ilustración 1 - Interfaces en la aplicación móvil Jira Cloud by Atlassian mostrando sus pantallas

La interfaz que utiliza propia aplicación de Jira es diferente pero no tan lejana a la que hemos estado desarrollando debido a que no tiene que contemplar todas las funcionalidades y repetir el diseño. Los nombres de los proyectos, tareas y subtareas y más elementos se ven de manera muy similar debido a que se extraen los datos de la API. En nuestra aplicación no está el menú con las

diferentes opciones ya que el usuario tiene que navegar de otra manera (Puede parecer incluso más intuitivo). Otras partes de la pantalla son parecidas ya que en la mayoría de las aplicaciones de este tipo son se ven igual.

## 2.2.2 JIRA DATA CENTER AND SERVER

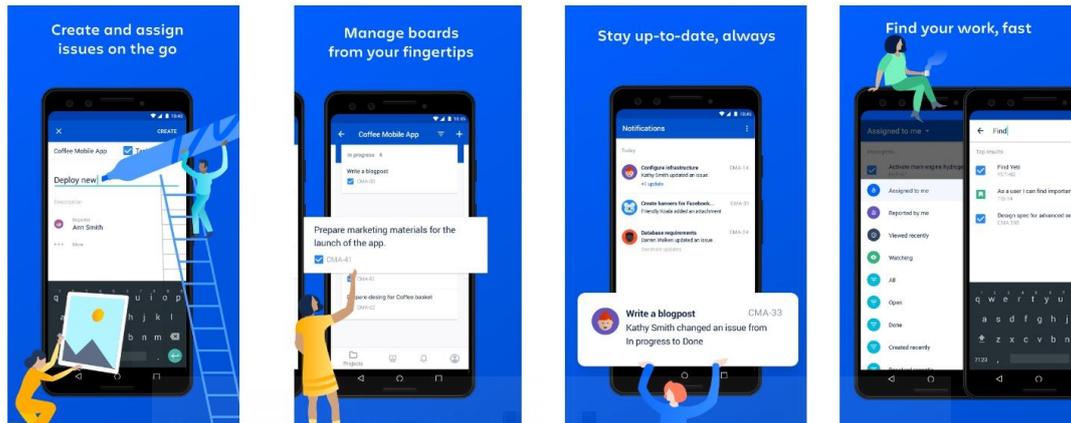


Ilustración 2 - Interfaces en la aplicación móvil Jira Cloud by Atlassian mostrando sus pantallas

Segunda aplicación de Atlassian está centrada en los centros de datos y servidores, es decir, tiene un propósito algo distinto al nuestro. Sin embargo, la interfaz es parecida tanto a la otra aplicación como a la nuestra. Tiene diferentes funciones y sólo algunas comunes. El color de esta aplicación es azul oscuro, detalle que la otra tenía un color blanco, en nuestro caso como la web tiene un color diferente, pues lo hemos usado ya que es un color corporativo.

## 2.3 CONCLUSIÓN

De lo expuesto anteriormente, podemos deducir que hay una necesidad en la empresa en la aplicación ya que el mercado laboral no cubre dicha necesidad. Para que la aplicación desarrollada en este proyecto pueda cubrir esa necesidad,

debe ser accesible para cualquier usuario independientemente del dispositivo que utilice.

En España la mayoría de los dispositivos móviles utilizan Android o iOS como sistema operativo [5], por lo que si se pretende que esta aplicación tenga utilidad en la empresa debería estar disponible en ambos mercados. Puesto que hacer dos desarrollos independientes para cada plataforma sería excesivamente costoso, se ha decidido realizar un desarrollo híbrido en Flutter. Aunque en esta ocasión sólo se trabaja con Android por ahora.





# CAPÍTULO 3

## OBJETIVOS DEL PROYECTO

---

### 3.1 OBJETIVOS GENERALES

- Desarrollar una aplicación híbrida mediante la cual se pueda observar el estado y gestionar las tareas de un proyecto, cubriendo la necesidad de la empresa.
- Diseñar una interfaz limpia e intuitiva que permita ser utilizada durante el uso de la aplicación para lograr objetivos propuestos al entrar en la aplicación.

### 3.2 OBJETIVOS TÉCNICOS

- Desarrollar el proyecto utilizando técnicas de diseño y programación que faciliten el mantenimiento de código.
- Utilizar, en la medida de lo posible, herramientas de software libre o código abierto.
- Que la aplicación final no presente fallos de funcionamiento.
- Que la empresa quede satisfecha con el trabajo realizado.

### **3.3 OBJETIVOS DIDÁCTICOS**

- Realizar una comparativa entre los diferentes frameworks para el desarrollo de aplicaciones híbridas.
- Reunir información tanto de las herramientas utilizadas para el desarrollo de este proyecto como de posibles alternativas.
- Obtener una documentación clara y entendible por el público objetivo.
- Explicar los problemas encontrados durante el desarrollo del proyecto y las soluciones implementadas.
- Presentar un trabajo que pueda servir como partida de seguir desarrollando la aplicación para futuros empleados asignados de la empresa.

### **3.4 OBJETIVOS PERSONALES**

- Aprender a trabajar con el framework Flutter.
- Realizar un proyecto completo, desde su fase inicial hasta su implantación final, adquiriendo así experiencia de gran utilidad para el futuro.
- Adquirir muchos conocimientos en la empresa a la hora de crear y llevar a cabo un proyecto.



# CAPÍTULO 4

## HIPÓTESIS DE TRABAJO

---

Una vez planteados los objetivos del proyecto, debemos mencionar las herramientas que se han utilizado para el desarrollo de este.

Desde el nacimiento del proyecto se decidió que la aplicación móvil se desarrollaría en Flutter. Esto es debido a diversos factores que veremos más adelante, pero principalmente a lo ágil que es el desarrollo en comparación con otros frameworks de desarrollo de aplicaciones híbridas y al fuerte apoyo de su comunidad.

Puesto que la aplicación está pensada para obtener datos de la API de Jira, por tanto, la información será igual para todos los empleados, excepto los datos temporales de un usuario logueado que se almacenan de forma cifrada en el dispositivo móvil que lo veremos más adelante. La información se recibe mediante la petición HTTP y se almacena en la memoria del dispositivo para mostrarla al usuario, lo mismo ocurre con enviar una petición, tras el envío recibimos una respuesta.

En todo el momento se utiliza el lenguaje de programación Dart con la ayuda del framework Flutter, de esta manera evitamos el uso de otros lenguajes. Sin embargo, al no usar otros lenguajes, hay que conocer los widgets (herramientas) y los plugins que ofrece Flutter. Existe página de repositorio [8] de paquetes de Dart & Flutter que ofrece al programador librerías que no vienen preinstaladas, son tanto de propio desarrollador del lenguaje como de la comunidad.

Durante el desarrollo de la aplicación se utilizará el software Postman que permite obtener la respuesta de una petición HTTP y de esta manera analizar qué contenido de la API hay que extraer y cual no.

Para no perder en ningún momento ninguna versión de programa que se va a necesitar ya sea porque al hacer un cambio no funciona algo o ya sea para ver cómo antes estaba implementada esa parte del código. Se usará Sourcetree que permite subir cada versión con sus notas a la plataforma GitHub, que es un repositorio de todas las versiones de la misma.

## 4.1 DART

El desarrollo de la aplicación se realiza en el lenguaje de programación Dart [9]. Dart es un lenguaje de código abierto desarrollado por Google con el objetivo de permitir a los desarrolladores utilizar un lenguaje orientado a objetos y, es bastante parecido a Java, JavaScript y C++.



Ilustración 3 - Logotipo de Dart

Tiene su propio gestor de paquetes, un compilador, un analizador y formateador. Dart permite compilar y ejecutar el código en función de milisegundos gracias a compilación *Just-in-time*.

No tiene necesidad de un entorno específico, puede usarse en todos, en nuestro caso hemos utilizado Visual Studio Code. En caso de desarrollar en la web, el código se transpila a JavaScript de forma automática. Tiene un gran apoyo para la asincronía que permite ejecutar el código mientras se está ejecutando otro código.

Dart es de propiedad de Google que lo distribuye de forma gratuita.

## 4.2 FLUTTER



Ilustración 4 - Logotipo de Flutter

Flutter [10] es un Framework híbrido de Google que utiliza el lenguaje de programación Dart para crear aplicaciones con un único código para diferentes plataformas. A diferencia de otros lenguajes multiplataformas, Flutter dispone de sus propios componentes nativos llamados Widgets. Gracias a esto, la aplicación se verá igual en todos los tipos de dispositivos, ya sea Android, iOS, Windows, etc. Se puede olvidar de la preocupación de la visión de la aplicación en otros o antiguos dispositivos.

Los widgets son clases propias de Flutter que proporcionan gran variedad de propiedades y métodos para el desarrollo.

```

Row(
  children: [
    const Text(
      '${EnglishTranslation.email}: ',
      style: TextStyle(
        color: Colors.black54,
        fontSize: 16,
        fontWeight: FontWeight.w500,
        overflow: TextOverflow.fade
      ) // TextStyle
    ), // Text
    Text(
      profile.emailAddress,
      maxLines: 1,
      overflow: TextOverflow.ellipsis
    ) // Text
  ]
) // Row

```

Ilustración 5 - Fragmento de código en Flutter que muestra en una fila dos textos, cada uno con diferente formato

En la ilustración 5 se puede observar un ejemplo de código. Se puede ver que todo está envuelto en un widget Row, y éste tiene dentro en una lista "children" dos hijos que son widgets de tipo Text. Un widget no es más que una clase que tiene sus atributos y sus métodos. Sin embargo, en función el widget que vamos escogiendo, será pintado de una o de otra manera, es decir, hay mucha variedad de widgets que cumplen la misma función, pero, sin embargo, cambia la apariencia en la aplicación final. Primer widget que es "const Text" tiene como primer parámetro obligatorio un texto, y después vienen los parámetros opcionales como viene dado en el ejemplo "style". Dentro del style definimos una clase anónima con sus propios atributos. Y dentro de cada atributo aparece una clase anónima. De esta manera se hace uso de muchas clases anónimas y por tanto no hace falta definir las en ninguna parte del código.

Full name: DMITRY ROMANYUTENKO

Email: dmitry.romanyutenko.st@everis.nttdata.com

Ilustración 6 - Demostración del ejemplo del código de la ilustración anterior

En la ilustración de arriba aparece el resultado del código de la ilustración 5. Flutter de forma interna tiene definida una serie de reglas y patrones de visualización, y el desarrollador tan solo tiene que cambiar atributos.

SDK de Flutter es completo, incluye motor de renderizado, los widgets ya vienen listos para usar, no hace falta inventar nuevo widget ya que se puede en un juego de columna/fila dentro de un contenedor crear tu propio widget.

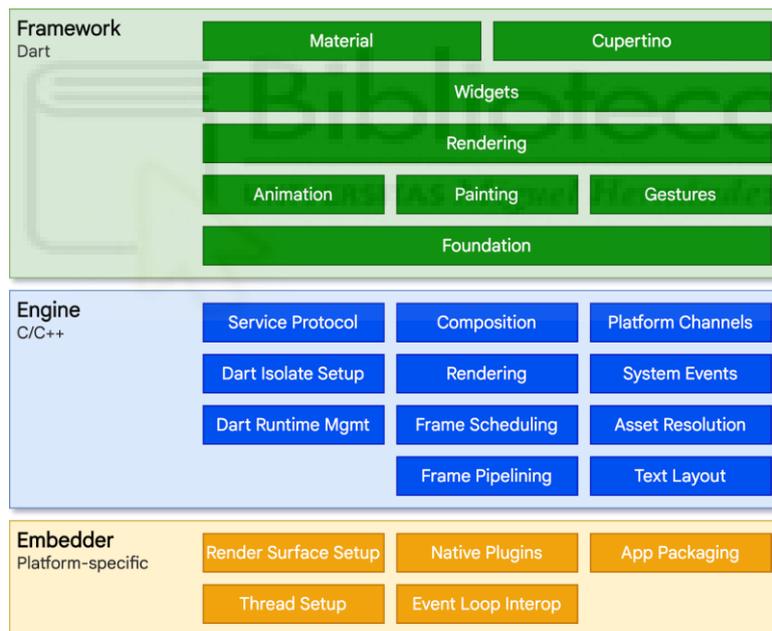


Ilustración 7 - Arquitectura Flutter

Las tres capas de Flutter están compuestas por un integrador (Embedder) que usa un lenguaje propio (muy específico de la plataforma) y de esta manera la aplicación podrá ser ejecutada desde cualquier dispositivo. Segunda capa es el motor que está escrito por completo en C/C++ que incluye la implementación a

bajo nivel de elementos como gráficos, diseños de texto, E/S de archivos y redes, una cadena de herramientas de compilación, etc. Y por último el framework que está escrito por completo en Dart, se puede escribir código tanto de back-end para que los componentes de front-end luzcan de una manera más bonita y profesional como el mismo front-end.

Flutter ofrece una herramienta de depuración cuyo nombre es Flutter DevTools que se utiliza para inspeccionar cualquier elemento de la pantalla (Al pinchar en cualquier parte de la pantalla, Flutter te lleva a donde está ubicado exactamente el componente), analizar el rendimiento y por supuesto, depurar la aplicación.

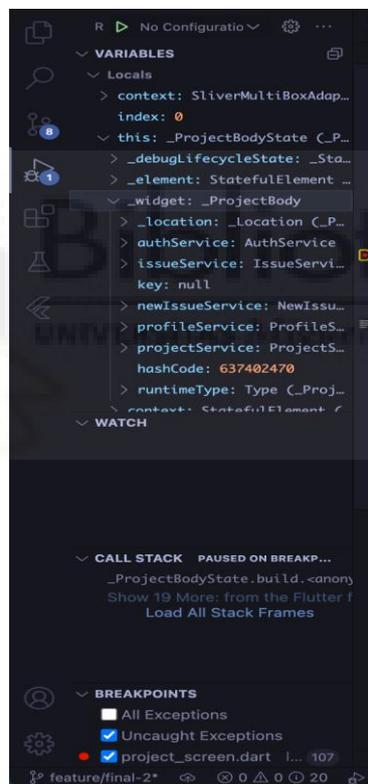


Ilustración 8 - Menú de depuración en Flutter

Por defecto si no hay ningún simulador instalado, Flutter ejecuta la aplicación en el navegador instalado por defecto. Pero para pruebas más profesionales se recomienda o instalar un simulador o ejecutar la aplicación en un dispositivo físico

real. En nuestro caso para probar en un simulador de ordenador, tenemos que instalar Android Studio y desde ahí instalar un simulador.

Para crear un proyecto de Flutter, en nuestro caso fue trabajar en el Visual Studio Code, hay que instalar el plugin que viene en las extensiones del editor del texto. Al instalar la extensión de Flutter se abre la posibilidad por consola del editor "Crear proyecto de Flutter", se puede crear tanto un proyecto completo, o si no, sólo un widget y pocas más opciones. Dicho proyecto ya viene con multitud de archivos y carpetas preinstalados. Vienen carpetas de Android, de ios, de web y de Windows.

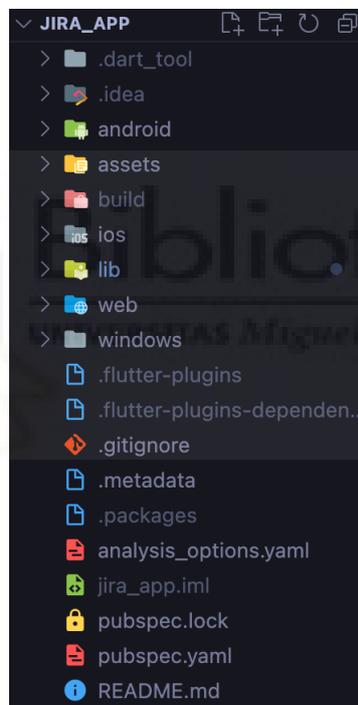


Ilustración 9 - Esqueleto de proyecto

Existe una segunda versión de Flutter que es la numero 2, de hecho, es la que hemos estado viendo en este apartado, la cual que permite ejecución de apps en un navegador y muchísimas más mejoras que facilitan el desarrollo de una aplicación.

En Flutter 2 se ha agregado un mecanismo llamado Null-Safety [11] que obliga al desarrollador a que ninguna variable se quede nula en ningún momento. De esta manera el programador está obligado a inicializar una variable o cuando toque utilizarla, en caso de que ésta sea no obligatoria que valga "algo" si es nula.

## **4.2.1 PAQUETES DE FLUTTER INTERNOS Y EXTERNOS**

A lo largo del desarrollo hemos dado casi todo el uso a las herramientas que proporciona el propio Flutter con sus librerías preinstaladas. Sin embargo, hace falta paquetes de framework que no vienen preinstalados y también necesitamos paquetes de terceros (Comunidad de Flutter) que ambos tipos están almacenados en un repositorio oficial (pub.dev [12]).

Para poder dar el uso de dichos paquetes hay que realizar dos cosas. Primero es entrar al repositorio, buscar el paquete que nos interesa, entrar a la opción de "Installing" y de ahí bien o copiar el código de la dependencia y agregarlo al proyecto o copiar el comando de consola y ejecutarlo en la consola de proyecto.

## dio 4.0.6

Published 3 months ago • flutterchina.club (Null safety)

SDK | DART | FLUTTER | PLATFORM | ANDROID | IOS | LINUX | MACOS | WEB | WINDOWS

4.0K

Readme | Changelog | Example | Installing | Versions | Scores

### Use this package as a library

Depend on it

Run this command:

With Dart:

```
$ dart pub add dio
```

With Flutter:

```
$ flutter pub add dio
```

This will add a line like this to your package's pubspec.yaml (and run an implicit `dart pub get`):

```
dependencies:  
  dio: ^4.0.6
```

### Ilustración 10 – Repositorio de Flutter - Importación de paquetes

Segundo, en el proyecto existe fichero llamado "pubspec.yaml" en el cual en caso de que no instalar la extensión vía consola, hay que pegar la línea de código copiada previamente del repositorio de Flutter.

```
pubspec.yaml
29 dependencies:
30   flutter:
31     sdk: flutter
32
33  /cupertino_icons: ^1.0.2
34   http: ^0.13.4
35   provider: ^6.0.2
36   flutter_secure_storage: ^5.0.2
37   dio: ^4.0.6
38   flutter_svg: ^1.1.0
39   url_launcher: ^6.1.0
```

Ilustración 11 - Archivo de las dependencias (Paquetes)

Continuemos con la lista de los paquetes usados en el proyecto:

- **Cupertino Icons [8]**



Ilustración 12 - Iconos de paquete de Cupertino Icons

Cupertino Icons es un paquete que tan sólo se centra en importar los iconos de aspecto de ios.

- **Provider [13]**

Provider [14] es un paquete que permite a diferentes pantallas de la aplicación a comunicarse entre sí. Por ejemplo, en la primera pantalla al solicitar los datos de una petición HTTP, se almacenan en una variable, en caso de que necesite coger datos de una variable en otra pantalla, se puede hacer llamando al provider.

```
final subTaskService = Provider.of<SubTaskService>(context);  
final authService = Provider.of<AuthService>(context);  
final projectService = Provider.of<ProjectService>(context);  
final issueService = Provider.of<IssueService>(context);
```

Ilustración 13 - Lista de llamadas a provider

De esta manera, todos los datos o métodos que no son privados, pueden ser accesibles por ejemplo mediante: subTaskService.CLASE.ATRIBUTO.

- **Flutter Secure Storage [15]**

Flutter Secure Storage permite almacenar los datos en un almacenamiento seguro. Una tercera persona no va a poder acceder a dichos datos.

```
_apiToken = await storage.read(key: 'key') ?? '';  
_user = await storage.read(key: 'name') ?? '';
```

Ilustración 14 - Ejemplo de uso de Flutter Secure Storage - Lectura de variables

- **Dio [16]**

Dio permite lanzar peticiones HTTP a una dirección URL. Este paquete permite mediante try catch captar errores de respuesta sin que se detenga la aplicación.

```
Dio dio = Dio();
try {
  await dio.post(
    urlWorklog,
    data: {
      "comment": addComment,
      "timeSpent": hoursAdd,
      "started": addDate
    },
    options: Options(
      headers: {
        'Authorization': 'Basic ${base64Encode(ascii.encode('${_user}:${_apiToken}'))}',
        'Content-Type': 'application/json'
      }
    )
  );
} on DioError catch (e) {
  if (e.response!.statusCode == 400) {
    print(e.response!.data);
    print(e.message);
  } else {
    print(e.message);
  }
}
```

Ilustración 15 - Caso real de uso de paquete, en caso fallido de petición imprimir los datos de error

- **URL Launcher [17]**

URL Launcher es un paquete que nos permite dejar nuestra aplicación abierta en segundo plano mientras se abre en primer plano el navegador con el enlace que nosotros indicamos. Además del navegador, nos permite abrir el correo electrónico, mandar un SMS, llamar a un número de teléfono o abrir un archivo o carpeta.

```
void lanzarUrl() async {  
  final Uri url = Uri.parse('https://umane.everis.com/jira');  
  await launchUrl(  
    url,  
    mode: LaunchMode.externalApplication  
  );  
}
```

Ilustración 16 - Código ilustrativo que abre el navegador con el url especificado

- **Loading Animation Widget** [18]

Loading Animation Widget es un Widget de diseño que permite poner muchos elementos de carga personalizados por waterydesert. Flutter dispone de un elemento de este tipo, pero he puesto éste porque coincide más con el diseño de la página web de Jira.



Ilustración 17 - Elemento de carga "discreteCircular"

## 4.2.2 VENTAJAS DE FLUTTER

- **Widgets listos:** La mayoría de los componentes que vamos a utilizar ya vienen preinstalados en el proyecto. Cualquier objeto en Flutter es un widget, desde un botón hasta un texto. El desarrollador tan solo tiene

definir en widgets con sus propiedades y lo tiene listo, en caso de querer personalizar o crear el widget nuevo, puede hacerlo a partir de widgets ya existentes. Los widgets en Flutter están organizados en árboles, lo cual es útil para renderizar, pero puede resultar en una complicación excesiva para la vista en caso de que haya más de 10 capas. Comparado con otros frameworks que usan diferentes objetos (Diseños, vistas y controladores) cuando el Flutter tiene un modelo de objetos consistente y unificado.

```
219         side: const BorderSide(color: Colors.black26),
220     ), // RoundedRectangleBorder
221     tileColor: Colors.grey[150],
222     style: ListTileStyle.list
223 ) // ListTile
224 ) // InkWell
225 ); // Material
226 }
227 ) // ListView.builder
228 ) // SizedBox
229 : Column(
230     children: [
231         const SizedBox(height: 5),
232         Text(
233             'No hay horas registradas',
234             style: TextStyle(color: Colors.red[300])
235         ) // Text
236     ]
237 ) // Column
238 ]
239 ), // Column
240 ) // Padding
241 ) // SingleChildScrollView
242 ) // SafeArea
243 : Column(
244     children: [
```

Ilustración 18 - Representación en árbol de los widgets entre otros

- **Dart – lenguaje simple y efectivo dirigido a programadores de Java:** Dart es un lenguaje moderno orientado a objetos que se parece bastante a Java o C++ por su sintaxis. Se puede declarar las variables tanto como un principiante, es decir, sin tipado (Dart adivina de que tipo es la variable), tanto con tipados. La compilación *Just-in-Time* permite dar ciclos de desarrollo más rápidos ya que compila en tiempo real lo cual permite ver el resultado al instante, sin embargo, la ejecución de la aplicación tarda más tiempo. No hay necesidad de archivos XML ya que el

propio Dart se ocupa de mantener el diseño y el código en un solo lugar – en un widget.

- **Curva de aprendizaje más suave:** No es difícil acostumbrarse a Dart según Google, ya que éste menciona que ha visto a personas sin conocimiento de programación o que antes nunca hayan desarrollado una aplicación para móvil, crear un prototipo o una aplicación. La documentación que proporciona Google es muy detallada y bien estructurada y además viene con ejemplos, por tanto, se hace muy fácil consultar y utilizar un widget que haga falta.
- **Amplia comunidad de desarrolladores:** Flutter lleva en el mercado poco tiempo y es lógico pensar que no hay mucha gente utilizando dicho framework, pero cabe destacar que no es cierto ya que el estudio anual de Stack Overflow mostró que Flutter es el tercer marco más requerido y cuarto más buscado de 2020. En cuanto personalmente a mí, siempre he podido encontrar multitud de soluciones a problemas que surgieron por el camino.



Ilustración 19 - Los frameworks más queridos de 2020, fuente: [Stack Overflow 2020 Developer Survey](#)

- **Google como garantía de soporte a largo plazo:** Dado que Flutter está desarrollado por Google y, de hecho, el propio Google es un gran usuario de esta tecnología, se puede estar seguro de que habrá soporte constante. Todos los errores se solucionarán de inmediato y el producto seguirá desarrollándose en el futuro. Google tiene otro proyecto que está en fase de desarrollo por varios años llamado Google Fuchsia [19] que es un sistema operativo de código abierto basado en el núcleo Zircon. Considerando todo lo anteriormente mencionado se puede estar seguro de que a largo plazo va a haber soporte de propio Google y de la comunidad de desarrollador.
- **Alto rendimiento:** Flutter promete un rendimiento constante de 60 fps o 120 fps en dispositivos que tengan una velocidad de actualización de pantalla de 120Hz, que es la velocidad a la cual aplicaciones modernas muestran la imagen. La velocidad tan rápida de Flutter se debe a la compilación directa de código sin puentes que ralentizan el rendimiento. Se ha hecho un estudio [20] de comparación de velocidad con Xamarin y React native y Flutter ocupa primer lugar en rendimiento, React native segundo y Xamarin tercer puesto. Estos frameworks los veremos más adelante.
- **Portabilidad:** Actualmente Flutter tiene soporte de 5 sistemas operativos que son Android, iOS, Linux, macOS y Windows (Y también web, pero no es un sistema operativo). Tiene tantos sistemas soportados debido a que el código se compila internamente y el sistema operativo sólo recibe la solución final. Además, Google está trabajando para expandir la lista de integración de Flutter en dispositivos como un coche, televisor, electrodomésticos inteligentes, dispositivos plegables, etc.

### 4.2.3 DESVENTAJAS DE FLUTTER

- **Falta de bibliotecas de terceros:** Los paquetes de terceros juegan un papel importante en la automatización de desarrollo de software para programadores y elimina la necesidad de codificar todo desde "cero". Estos paquetes suelen ser gratuitos y de código abierto que publican los desarrolladores tras realizar un proyecto con este paquete y tras haberlo probado. Hoy en día en el repositorio oficial podemos encontrar más de 15.000 paquetes, lo cual es mucho, pero frente a lenguajes más antiguos no es tanto. Pero no hay duda de que en unos años esa desventaja pasará a ser una ventaja ya que habrá muchísimos más paquetes.
- **Dart:** Una comparación [21] que ha hecho altexsoft.com con JavaScript, C#, Objective-C y Java nativos y ha llegado a la conclusión de que Dart pierde frente a estos lenguajes. No todos los desarrolladores pasarán a usar Dart & Flutter si antes estaban en estos 4 lenguajes.
- **Tamaño de aplicación más grande:** Los desarrolladores de las aplicaciones multiplataformas intentan reducir el tamaño de la misma, limitando las animaciones, compresión de imágenes, evitando importar muchas librerías y más cosas con fin de minimizar el tamaño de la misma y aumentando el rendimiento. Flutter tiene los widgets incorporados en vez de tenerlos externamente y esto obliga a que una aplicación ocupe como mínimo 4MB en la memoria del dispositivo. Comparando con otras aplicaciones nativas de Java que ocupan mínimo 539KB o Kotlin 550KB, sin embargo, otras alternativas híbridas tienen un tamaño mínimo superior, como por ejemplo Xamarin con casi 16MB y React Native con 7MB. Esta desventaja no tiene que suponer un gran problema en el año 2022 o en el futuro, ya que la memoria ya no es un problema como lo era antes.

## 4.2.4 ALTERNATIVAS PARA EL DESARROLLO HÍBRIDO

Existe gran variedad de alternativas de Flutter, ya sea antiguas o también nuevas, aquí vamos a mencionar las más parecidas.

### 4.2.4.1 XAMARIN [22]



Ilustración 20 - Logotipo de Apache Cordova

Xamarin es framework de código abierto de Microsoft que está desarrollado en C#. Es una tecnología multiplataforma que permite desarrollar aplicaciones para Android, iOS y Windows Mobile. Para aprovechar todo el potencial de Xamarin el desarrollador necesita tener algún conocimiento de iOS y Android además de C#. Hace uso de .NET y funciona como una capa de abstracción para la gestión de código compartido y de plataforma. Incluye su propio SDK para Windows y MacOS cuyo nombre es XamarinStudio. En Xamarin, el código escrito en C# se compila en un código de máquina y luego se empaqueta en .app.

#### 4.2.4.2 APACHE CORDOVA (PhoneGap) [23]

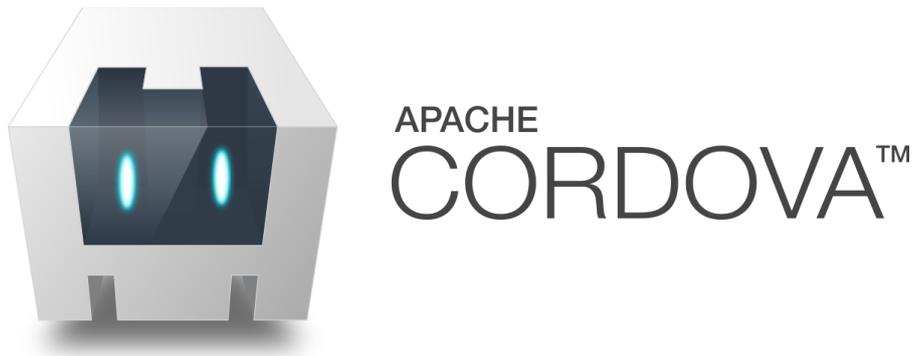


Ilustración 21 - Logotipo de Apache Cordova

Apache Cordova es un framework que fue creado por Nitobi en el año 2009, varios años más tarde, Adobe Systems compró la empresa y Cordova pasó a llamarse PhoneGap y luego otra vez a Apache Cordova. Las aplicaciones más populares desarrolladas en este framework son Untappd, Localeur, SparkChess y otras.

Es un framework de código abierto que se ejecuta en tecnologías web estándar como JavaScript, HTML5 y CSS3. Es una aplicación creada que parece que es un sitio web metido dentro de una aplicación. Toda la aplicación sólo usa un widget en el cual se realizan todas las acciones e interfaces de la aplicación y de esta manera puede parecer como una nativa. Cordova no tiene análisis de código automatizado como Flutter, pero lo que tiene parecido es LiveReload, que vuelve a compilar en caliente sin necesidad de reiniciar la aplicación. Apache Cordova es un caso ideal si hace falta crear una aplicación rápido y no hay tiempo para aprender un nuevo lenguaje si ya tienes conocimientos de desarrollo de la web.

#### 4.2.4.3 REACT NAVITE [24]

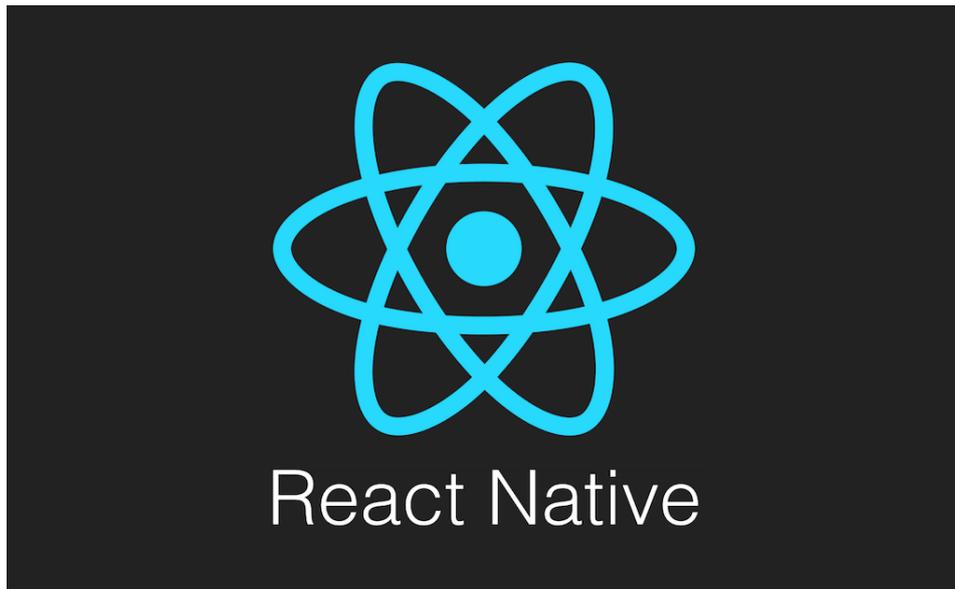


Ilustración 22 - Logotipo de React Native

React Native es un framework de código abierto, multiplataforma desarrollado por ingenieros de Facebook. Las aplicaciones como Instagram, Tesla, Uber Eats, etc. están desarrolladas en React Native. El aspecto de la aplicación parece nativo porque está basado en React y utiliza los mismo elementos de interfaz que utilizaría una aplicación nativa. Las aplicaciones finales están escritas en JavaScript y un marcado basado en JSL. La mayoría de los componentes se puede usar tanto en Android como en iOS, pero no todos.

#### 4.2.5 CONCLUSIÓN

Entre las opciones analizadas, no podemos decir cuál es mejor y cuál es peor ya que cada uno usa su(s) lenguaje(s) y cada persona tiene su preferido o el que más sepa. En cuanto para nuestro desarrollo, Flutter es el mejor ya que me apasiona mucho Java & POO y además tenemos conocimiento de estas dos tecnologías. Flutter es muy parecido, por tanto, se disfruta mucho desarrollando. En cuanto si una persona tiene mucho dominio y le gusta mucho web y quiere

una aplicación para un dispositivo móvil y no quiere meterse al mundo de lenguajes de programación, pues le vendría muy bien Apache Cordova. Si la persona domina mucho de JavaScript y de XML, React native sería muy buena opción.

### **4.3 LUCID**

Para la creación de los diferentes diagramas que ayudan para el desarrollo del proyecto vamos a utilizar tecnología cuyo nombre es Lucidchart [25]. Es un creador de diagramas que proporciona multitud de plantillas de todos los tipos que hay y especialmente las que vamos a utilizar, de esta manera se ahorra el tiempo de trabajo y esto es muy importante.

Se ha elegido Lucidchart por su diseño moderno e intuitivo. Además, se puede hacer de forma gratuita, en caso de utilizar opciones más profesionales existe una versión de pago, aunque para el proyecto una versión gratuita es más que suficiente. Cuenta con el almacenamiento en la nube, para uso de dicha función hace falta tener una cuenta.

Entre otras alternativas destacan Microsoft Visio, pero al contrario que Lucid, Visio no ofrece una versión gratuita y, Dia, es otro creador de diagramas, pero gratuito, pero no está a un nivel de Lucidchart.

### **4.4 API & JSON**

A la hora de crear, modificar o eliminar datos del proyecto hace falta estar conectado a la página web de Jira. De mismo modo, para hacerlo desde fuera de la propia web, hay que hacerlo a través de la API que proporciona mismo Atlassian.

API [26] (Application Programming Interfaces) es una especie de módulo que permite interactuar dos aplicaciones (en nuestro caso web y aplicación) entre sí con una serie de reglas de comunicación. API permite tanto solicitar los datos, ver los nombres, estados, operaciones posibles, etc, como realizar operaciones como modificar, crear, borrar, etc.

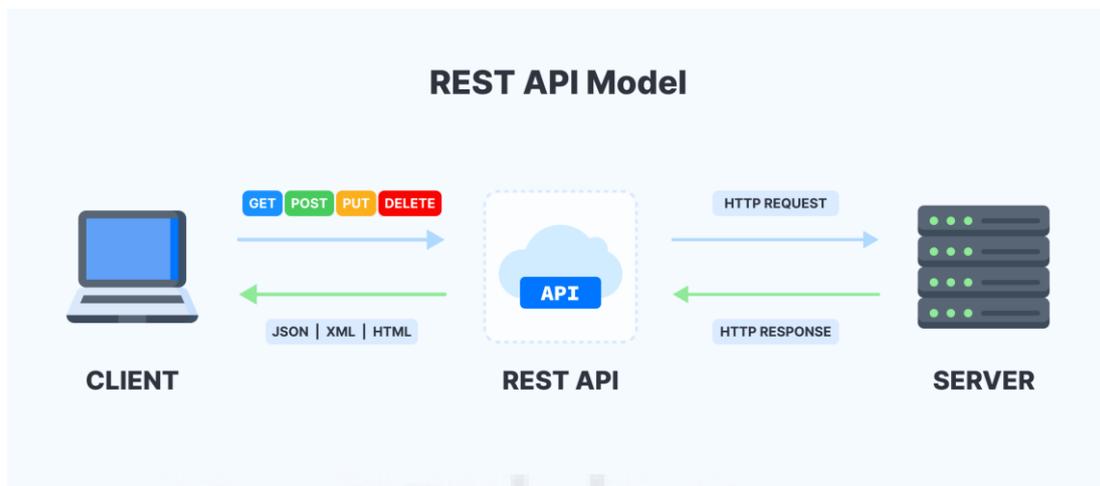


Ilustración 23 - Funcionamiento de la api. Fuente: <https://appmaster.io/es>

Para poder interactuar con la API, hay que enviar peticiones HTTP a la misma. Para obtener datos existe una petición de tipo GET, para agregar información se usa POST, para reemplazar PUT, y para eliminar DELETE. Hay más opciones, pero Jira sólo utiliza estas cuatro.

JSON [27] es una especie de árbol en la cual hay ramas, cada rama tiene propio nombre, dicha rama puede tener más ramas que salgan de esta. Cada rama puede tener diferente tipo, ya sea un texto, un numero o un booleano. JSON es una extensión de archivo que permite guardar e intercambiar la información estructurada entre un servidor y un cliente.

```
{
  "nombre": "Tom",
  "apellido": "Jackson",
  "género": "masculino",
  "hobby": ["fútbol", "lectura", "natación"]
}
```

Ilustración 24 - Ejemplo de un .json. Fuente: <https://www.hostinger.es/>

## 4.5 POSTMAN

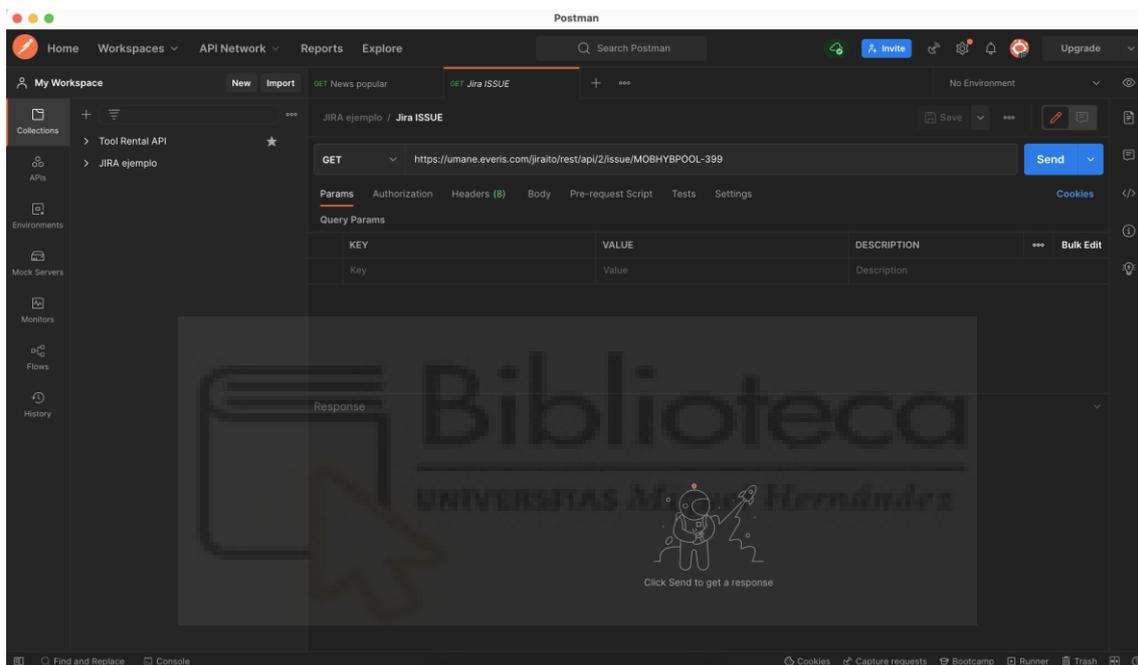


Ilustración 25 - Entorno de Postman

A la hora de leer los datos que están almacenados en formato Json que proporciona la API de Jira, vamos a usar un programa de escritorio Postman [28]. Este programa permite realizar pruebas de la API con una interfaz gráfica, ahorrando el tiempo que se tardaría con la consola del ordenador.

Postman permite agilizar el testeo de la API gracias a creaciones de colecciones de envíos HTTP. Cada colección puede tener algunas o muchas peticiones que se van a enviar a la API de forma secuencial o desordenada, asimismo antes de

integrar estas funcionalidades a la aplicación móvil, es conveniente probar hacerlo desde Postman.

Al testear la API de Jira nos podremos encontrar ante códigos de respuesta que nos indican el estado de nuestra petición. Los que vayamos a encontrar por el camino y los más comunes que hay son: 200 OK, este código quiere decir que la petición se ha enviado y procesado correctamente, 403 Forbidden, este código quiere decir que la petición ha sido enviada pero fue denegada por el camino debido a que no hay suficientes permisos para realizarla, 404 Error, éste código sale en caso de que la petición que mandamos no coincide con ninguna de las peticiones que proporciona la API, es decir, no existe o no existen los datos con estos nombres. El código 401 indica que el usuario no está identificado y por tanto la API no puede responder nada a la petición. Y el último código es 400, que indica que la petición es correcta pero sólo uno o varios campos son erróneos.

## 4.6 QUICKTYPE.IO

El formato de datos que nos proporciona la api de Jira es un fichero de tipo Json. Sin embargo, para no escribir el camino a cada dato mediante la url específica, se pone la URL general, y toda la respuesta que da api, se copia y se pega a Quicktype.io [29].

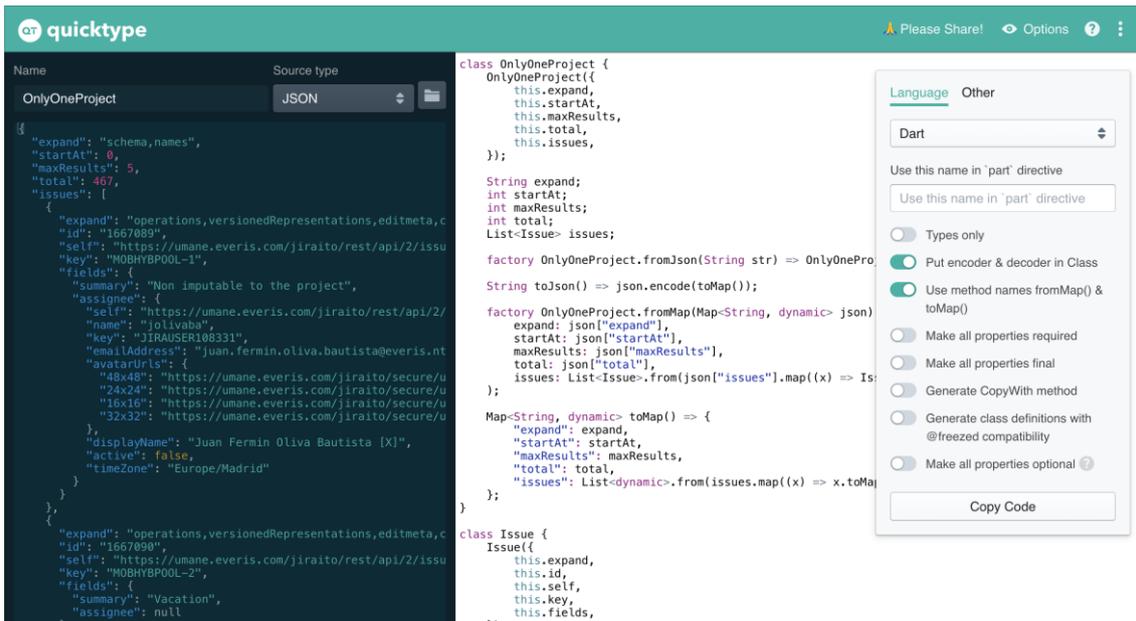


Ilustración 26 - Conversión de JSON (Izquierda) a una clase (Derecha)

Lo que hace quicktype es, tras la lectura de los datos, generalizarlos y crear clase que tiene sus atributos, listas, otras clases, etc. Además, crea dos métodos, uno de ellos es encarga de tras realizar la petición HTTP, almacena toda la salida en una cadena muy larga, al llamar al método, se pasa de parámetro dicha cadena y se "crea" una clase con todos los datos recibidos; otro método hace todo lo contrario para enviar los datos a la api.

## 4.7 VISUAL STUDIO CODE

Para el desarrollo del proyecto es necesario un editor de texto o un IDE en el que se puede escribir el código fuente de la aplicación y observar la jerarquía de los archivos del proyecto y, más opciones que agilicen el desarrollo de la misma.

Hoy en día disponemos de una amplísima gama de alternativas para lograr esta tarea. Disponemos de una amplia cantidad de editores gratuitos como [30] Atom,

Sublime Text, Notepad++, y más. La elección final ha sido Visual Studio Code el editor de código gratuito y open source de Microsoft.

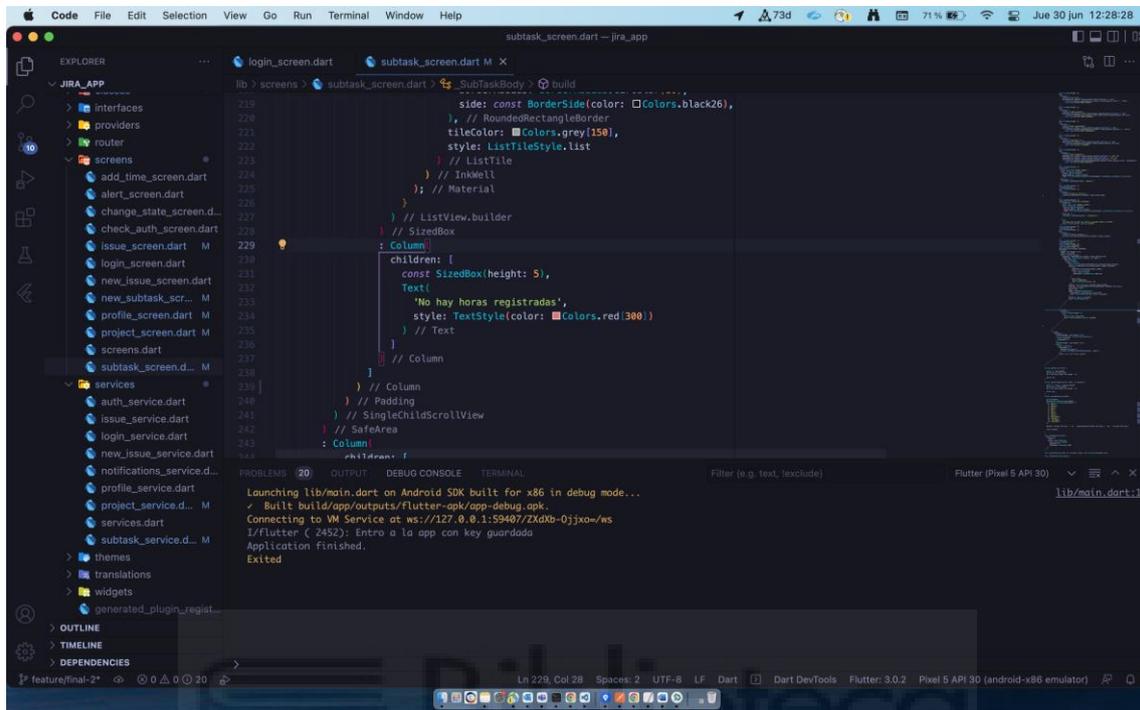


Ilustración 27 - Interfaz de usuario de Visual Studio Code

Visual Studio Code ofrece muchísimos temas para cada gusto, dispone de muchas extensiones que favorecen al desarrollo, consola y la opción de “debuggear”.

## 4.8 GIT & GITHUB

Para registrar los cambios que se realizan en el desarrollo de la aplicación vamos a usar GIT [31]. GIT es un sistema de control de versiones más moderno y usado que a día de hoy existe. El rendimiento que muestra esta tecnología es superior haciendo muy rápidos los cambios que se agregan al proyecto. Además, cada desarrollador que esté implicado en el desarrollo puede tener su propio historial de cambios y una vez finalice, fusionar nuevas agregas al sistema común dónde están almacenados todos los cambios. De este modo, cada desarrollador no

depende de otro desarrollador, cada uno puede estar trabajando en sus misiones de forma independiente y siempre cuando ese cambio no pueda afectar al desarrollo del otro programador.

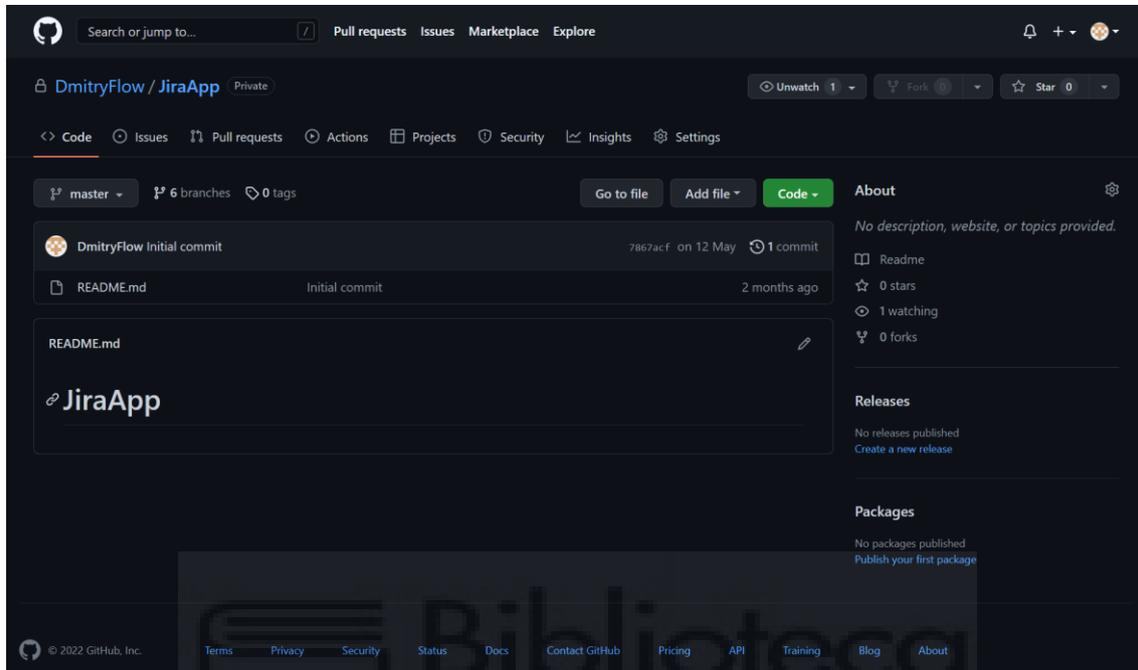


Ilustración 28 - Interfaz del sitio web de Github

El único inconveniente es que hay que dedicar bastante tiempo para familiarizarse con los comandos y el uso de dicha tecnología si previamente nunca has oído hablar de ello. Pero aquí viene un “apoyo” que puede ahorrar el tiempo de tecleo y memorización de comandos. Dicho apoyo se denomina Git Flow [32]. La función que cumple Git Flow es resumir varias líneas de comandos en una sola, así mismo ahorrando el tiempo y agilizar el proceso de registro de cambios.

Además de Git Flow, si se desea aún más fácil hacer el desarrollo, existe una versión gráfica. No es una alternativa, aunque permite trabajar como antes, pero viendo todo con una interfaz visual. Es un programa de escritorio cuyo nombre es Sourcetree [33].

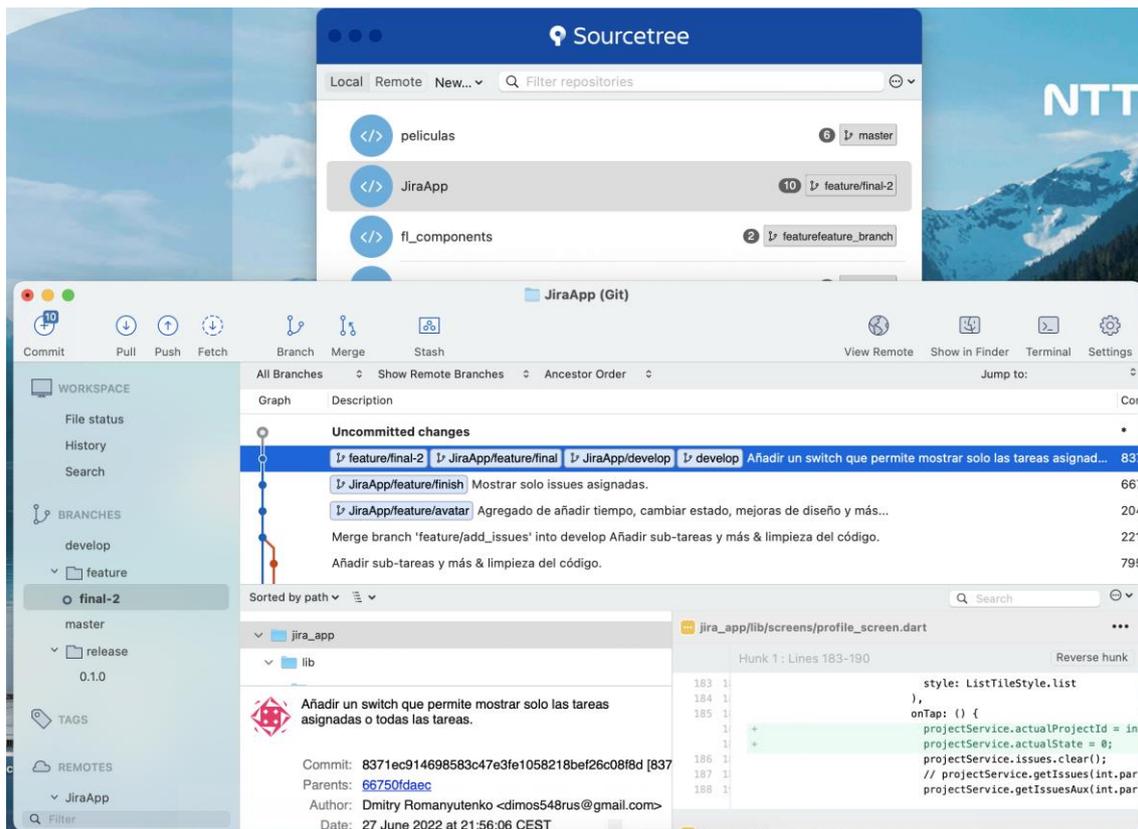


Ilustración 29 - Interfaz de Sourcetree

Sourcetree permite ver los cambios que aún no han sido añadidos al sistema sin necesidad de escribir y buscar por consola dichos cambios que es una forma muy tediosa. Además, puedes ver más rápido el historial de los cambios de cada versión, etc. Pero es más conveniente trabajar con la consola desde dentro del programa que el mismo programa lo permite ya que hay más control.

## 4.9 ANDROID STUDIO

A la hora de probar la aplicación hay que tener en cuenta que la aplicación se va a ejecutar en diferentes dispositivos. Tener tantos dispositivos físicamente es imposible ya que supondría un gasto muy elevado. Para resolver este problema, viene Android Studio con su Android Virtual Device [34] (AVD). AVD es un

dispositivo virtual (emulador) que permite hacer uso del mismo sin tenerlo físicamente.

Android Studio nos permite crear y utilizar dispositivos AVD de forma sencilla, eligiendo entre una gran variedad de dispositivos para emular. Además, Android Studio es completamente gratuito, al contrario que algunas alternativas como Genymotion.

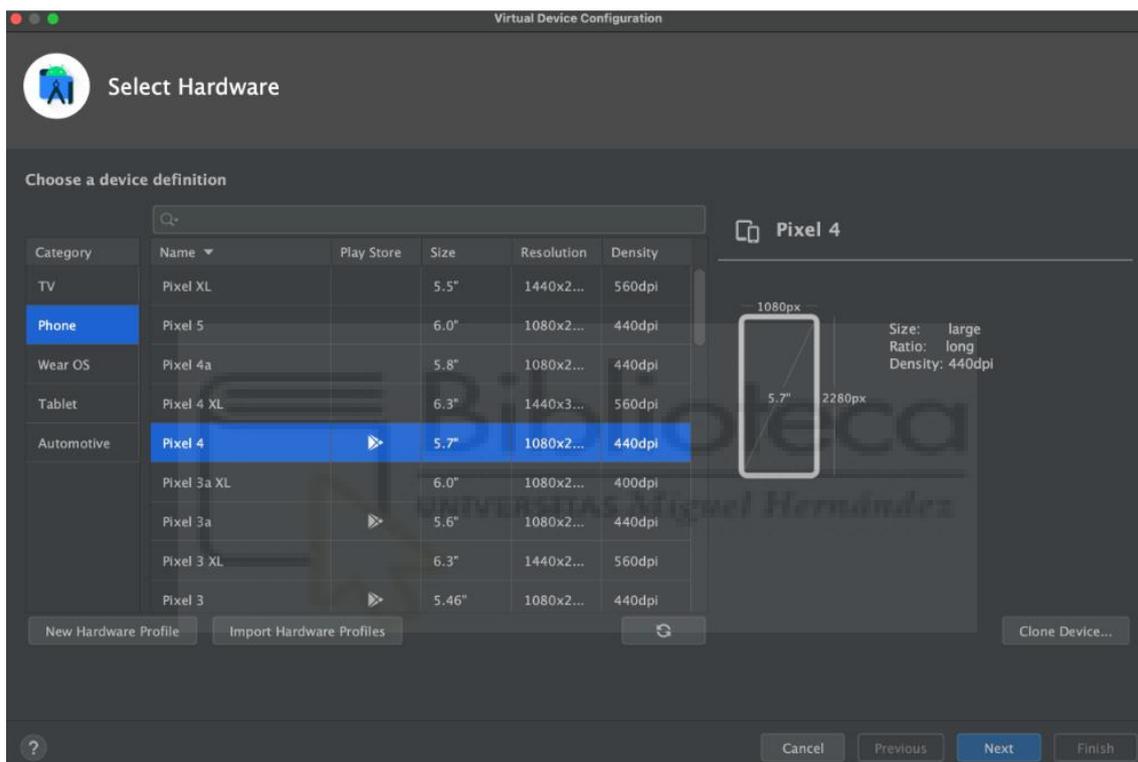


Ilustración 30 - Lista de dispositivos que se puede crear para emular. Fuente:

<https://developer.android.com/>

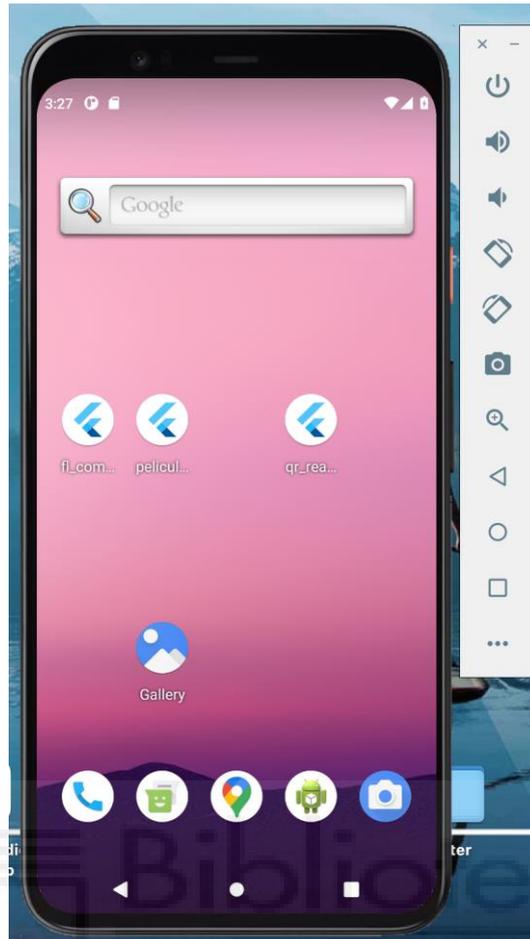


Ilustración 31 - Emulando el dispositivo Pixel 5

## 4.10 EQUIPOS FÍSICOS

Para el desarrollo de este proyecto vamos a usar un ordenador de sobremesa, un ordenador portátil, un dispositivo móvil físico y un emulador donde probar la aplicación.

### 4.10.1 EQUIPO DE SOBREMESA

- Sistema Operativo Microsoft Windows 10 (64 bits)
- Procesador Intel Core i5 – 10400F - 2.90 GHz

- 16 GB de memoria RAM DDR4
- Disco duro HDD de 1 TB
- Disco duro SSD de 240 GB
- Disco duro SSD (M.2) de 400 GB
- Tarjeta gráfica NVIDIA GeForce GTX 1660S 6GB

#### **4.10.2 EQUIPO PORTÁTIL [35]**

- Portátil MacBook Pro 13"
- Sistema Operativo macOS (64 bits)
- Intel Core i5-7360U
- 8 GB de memoria RAM DDR4
- Disco duro SSD de 128 GB
- Intel Iris Plus Graphics 640

#### **4.10.3 DISPOSITIVO MÓVIL FÍSICO [36]**

- Samsung A31
- Sistema Operativo One UI 4.1 (Android 12)
- Resolución de pantalla 1080 x 2400 (FHD+)
- Procesador Octa-Core 2GHz, 1.7GHz
- Almacenamiento interno de 128 GB
- 4 GB de memoria RAM
- Cámara principal de 48.0 + 5.0 + 8.0 + 5.0 Megapíxeles
- Cámara frontal de 20 Megapíxeles

#### 4.10.4 DISPOSITIVO MÓVIL EMULADO [37]

- GOOGLE PIXEL 5
- Sistema Operativo Android 11
- Resolución de pantalla 6", 1080 x 2340 píxeles
- Procesador Snapdragon 765G 2.4GHz
- Almacenamiento interno 128 GB
- 8 GB de memoria RAM
- Cámara Dual, 12.2MP+16MP





# CAPÍTULO 5

## METODOLOGÍA Y RESULTADOS

---

### 5.1 CICLO DE VIDA DEL PROYECTO

El ciclo de vida de desarrollo del proyecto de software contempla las fases necesarias para un desarrollo válido y exitoso. Con las fases posteriormente mencionadas se pretende garantizar que se cumplan requisitos propuestos por la empresa y verificación de los procedimientos de desarrollo asegurándose de que los métodos usados son apropiados.

La causa de creación de diferentes fases nace desde la necesidad de no permitir errores graves durante el desarrollo, de esta manera no perder una cantidad enorme del tiempo corrigiéndolos y asimismo prevenir posibles incertidumbres que desmotivarían al programador. Utilizando metodologías apropiadas, se podría evitar los puntos negativos mencionados anteriormente y asimismo ahorrando el costo de desarrollo.

Durante el desarrollo, posiblemente no todo será cumplido al 100% debido a que no siempre es el camino correcto que se implante al principio. Ya que en la mitad del camino podría surgir una idea mejor que así mismo se vería más innovadora y aceleradora para el proyecto.

Por este motivo, se ha elegido el modelo de ciclo de vida en cascada mejorado [38], un modelo que puede ser exitoso con mayor porcentaje. Es sencillo de implantar y gestionar, y es especialmente útil para desarrollos pequeños o medianos que es nuestro caso.



El back-end de la aplicación tiene que recibir los datos desde la API de Jira donde la empresa facilita el acceso.

Es necesario que la interfaz de la aplicación sea lo suficiente intuitiva y parecida a la página web de Jira Software para que el usuario final no se pierda por el camino hacia el objetivo propuesto al entrar a la aplicación.

## **5.1.2 ESPECIFICACIÓN Y DISEÑO**

Este apartado comprende el análisis de los requerimientos de la empresa y el diseño de los componentes de la aplicación referentes al código, la api, etc.

En primer lugar, la aplicación solamente puede usar un empleado de la empresa cuyo nombre es NTT Data. Esto se debe a que las peticiones HTTP se realizan a un dominio reservado por la empresa. Todo usuario ajeno a la empresa va a recibir un mensaje de validación denegada. Por tanto, solamente va a haber un rol en la aplicación, y su nombre es "empleado".

En segundo lugar, un empleado va a tener acceso a diversas pantallas en las cuales podrá leer los datos, crear contenido y modificar contenido. Dichas pantallas tienen relación directa entre sí, ya que comparten los datos debido a que en todo el momento va a estar registrado el mismo usuario. No puede haber más de un usuario a la vez en la aplicación.

Una vez aclarados dos puntos anteriores, pasamos a la parte del diseño y el back-end de la aplicación. Como hemos mencionado antes, solamente tenemos un rol en la aplicación, así que, a partir de ahora hablemos solamente de este rol.

Primera pantalla que se va a encontrar el empleado será la pantalla de "Inicio de sesión". En ella el usuario puede introducir sus datos correspondientes y entrar

al sistema, guardando sus credenciales hasta que no cierre la sesión o no caduque la API Key [39]. API Key es un código que se utiliza para identificar y autenticar a un usuario, dicha clave puede tener una fecha de caducidad y dos modos, uno de sólo lectura y otro de lectura y escritura. Se recomienda solicitar el modo de lectura y escritura ya que en la aplicación se puede crear y modificar datos. NTT Data cuenta con un sistema de seguridad que no deja a un empleado sin acceso obtener la API Key. El empleado para poder usar la aplicación tiene que generar la API Key desde la página de Jira y almacenarla en un sitio seguro, pero para generarla tiene que entrar en el menú de las claves en Jira, pero para poder entrar al menú el empleado tiene que solicitar el acceso que proporciona un servicio de la empresa.

Aparte de los campos dónde se introducen las credenciales como el nombre del empleado y su API Key, la pantalla dispone de un botón que lleva a la siguiente pantalla. Todo usuario que no tenga una propia clave de acceso dispone de un botón que pone la aplicación en segundo plano y abre el navegador por defecto en el url de servicio de la empresa, dónde puede solicitar el acceso al menú en Jira dónde puede generar la clave. Y por último la pantalla luce con logos de Atlassian y con unas fuentes extraídas de la página web de Jira.

Antes de pasar a la segunda pantalla tenemos que mencionar una pantalla que aparece durante todo el uso de la aplicación. La pantalla se llama ni más ni menos como "Pantalla de carga". Esta pantalla se encarga de mostrar al usuario que el estado actual de la aplicación es "cargando". Cada pantalla tiene sus variables que identifican el estado, si alguna de las variables tiene un valor negativo, aparece la pantalla de carga, en caso positivo, se abre la pantalla requerida. De la pantalla se puede remarcar la parte superior donde aparece el texto "CARGANDO", en el centro de la pantalla aparece un elemento de carga (Una rueda girándose alrededor de un eje) y, debajo del elemento hay un botón de cierre de sesión, el funcionamiento de dicho botón se describe más adelante.

Segunda pantalla de navegación se llama "Perfil". En ella, en la parte superior de la pantalla disponemos de un texto que indica el nombre y apellidos del empleado y su foto de perfil. Dentro de la pantalla también se pintan los detalles del usuario identificado como el nombre de usuario, nombre completo y el correo electrónico de Jira. Más abajo se encuentra una lista dinámica de proyectos asignados al empleado, en cada elemento de la lista aparece el nombre del proyecto, tipo de proyecto y el código del proyecto. Después de la lista aparece un botón de cierre de sesión. He de remarcar que existe un elemento indicativo que separa dichos elementos con una línea gris medio transparente que mejora la visibilidad de la interfaz.

En esta pantalla el empleado dispone de dos acciones que puede realizar:

- **Entrar a un proyecto de la lista:** Esta acción abre la pantalla cuyo nombre es "Proyecto" con el proyecto elegido de la lista que lo veremos en el siguiente más abajo.
- **Cerrar sesión:** Esta acción cierra todas las pantallas abiertas, borra todos los datos de inicio del usuario y navega a la pantalla "Inicio de sesión".

Pasamos a la pantalla de "Proyecto" a la cual se accede desde la pantalla de "Perfil". En esta pantalla en la parte superior encontramos el nombre del proyecto, un botón que se detalla más adelante y un botón de retroceso. Además, hay un elemento de texto indicativo dándole a saber al usuario qué lista se está mostrando en este momento. Todo el contenido de la pantalla es una lista dinámica que refleja las tareas asignadas a dicho proyecto. Dentro de un elemento de lista podemos ver el nombre de la tarea, el código de la tarea, la descripción de la tarea y un ícono de decoración. En caso de que el usuario no tenga ninguna tarea asignada, aparece un mensaje de texto en el centro informando que no hay ninguna tarea asignada, por tanto, el empleado puede cambiar de lista y ver todas las tareas del proyecto.

Esta pantalla dispone de tres acciones posibles que el empleado puede realizar:

- **Volver a la pantalla anterior:** Un botón que se encuentra en la parte superior izquierda cumple con la función de volver a la pantalla anterior.
- **Cambio de la lista de tareas:** Esta acción permite ver dos listas diferentes, una de ellas muestra las tareas asignadas al usuario identificado en la aplicación y, la segunda lista muestra todas las tareas asignadas al proyecto actual.
- **Entrar a una tarea de la lista:** Esta acción abre la pantalla cuyo nombre es "Tarea" con la tarea elegida de la lista que lo veremos en el siguiente apartado que lo veremos a continuación.

Continuamos con la pantalla cuyo nombre es "Tarea" y a la cual se accede desde la pantalla "Proyecto" y desde la pantalla "Crear Subtarea". Comenzamos a comentar el contenido desde la parte superior izquierda dónde disponemos de un botón de retroceso, en el centro aparece el nombre de la tarea y, en la parte de derecha se encuentra un botón de agregado de subtarea cuya función se detalla más adelante. El resto de la pantalla es todo una lista dinámica que contiene el listado de subtareas asignadas a dicha tarea. Dentro de un elemento de la lista disponemos de dos textos, uno de ellos es el nombre de subtarea y otro es la descripción de la misma y, tercer elemento es el número (código) de la subtarea. Y como en la pantalla anterior, cada elemento dispone de un ícono decorativo.

En esta pantalla también disponemos de tres acciones que se detallan a continuación:

- **Volver a la pantalla anterior:** Hace la misma función que la pantalla anterior, pero lleva a la pantalla de "Proyecto".
- **Crear una nueva subtarea:** Esta acción abre una nueva pantalla que se denomina "Crear Subtarea". El detalle de ésta lo veremos más adelante.

- **Entrar a una subtarea de la lista:** Esta acción abre la pantalla "Subtarea" con la subtarea elegida de la lista.

Avanzamos con la pantalla de "Subtarea" a la cual se accede desde la pantalla de "Tarea" y también desde las pantallas como "Añadir Tiempo" y "Cambiar estado". Esta pantalla es la que más contenido tiene comparado con las otras ya que tiene que llevar toda la información de utilidad de una subtarea. En la parte superior disponemos de un botón de retroceso y del título de la subtarea. En la parte de contenido de la pantalla comenzamos con los textos informativos como el código de la tarea padre, el nombre de la tarea actual, el código de la tarea actual, estos textos nos sirven para indicar al usuario dónde se encuentra actualmente y no tener que navegar a las pantallas anteriores para adivinarlo.

Después de estos campos nos encontramos con una línea separativa de color gris medio transparente que ayuda al usuario distinguir diferentes partes de texto y, posteriormente habrá más líneas de distinguido. Continuamos con el apartado de texto de las horas registradas, dónde se muestran total de horas, tiempo restante, tiempo completado en % y las horas restantes para acabar la tarea. Justo debajo de estos textos disponemos de un botón cuyo nombre es "Añadir tiempo", el detalle de este botón lo veremos más en detalle más adelante. En caso de que no haya horas en un subtarea, en los campos de texto aparece el texto por defecto informando al usuario que no hay horas registradas.

Continuamos con el siguiente subapartado del estado de la subtarea. De color verde aparece si el estado es abierto, de color rojo si está cerrada la subtarea, de color azul que está en estado "Delivered" y de color negro el resto de los estados. Debajo del texto informativo aparece el botón "Cambiar estado" cuya función la veremos más en detalle a posteriori. En caso de que el estado esté en modo "Closed", desaparece el botón de cambio de estado ya que Jira no permite cambiar el estado si la tarea está cerrada.

Siguiente subapartado es bastante corto ya que muestra la fecha de creación de la subtarea, por ejemplo "5 de abril del 2022".

Y finalizamos con el subapartado de las horas registradas. En este subapartado tenemos un texto "Horas registradas:" y justo debajo aparece una lista dinámica con el historial de las horas registradas en la tarea. Dentro de cada elemento de la lista disponemos de la fecha de agregado del tiempo, el texto del tiempo agregado, en caso de que haya un comentario, un comentario y, cada uno de los elementos tiene un icono decorativo de calendario.

En esta pantalla podemos encontrarnos con tres posibles acciones:

- **Volver a la pantalla anterior:** Esta acción vuelve a la pantalla de las tareas.
- **Añadir tiempo a la tarea:** Esta acción lleva a una nueva pantalla cuyo nombre es "Añadir Tiempo", cuyo contenido se detalla posteriormente,
- **Cambiar estado de la tarea:** Esta acción lleva a una nueva pantalla "Cambiar Estado" para cambiar el estado de la tarea. Este botón no aparece solo en caso de que la tarea este cerrada.

Siguiente pantalla de la cual vamos a hablar se llama "Añadir tiempo" a la cual se puede acceder solamente desde la pantalla de "Subtarea". En la parte superior disponemos de un botón de retroceso a la pantalla anterior y de un texto del nombre de la subtarea actual. En el contenido de la pantalla tenemos dos campos de tipo texto. Más abajo aparece el botón de selección de la fecha de agregado del tiempo. Y abajo del todo hay un botón de "Añadir" el tiempo.

Lista de las acciones de la pantalla de "Añadir Tiempo":

- **Volver a la pantalla anterior:** Volver a la pantalla de "Subtarea".

- **Dos campos de texto:** Uno de ellos es el tiempo para añadir a la subtarea y otro es el comentario sobre el agregado. Dentro del campo de texto disponemos de un texto de ayuda que indica el formado del tiempo a añadir y el comentario en el texto de comentario.
- **Botón de selección de la fecha:** Esta acción permite seleccionar el día con un rango entre un año antes de creación de la tarea y dos años después de la creación de la tarea (Estos criterios ha comentado la empresa).
- **Botón de añadir el tiempo y ventana emergente:** Al pinchar sobre el botón se abre una ventana emergente que pregunta al usuario si está seguro de que quiere añadir el tiempo, en caso negativo cierra la ventana emergente y en caso positivo agrega el tiempo y cierra la pantalla actual y vuelve a la pantalla de subtarea actual.

Penúltima pantalla se llama "Cambiar Estado" a la cual solamente se puede acceder desde la pantalla de subtarea. Es la pantalla más corta de todas ya que solamente dispone de varios elementos. En la parte superior disponemos de botón de retroceso y el texto de nombre de la subtarea actual. El contenido de la pantalla tiene un texto que indica el estado actual de la tarea, los colores son los mismos que en la pantalla de subtarea. Debajo tenemos un menú desplegable que es una lista de estados de subtarea disponibles ahora mismo. Se ha elegido el widget de un menú desplegable porque la lista dinámica puede llegar a ser muy larga debido a que en Jira se puede crear estados personalizados y una subtarea puede llegar a tener más de 10 estados. Y más abajo hay un botón de guardado.

- **Volver a la pantalla anterior:** Volver a la pantalla de "Subtarea".
- **Lista desplegable:** Esta acción permite seleccionar el estado al cual se desea pasar la tarea.
- **Botón de guardar el tiempo y ventana emergente:** Al pinchar sobre el botón se abre una ventana emergente que pregunta al usuario si está

seguro de que quiere cambiar el estado de la subtarea, en caso negativo cierra la ventana emergente y en caso positivo cambia el estado y cierra la pantalla actual y vuelve a la pantalla de subtarea actual.

Última pantalla se llama "Crear Subtarea" y solamente se puede acceder a la misma desde la pantalla de "Tareas". En la parte superior aparece el botón de retroceso a la pantalla anterior y el texto con nombre de la tarea actual. En el contenido de la pantalla disponemos el texto que informa al usuario en qué proyecto estamos ahora mismo. Más abajo tenemos dos campos de texto. Debajo aparece un menú desplegable con una lista dinámica de tipos de subtarea que se desea crear. Y abajo del todo hay un botón de crear la subtarea.

- **Volver a la pantalla anterior:** Volver a la pantalla de "Tarea".
- **Dos campos de texto:** Uno de ellos tiene que ser rellenado con el nombre de subtarea y otro con la descripción
- **Botón de selección de la fecha:** Esta acción permite seleccionar el tipo de subtarea que se desea crear.
- **Botón de añadir el tiempo y ventana emergente:** Al pinchar sobre el botón se crea la subtarea, se cierra la pantalla actual y vuelve a la pantalla de las tareas.

### 5.1.3 IMPLEMENTACIÓN Y PRUEBAS UNITARIAS

Esta fase de ciclo consiste en traducir el diseño a código, crear clases a partir de Json proporcionado por la API, etc.

Toda la aplicación se codificará con Dart utilizando el framework Flutter, tanto la parte de front-end como la parte de back-end. La aplicación se comunicará con

la API mediante peticiones GET, POST y PUT para obtener, publicar y modificar los datos de la api.

Cada nuevo elemento creado con el código deberá ser comprobado antes de subirlo al repositorio de las versiones (GitHub).

## **5.1.4 VALIDACIÓN Y PRUEBAS**

Esta etapa del ciclo se ocupa de comprobar cada una de las operaciones que se puede realizar utilizando la aplicación.

Aunque las pruebas unitarias se realizaron en el ciclo anterior, éstos fueron probado uno por uno, en esta fase tiene que comprobarse todo de manera conjunta para comprobar correcto funcionamiento de toda la aplicación

En esta etapa tenemos que estar probando en un dispositivo virtual y posteriormente en un dispositivo físico como si fuera un usuario real. Los dispositivos de comprobación se detallan el apartado de equipos físicos.

Durante las pruebas de la aplicación surgieron diferentes problemas, estos inconvenientes eran esperados ya que es raro que una aplicación funcione correctamente del todo desde la primera vez. Para corregir los problemas, hay que regresar al ciclo anterior, programar código sin errores, probar, pasar al ciclo actual y volver a probarlo todo junto.

Debido a que cada proyecto, tarea y subtarea tienen datos diferentes, o algunos tienen datos que otros no, pues esto provoca error debido al mecanismo de Flutter Null Safety cuya definición está definida previamente.

Solo disponemos de una cuenta del empleado, de este modo, solamente podemos realizar las pruebas desde una perspectiva, aunque se ha desarrollado de modo que tiene que salir la información independientemente si es un empleado u otro. En caso de que la API devuelva campo nulo o vacío (En API hay diferencia entre ambos), se mostrará al empleado mensaje por defecto.

### **5.1.5 IMPLANTACIÓN, USO Y MANTENIMIENTO**

En esta última fase del ciclo, el sistema está comprobado y listo para funcionar. Es en este momento donde los empleados que son usuarios de la aplicación entran a escena los cuales harán uso de la aplicación sin conocer los detalles técnicos de la misma.

Al contar con una empresa dónde hay una gran multitud de los empleados hasta la fecha y, teniendo en cuenta su desconocimiento de la aplicación, aunque sí de la página web, es probable que aparezcan errores o funcionamientos no deseados, que se arreglarán gracias al mantenimiento de la aplicación.

## **5.2 DIAGRAMA DE GANTT**

Vamos a utilizar un diagrama de Gantt [40] para planificar las distintas tareas de nuestro proyecto, ofreciendo una visión general de éste y haciendo un seguimiento visual. Gráficamente suele ser un sistema con dos ejes, en el eje vertical se encuentran las tareas y en el eje horizontal el tiempo (Puede ser desde años hasta las horas, o incluso otra medida del tiempo).

Para crear diagrama de Gantt que describe todos los procesos y el tiempo de desarrollo vamos a utilizar una herramienta gratuita en línea cuyo nombre es Canva [41].

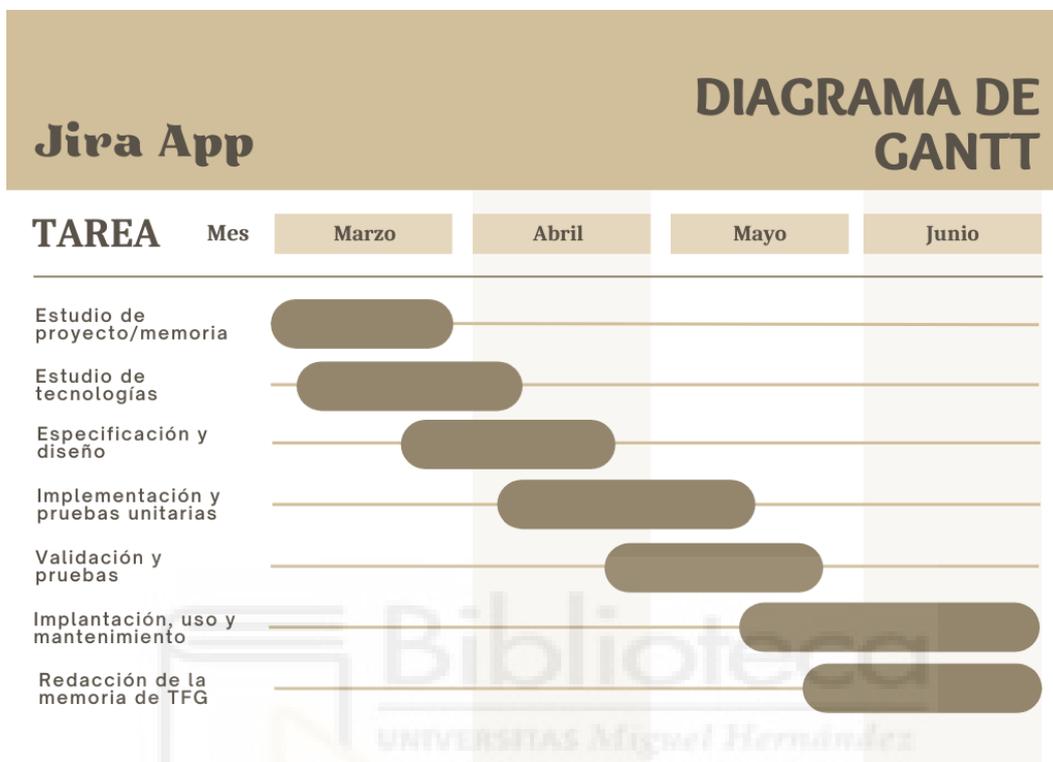


Ilustración 33 - El proceso de desarrollo del proyecto utilizando diagrama de Gantt

### 5.3 ACTORES Y CASOS DE USO

Un caso de uso [42] es una secuencia de campos y acciones de un actor que se lleva a cabo en un sistema dónde hay procedimientos sobre un objeto. Cada caso de uso proporciona al menos un escenario que muestra cómo debe comportarse e interactuar en el sistema un actor.

A continuación, se detallan dos tablas ejemplares que muestran el formato y el estilo de cómo tiene que verse un caso de uso y un actor. Dichas plantillas van a ser reutilizadas para determinar actor y los casos de uso de la aplicación.

<b>Actor</b>	<Id>	<b>&lt;Nombre descriptivo&gt;</b>
<b>Descripción</b>	<Rol del actor en el sistema>	
<b>Comentarios</b>	<Comentarios adicionales sobre el actor>	
<b>Casos de uso en los que interviene</b>	<Casos de uso en los que el actor tiene participación>	

Tabla 1 - Plantilla de actores

<b>Caso de uso</b>	<Id>	<b>&lt;Nombre descriptivo&gt;</b>	
<b>Actor principal</b>	<Actor principal>		
<b>Actores secundarios</b>	<Actores secundarios>		
<b>Descripción</b>	<Descripción, lo que hace y no hace, y cómo lo hace>		
<b>Precondición</b>	<Precondición del caso de uso>		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	<Secuencia de acciones>	
	Paso 2		
	Paso N		
<b>Postcondición</b>	<Condición posterior que debe darse en este caso de uso>		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	<Secuencia de acciones en caso de error>	
	Paso 2		
	Paso N		
<b>Comentarios</b>	<Comentarios adicionales sobre el caso de uso>		

Tabla 2 - Plantilla de casos de uso

### 5.3.1 EMPLEADO

<b>Actor</b>	A-01	<b>Empleado</b>
<b>Descripción</b>	Usuario que puede gozar de todas las opciones de la aplicación, leer, modificar y crear	
<b>Comentarios</b>	Hay que ser empleado de la empresa	
<b>Casos de uso en los que interviene</b>	<ul style="list-style-type: none"><li>- C-01: Carga de la pantalla "Perfil"</li><li>- C-02: Carga de la pantalla "Proyecto"</li><li>- C-03: Cambio de la lista de tareas</li><li>- C-04: Carga de la pantalla "Tarea"</li><li>- C-05: Crear nueva subtarea</li><li>- C-06: Carga de la pantalla "Subtarea"</li><li>- C-07: Añadir tiempo</li><li>- C-08: Cambiar estado</li></ul>	

Tabla 3 - Actor empleado



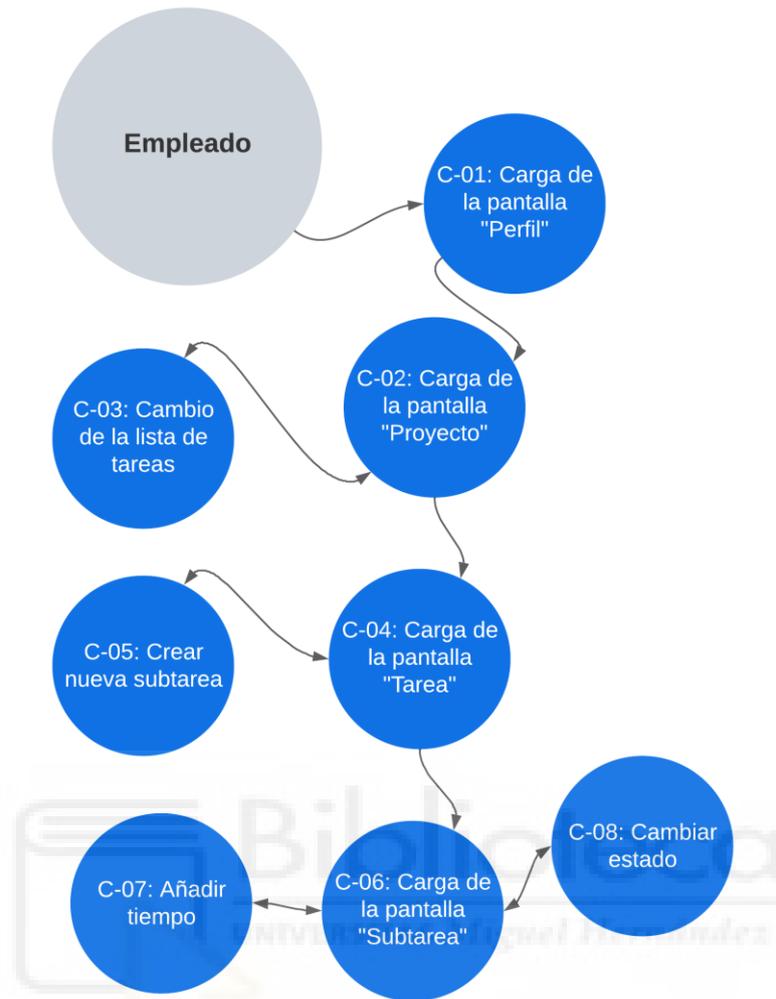


Ilustración 34 - Casos de uso del empleado

<b>Caso de uso</b>	C-01	<b>Carga de la pantalla "Perfil"</b>	
<b>Actor principal</b>	Empleado		
<b>Actores secundarios</b>	Usuario no identificado		
<b>Descripción</b>	Tras el usuario no identificado introduce los datos en la pantalla de inicio de sesión correctamente, pasa directamente a la categoría de empleado y se abre la pantalla de perfil.		
<b>Precondición</b>	Introducir los credenciales correctos en la pantalla de Inicio de sesión.		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El usuario no identificado introduce los datos	
	Paso 2	El sistema valida los datos introducidos	
	Paso 3	El sistema inserta los datos en la base de datos	
	Paso 4	El sistema devuelve respuesta positiva	
	Paso 5	Se abre la pantalla de perfil	
<b>Postcondición</b>	Si el servidor Jira de la empresa está en el estado de mantenimiento o no está disponible, el proceso se queda en la panta de carga y hasta que no haya alguna respuesta los datos se quedan grabados (Cifrados) en el dispositivo móvil		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si el sistema detecta errores en los datos o se produce un error al insertar, se anula la operación y se envía respuesta negativa	
	Paso 2	Se muestra mensaje de respuesta al usuario	
<b>Comentarios</b>	En el caso de que el servidor no esté disponible, el empleado puede cerrar la sesión en cualquier momento		

Tabla 4 - Caso de uso C-01: Carga de la pantalla "Perfil"

C-01

Dmitry Romanyutenko

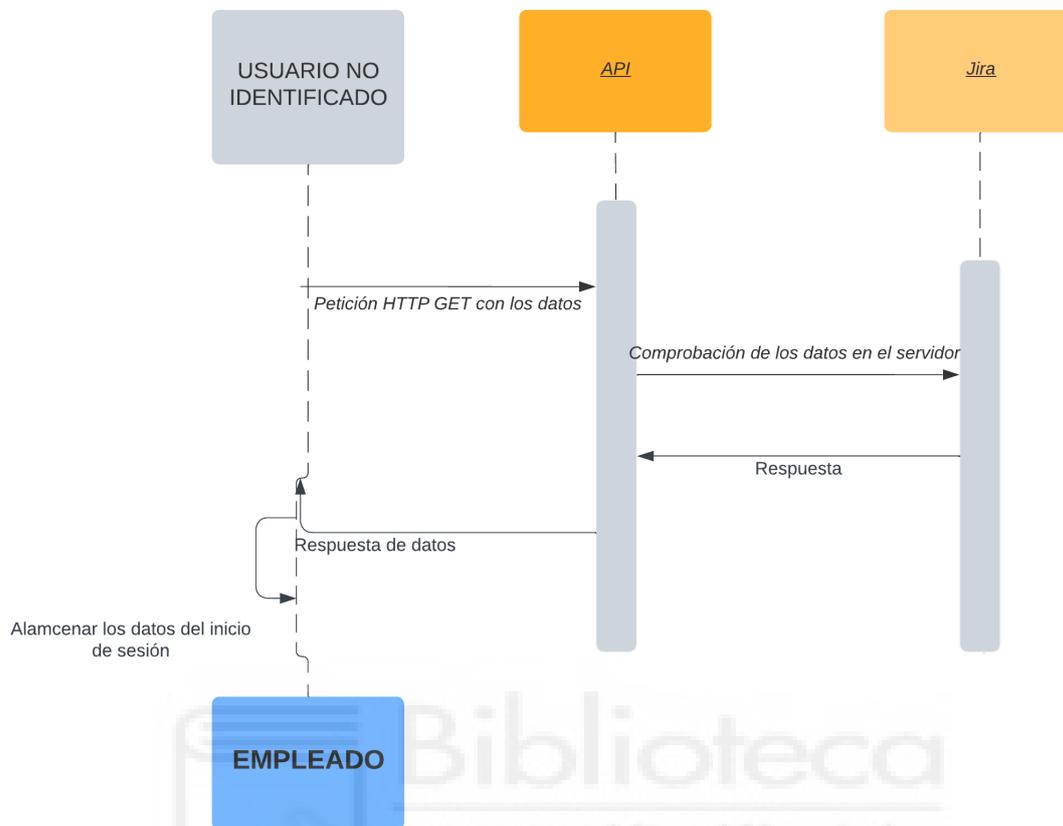


Ilustración 35 - Diagrama de secuencia C-01: Carga de la pantalla "Perfil"

<b>Caso de uso</b>	C-02	<b>Carga de la pantalla "Proyecto"</b>	
<b>Actor principal</b>	Empleado		
<b>Actores secundarios</b>	-		
<b>Descripción</b>	Obtener una lista de tareas asignadas		
<b>Precondición</b>	Tener un proyecto seleccionado anteriormente		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El empleado selecciona el proyecto	
	Paso 2	La API devuelve los datos de proyecto y la lista de las tareas asignadas	
	Paso 3	La aplicación cambia a la pantalla "Proyecto" con los datos y la lista	
<b>Postcondición</b>	Se tiene que mostrar los datos del proyecto y la lista de las tareas asignadas		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si se produce un error, se queda la pantalla de carga activa	
	Paso 2	El empleado puede cerrar la sesión	
	Paso 1	Si el empleado pincha botón de cambio de lista, aparece una lista de todas las tareas del proyecto	
	Paso 2	La API devuelve los elementos de la lista	
	Paso 3	Se pinta nueva lista con nuevos elementos	
	Paso 1	En caso de que la lista no tenga ningún elemento, aparece un mensaje indicativo	
	Paso 2	Se muestra un mensaje indicativo de que no hay tareas	
<b>Comentarios</b>	En el caso de que el servidor no esté disponible, el empleado puede cerrar la sesión en cualquier momento		

Tabla 5 - Caso de uso C-02: Carga de la pantalla "Proyecto"

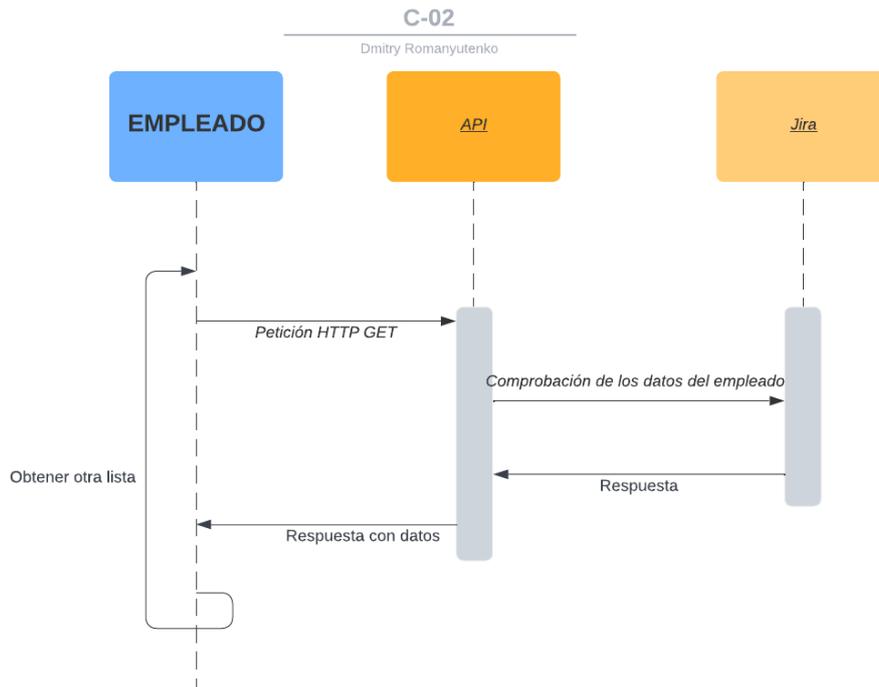


Ilustración 36 - Diagrama de secuencia C-02: Carga de la pantalla "Proyecto"



<b>Caso de uso</b>	C-03	<b>Cambio de la lista de tareas</b>	
<b>Actor principal</b>	Empleado		
<b>Actores secundarios</b>	-		
<b>Descripción</b>	Obtener o la lista de las tareas asignadas o todas las tareas del proyecto seleccionado		
<b>Precondición</b>	Tener un proyecto seleccionado anteriormente		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El empleado pincha sobre el botón de cambio de lista	
	Paso 2	La API devuelve los datos de proyecto y nueva lista	
	Paso 3	Aparece nueva lista, o bien de todas las tareas, o bien solo las tareas asignadas al empleado	
<b>Postcondición</b>	Se tiene que cambiar de lista después de la pantalla de la carga		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si se produce un error, se queda la pantalla de carga activa	
	Paso 2	El empleado puede cerrar la sesión	
<b>Comentarios</b>	En el caso de que el servidor no esté disponible, el empleado puede cerrar la sesión en cualquier momento		

Tabla 6 - Caso de uso C-03: Cambio de la lista de tareas

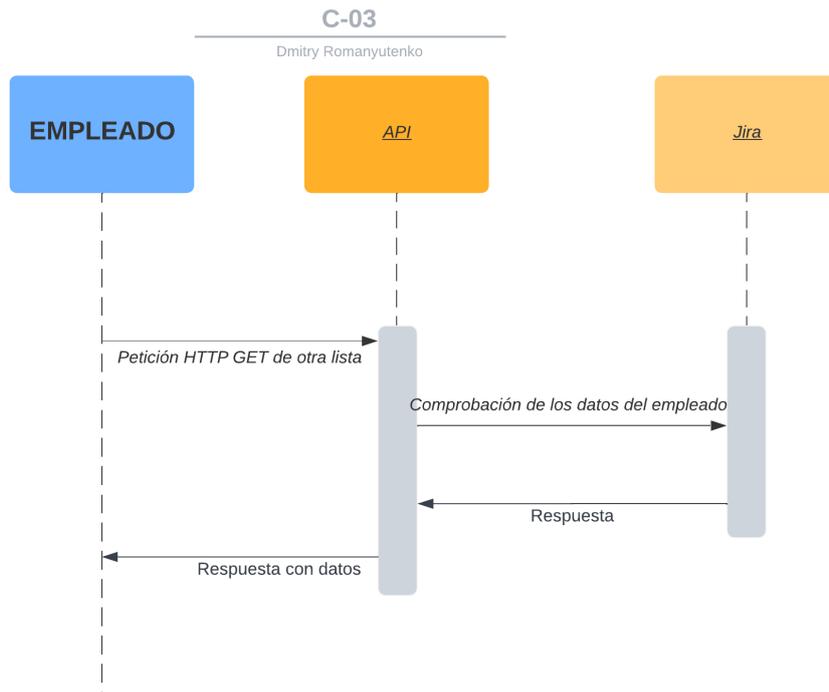


Ilustración 37 - Diagrama de secuencia C-03: Cambio de la lista de tareas



<b>Caso de uso</b>	C-04	<b>Carga de la pantalla "Tarea"</b>	
<b>Actor principal</b>	Empleado		
<b>Actores secundarios</b>	-		
<b>Descripción</b>	Obtener una lista de subtareas asignadas		
<b>Precondición</b>	Tener una tarea seleccionada previamente		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El empleado selecciona la tarea	
	Paso 2	La API devuelve los datos de tarea y la lista de las subtareas asignadas	
	Paso 3	La aplicación cambia a la pantalla "Tarea" con los datos y la lista	
<b>Postcondición</b>	Se tiene que mostrar los datos de tarea y la lista de las subtareas asignadas		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si se produce un error, se queda la pantalla de carga activa	
	Paso 2	El empleado puede cerrar la sesión	
	Paso 1	Si el empleado pincha botón de nueva subtask, se abre nueva pantalla	
	Paso 2	Abrir nueva pantalla "Crear Subtask"	
	Paso 1	En caso de que la lista no tenga ningún elemento, aparece un mensaje indicativo	
	Paso 2	Se muestra un mensaje indicativo de que no hay tareas	
<b>Comentarios</b>	En el caso de que el servidor no esté disponible, el empleado puede cerrar la sesión en cualquier momento		

Tabla 7 - Caso de uso C-04: Carga de la pantalla "Tarea"

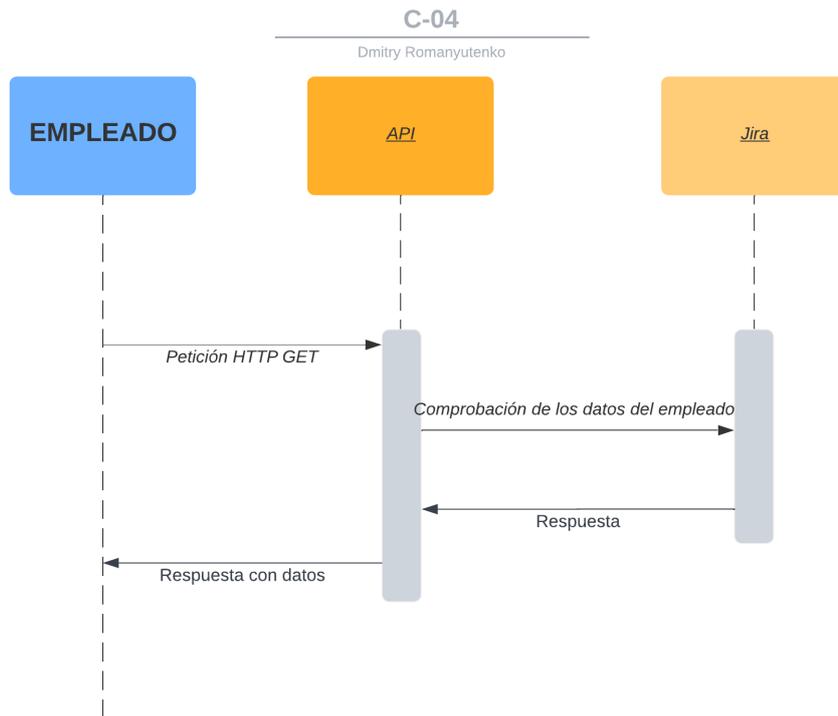


Ilustración 38 - Diagrama de secuencia C-04: Carga de la pantalla "Tarea"



<b>Caso de uso</b>	C-05	<b>Crear nueva subtarea</b>	
<b>Actor principal</b>	Empleado		
<b>Actores secundarios</b>	-		
<b>Descripción</b>	Se abre nueva pantalla para la creación de una nueva subtarea		
<b>Precondición</b>	Tener una tarea seleccionada previamente		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El empleado pincha en botón de crear nueva tarea	
	Paso 2	Se abre la pantalla con los datos de la tarea actual y se espera acción del empleado	
	Paso 3	Tras rellenar los datos que se piden el usuario pincha sobre el botón de crear	
	Paso 4	Al pinchar el botón de crear se abre una ventana de confirmación, en caso afirmativo se cierra la pantalla actual y se abre la pantalla "Tarea"	
	Paso 5	Se abre la pantalla "Tarea"	
<b>Postcondición</b>	Se tiene que introducir los datos necesarios y pinchar sobre el botón de crear		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si se produce un error, se navega a la pantalla de tareas y no se crea la subtarea	
	Paso 1	Si el empleado pincha el botón de crear tarea y le da a la opción de no, se cierra la ventana emergente	
<b>Comentarios</b>	-		

Tabla 8 - Caso de uso C-05: Crear nueva subtarea

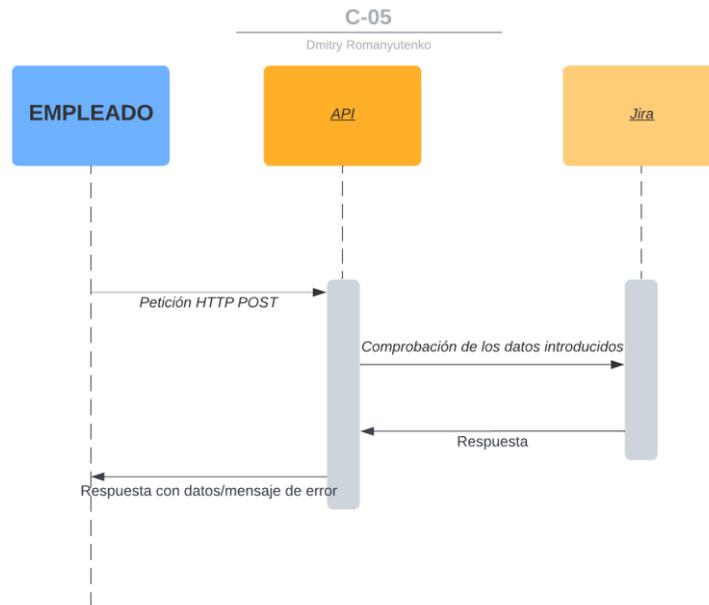


Ilustración 39 - Diagrama de secuencia C-05: Crear nueva subtarea



<b>Caso de uso</b>	C-06	<b>Carga de la pantalla "Subtarea"</b>	
<b>Actor principal</b>	Empleado		
<b>Actores secundarios</b>	-		
<b>Descripción</b>	Se carga la pantalla con todos los datos relacionados directamente con el proyecto, tarea y subtarea		
<b>Precondición</b>	Tener una subtarea seleccionada previamente		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El empleado selecciona la subtarea	
	Paso 2	La API devuelve los datos de subtarea	
	Paso 3	La aplicación cambia a la pantalla "Subtarea" con los datos	
<b>Postcondición</b>	Se tiene que mostrar los datos de subtarea		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si se produce un error, se queda la pantalla de carga activa	
	Paso 2	El empleado puede cerrar la sesión	
	Paso 1	Si el empleado pincha el botón de añadir tiempo, se abre nueva pantalla	
	Paso 2	Abrir nueva pantalla "Añadir Tiempo"	
	Paso 1	Si el empleado pincha el botón de cambiar estado, se abre nueva pantalla	
	Paso 2	Abrir nueva pantalla "Cambiar Estado"	
	Paso 1	Si el estado de subtarea es "Closed", no se puede cambiar el estado	
	Paso 2	Desaparece el botón de cambiar estado	
	Paso 1	Si la subtarea no dispone de tiempos, aparecen textos indicativos	
Paso 2	Los campos de tiempo indican que la subtarea no dispone de los tiempos agregados		
<b>Comentarios</b>	En el caso de que el servidor no esté disponible, el empleado puede cerrar la sesión en cualquier momento		

Tabla 9 - Caso de uso C-06: Carga de la pantalla "Subtarea"

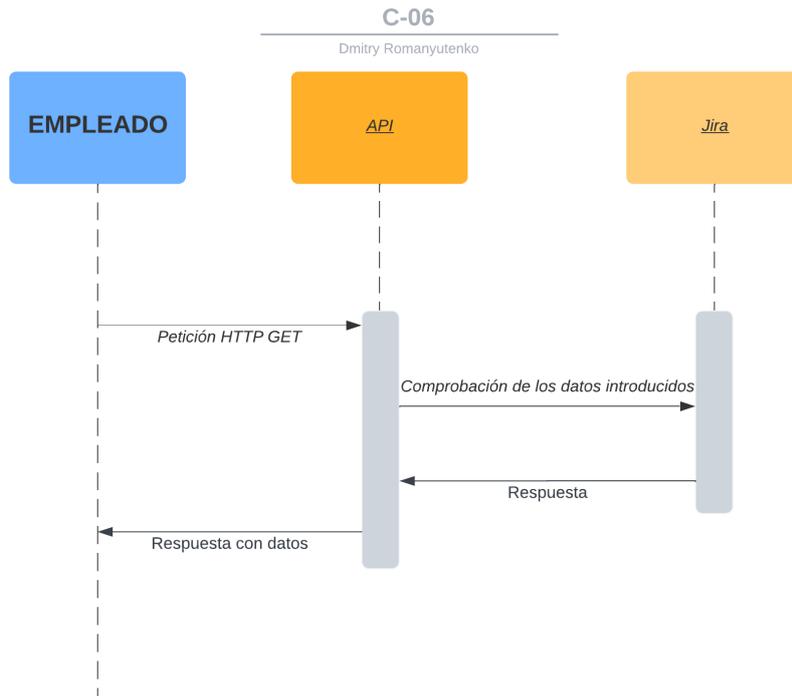


Ilustración 40 - Diagrama de secuencia C-06: Carga de la pantalla "Subtarea"



<b>Caso de uso</b>	C-07	<b>Añadir tiempo</b>	
<b>Actor principal</b>	Empleado		
<b>Actores secundarios</b>	-		
<b>Descripción</b>	Se abre nueva pantalla para añadir el tiempo a la subtarea actual		
<b>Precondición</b>	Estar en la pantalla "Subtarea"		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El empleado pincha en el botón de añadir el tiempo	
	Paso 2	Se abre la pantalla dónde hay que rellenar los campos	
	Paso 3	Tras rellenar los datos que se piden, el usuario pincha sobre el botón de añadir	
	Paso 4	Al pinchar el botón de añadir se abre una ventana de confirmación, en caso afirmativo se cierra la pantalla actual y se abre la pantalla "Subtarea"	
	Paso 5	Se abre la pantalla "Subtarea"	
<b>Postcondición</b>	Se tiene que introducir los datos necesarios y pinchar sobre el botón de añadir		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si se produce un error, se navega a la pantalla de subtarea y no se agrega el tiempo	
	Paso 1	Si el empleado pincha el botón de añadir tiempo y le da a la opción de no, se cierra la ventana emergente	
<b>Comentarios</b>	-		

Tabla 10 - Caso de uso C-07: Añadir tiempo

C-07

Dmitry Romanyutenko

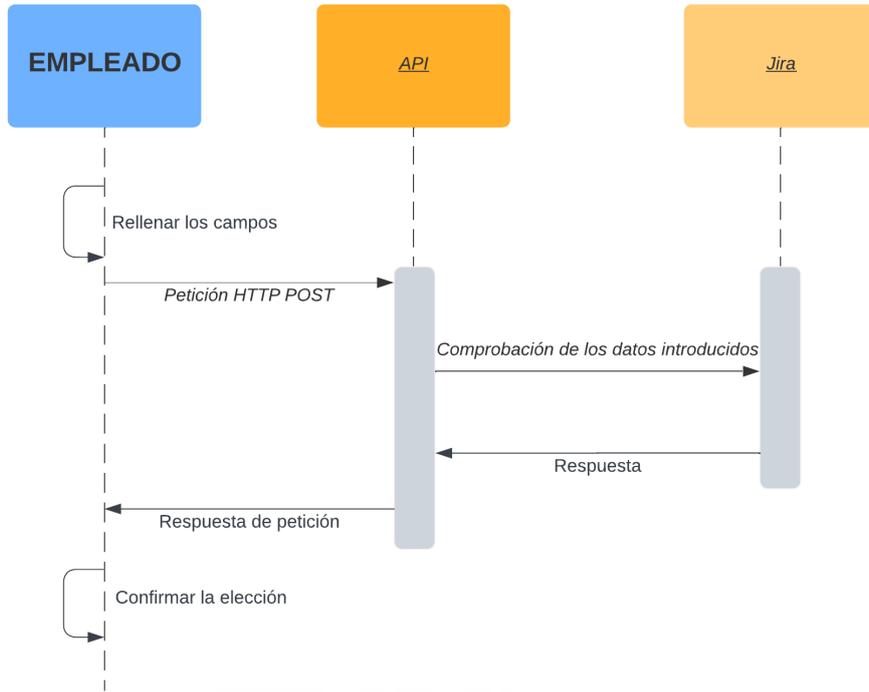


Ilustración 41 - Diagrama de secuencia C-07: Añadir tiempo



<b>Caso de uso</b>	C-08	<b>Cambiar estado</b>	
<b>Actor principal</b>	Empleado		
<b>Actores secundarios</b>	-		
<b>Descripción</b>	Se abre nueva pantalla para cambiar el estado de la subtarea actual		
<b>Precondición</b>	Estar en la pantalla "Subtarea"		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El empleado pincha en el botón de cambiar el estado	
	Paso 2	Se abre la pantalla dónde hay que seleccionar el estado	
	Paso 3	Tras seleccionar el estado, el usuario pincha sobre el botón de cambiar	
	Paso 4	Al pinchar el botón de cambiar se abre una ventana de confirmación, en caso afirmativo se cierra la pantalla actual y se abre la pantalla "Subtarea"	
	Paso 5	Se abre la pantalla "Subtarea"	
<b>Postcondición</b>	Se tiene que seleccionar el estado deseado y pinchar sobre el botón de cambiar		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si se produce un error, se navega a la pantalla de subtarea y no se cambia el estado	
	Paso 1	Si el empleado pincha el botón de cambiar el estado y le da a la opción de no, se cierra la ventana emergente	
<b>Comentarios</b>	-		

Tabla 11 - Caso de uso C-08: Cambiar estado

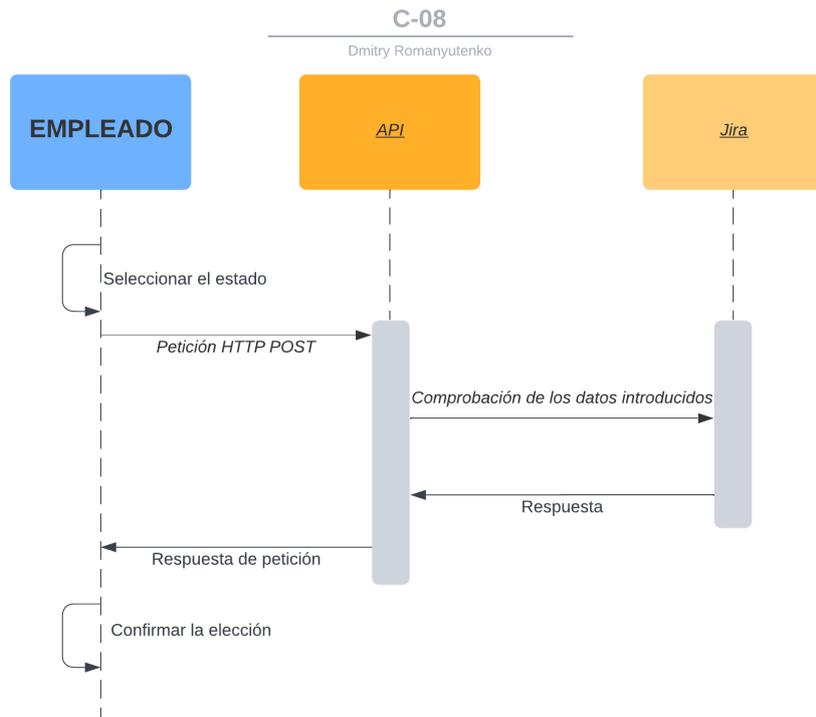


Ilustración 42 - Diagrama de secuencia C-08: Cambiar estado

### 5.3.2 USUARIO NO IDENTIFICADO

<b>Actor</b>	A-02	<b>Usuario no identificado</b>
<b>Descripción</b>	Toda persona que entra a la aplicación por primera vez o no tenga credenciales de acceso	
<b>Comentarios</b>	-	
<b>Casos de uso en los que interviene</b>	- C-09: Acceso web	

Tabla 12 - Actor usuario no identificado

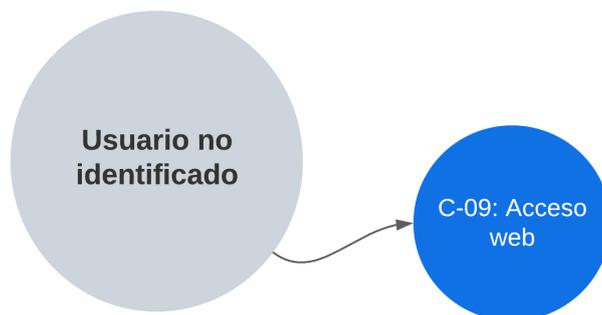


Ilustración 43 - Caso de uso del usuario no identificado

<b>Caso de uso</b>	C-09	<b>Acceso web</b>	
<b>Actor principal</b>	Usuario no identificado		
<b>Actores secundarios</b>	-		
<b>Descripción</b>	En la pantalla de login si el usuario pincha sobre el botón de obtener la clave API, se abre el navegador por defecto en el enlace correspondiente		
<b>Precondición</b>	-		
<b>Secuencia normal</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	El empleado pincha en el botón de obtener la clave	
	Paso 2	Se abre el navegador en el url preasignado	
<b>Postcondición</b>	Se tienen que realizar operaciones necesarias y tras finalizar cerrar navegador y volver a la aplicación		
<b>Excepciones</b>	<b>Paso</b>	<b>Acción</b>	
	Paso 1	Si el dispositivo no se encuentra conectado al internet, la página web no se carga	
<b>Comentarios</b>	-		

Tabla 13 - Caso de uso C-09: Acceso web

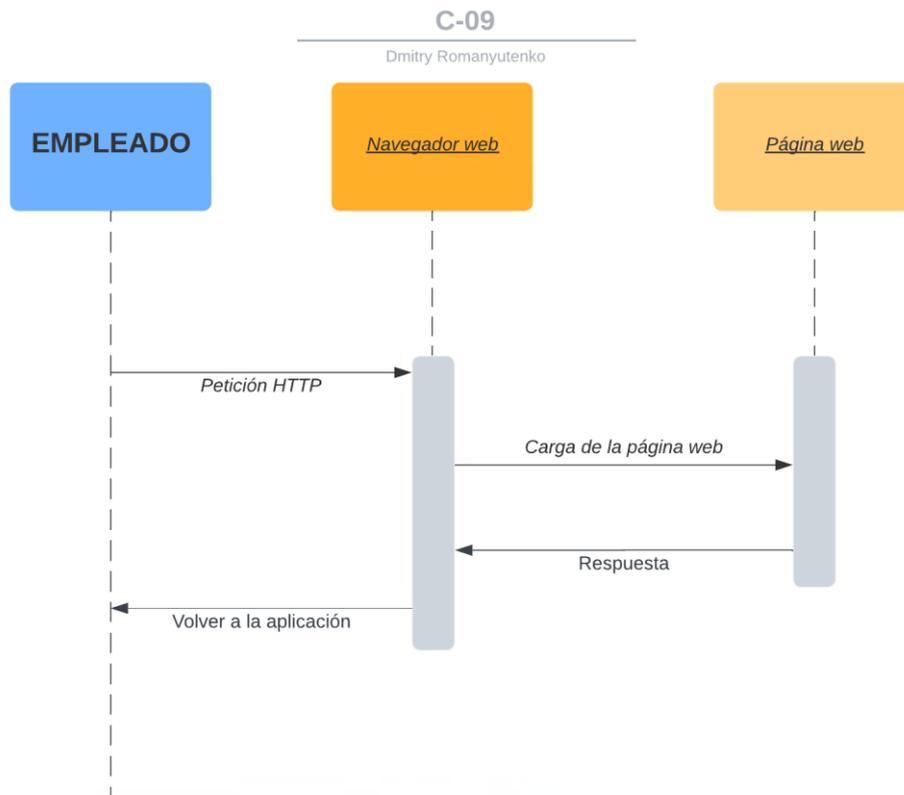


Ilustración 44 - Diagrama de secuencia C-09: Acceso web

## 5.4 PROTOTIPOS

Antes de empezar a implementar el diseño de la aplicación, hay que tener una guía ilustrativa para tener claro si el diseño es coherente. Para realizar dicho diseño vamos a seguir una estructura de tipo Wireframe a mano [43].

### 5.4.1 WIREFRAME

El prototipo de las distintas pantallas de la aplicación se va a dibujar en un papel cuadriculado para tenerlo lo más entendible posible y con un lápiz para poder borrar con la goma en caso de que algo no sea coherente.

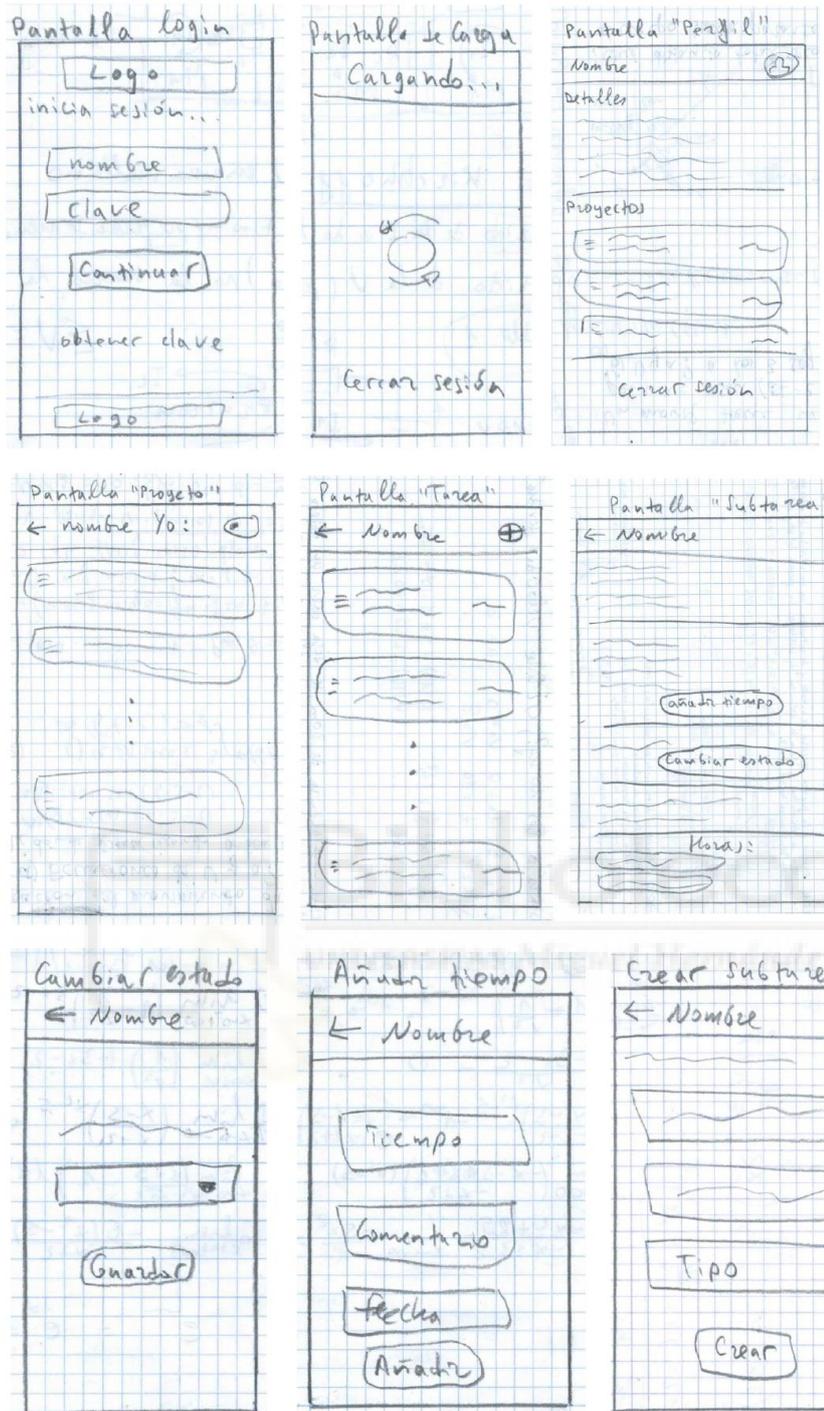


Ilustración 45 - Pantallas Wireframe

## 5.4.2 RESULTADO FINAL

Una vez dibujados los wireframes de las pantallas, se puede codificar el diseño de las pantallas. Los wireframes y el resultado final pueden ser algo diferentes debido a que no siempre un wireframe es una solución ideal. En el siguiente apartado se muestra una comparativa.

## 5.4.3 COMPARATIVA

A continuación, se muestra una comparativa entre los wireframes y el resultado final. Las imágenes a la izquierda son wireframe y a la derecha son capturas de pantalla de la aplicación final.

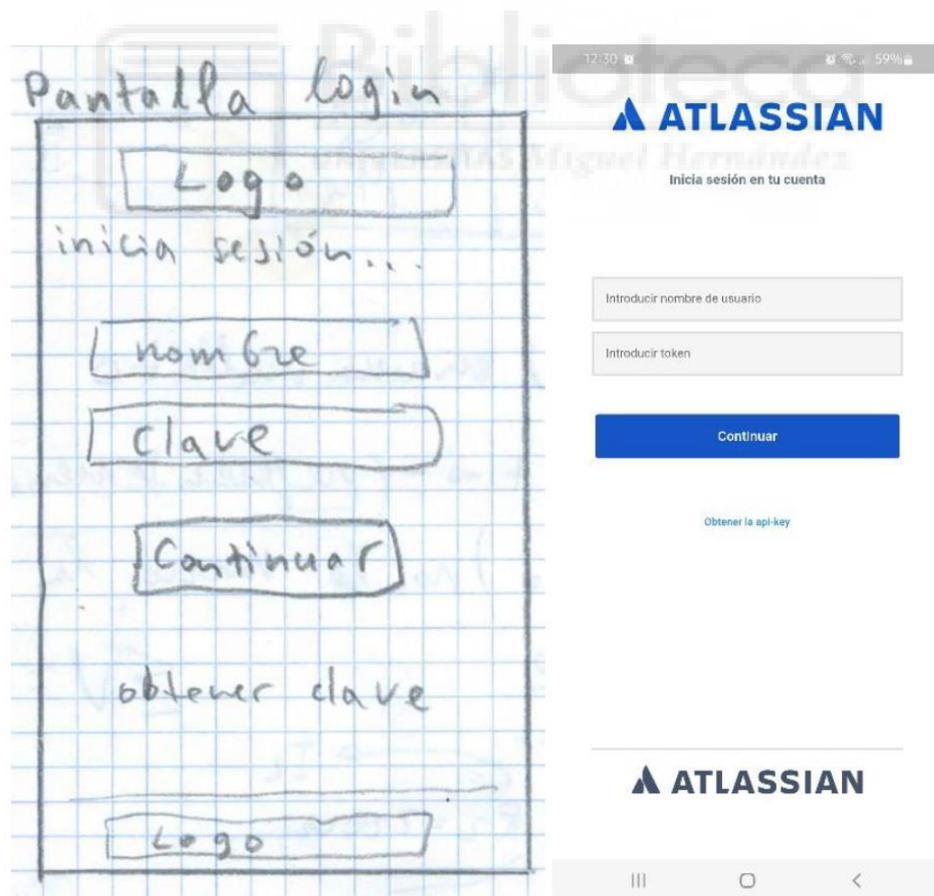


Ilustración 46 - Pantalla de login

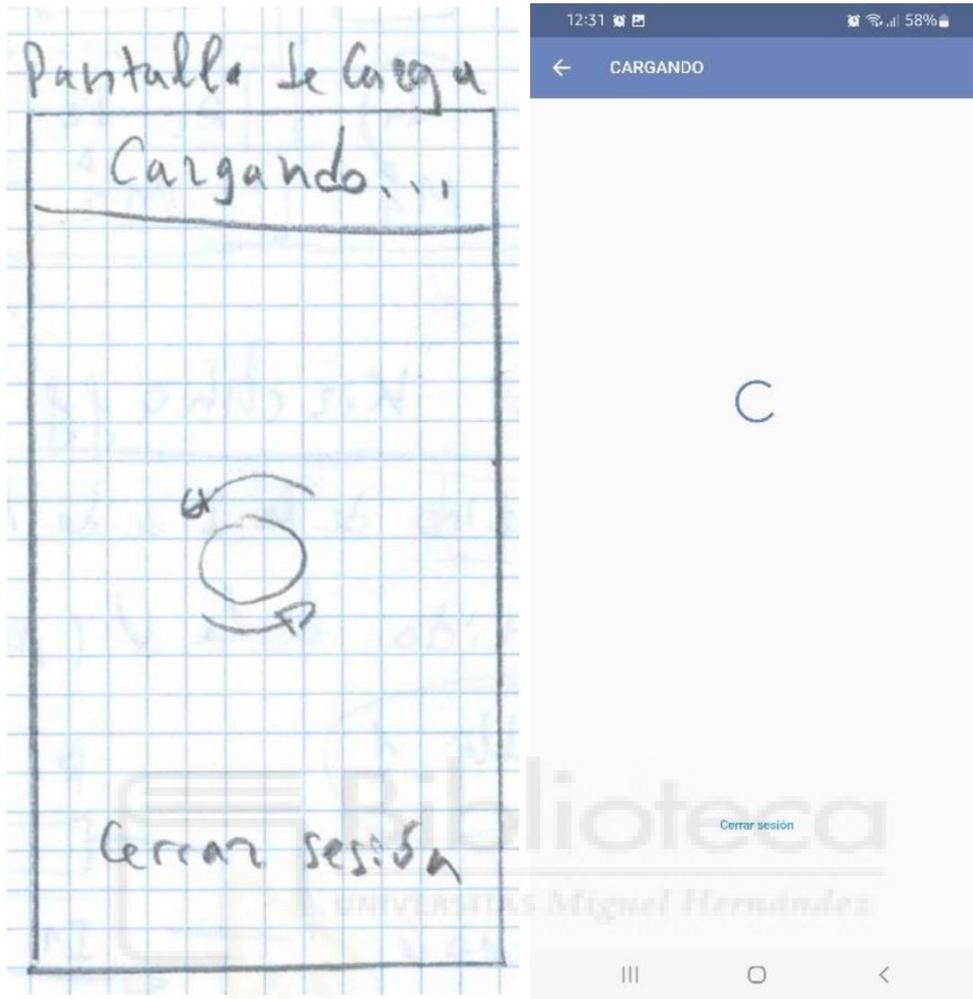


Ilustración 47 - Pantalla de carga

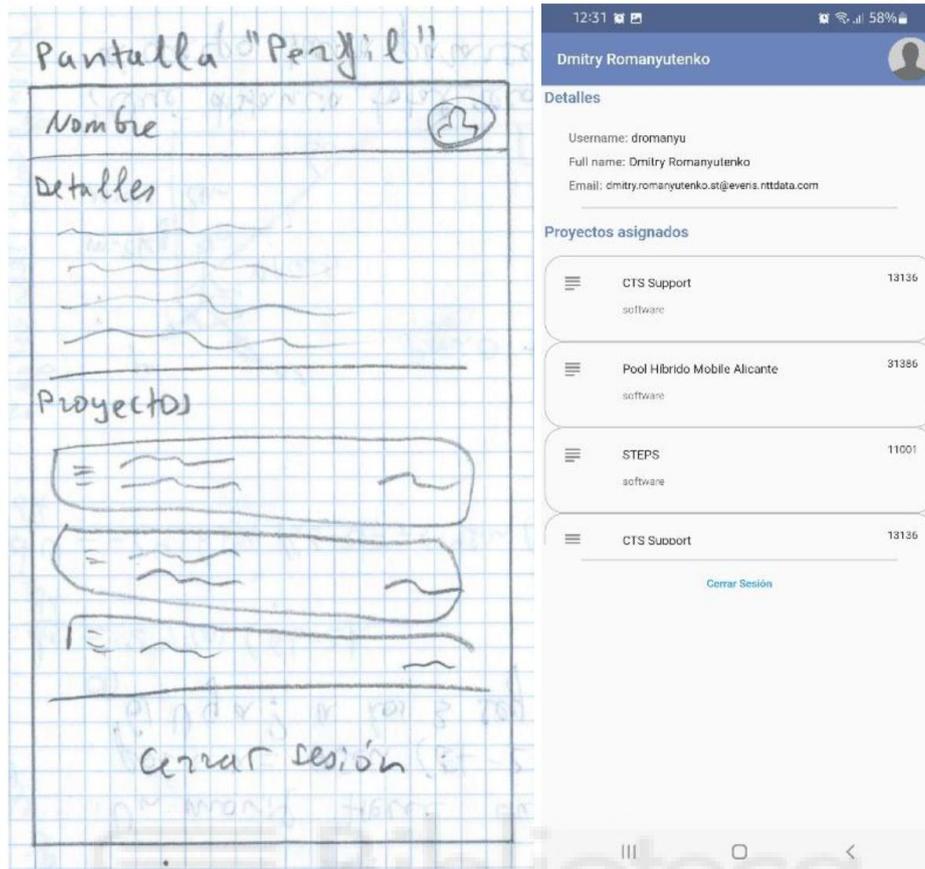


Ilustración 48 - Pantalla "Perfil"

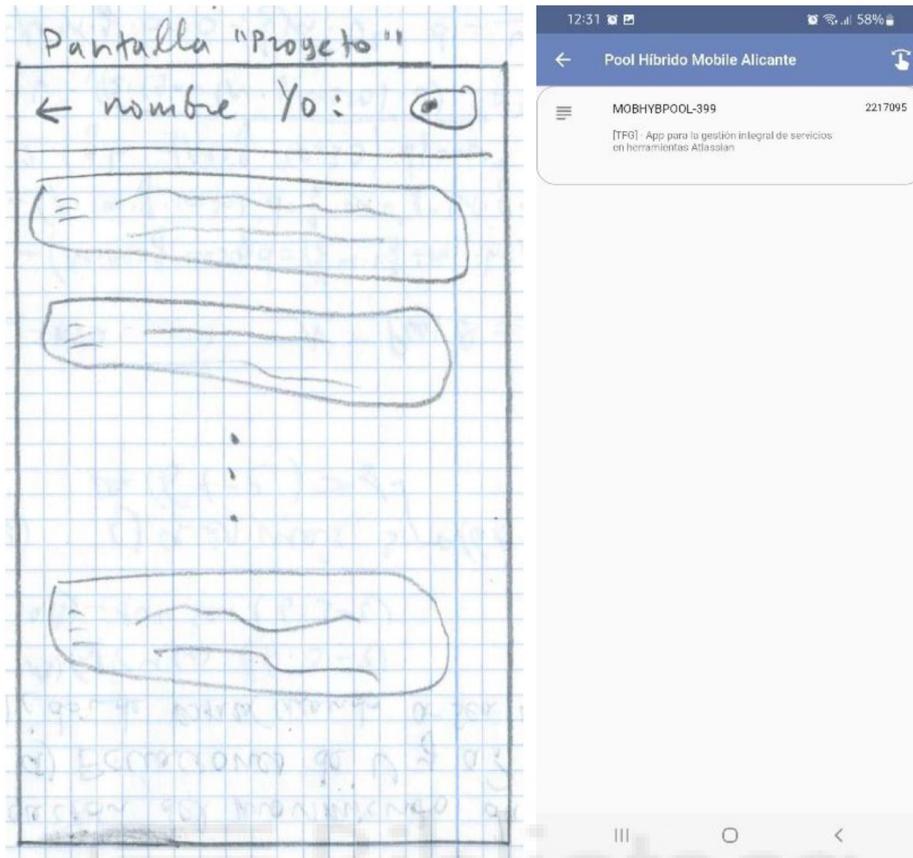


Ilustración 49 - Pantalla "Proyecto"

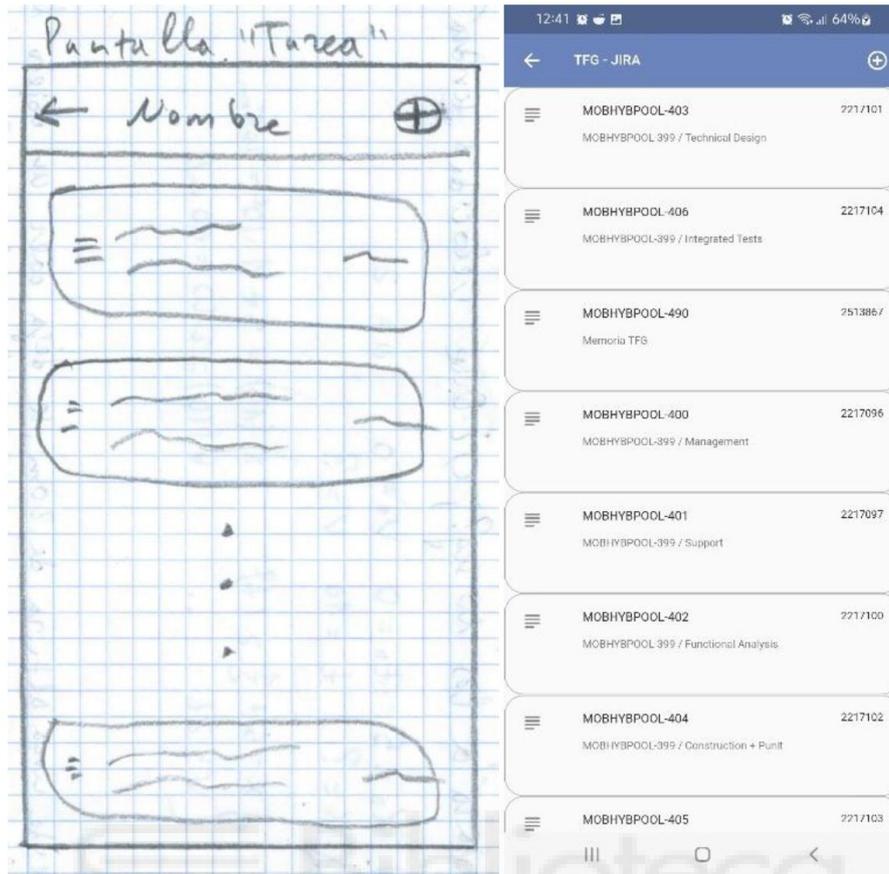


Ilustración 50 - Pantalla "Tarea"

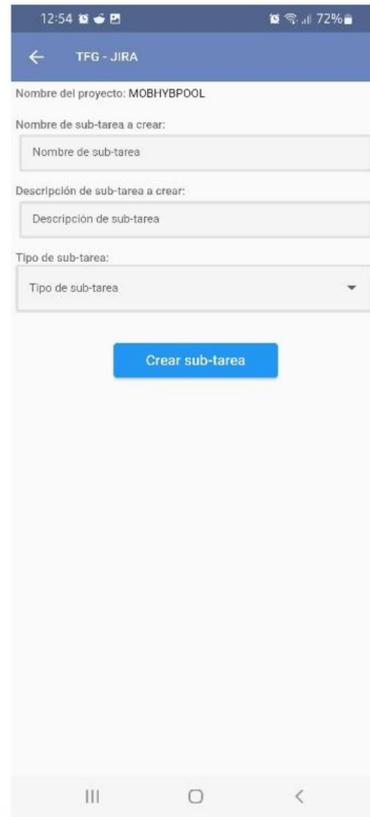
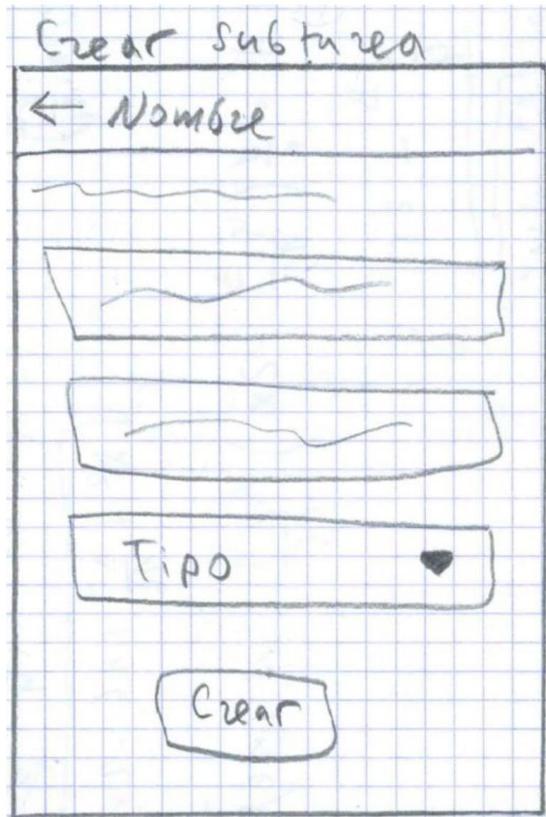


Ilustración 51 - Pantalla de crear subtarea

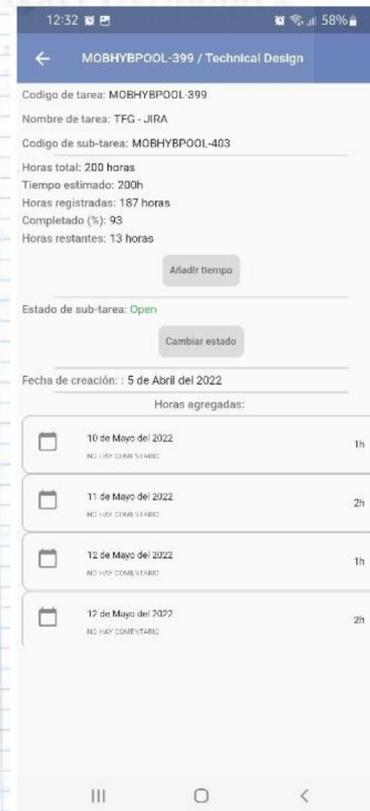
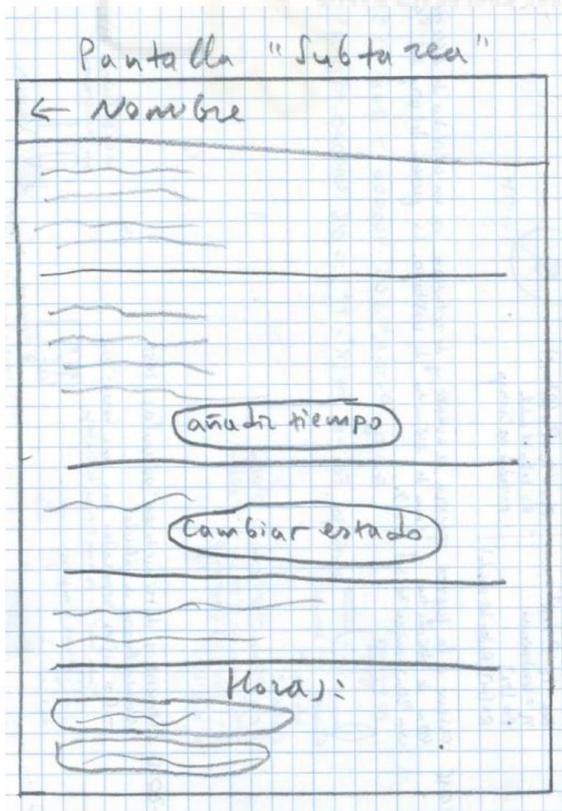


Ilustración 52 - Pantalla "Subtarea"

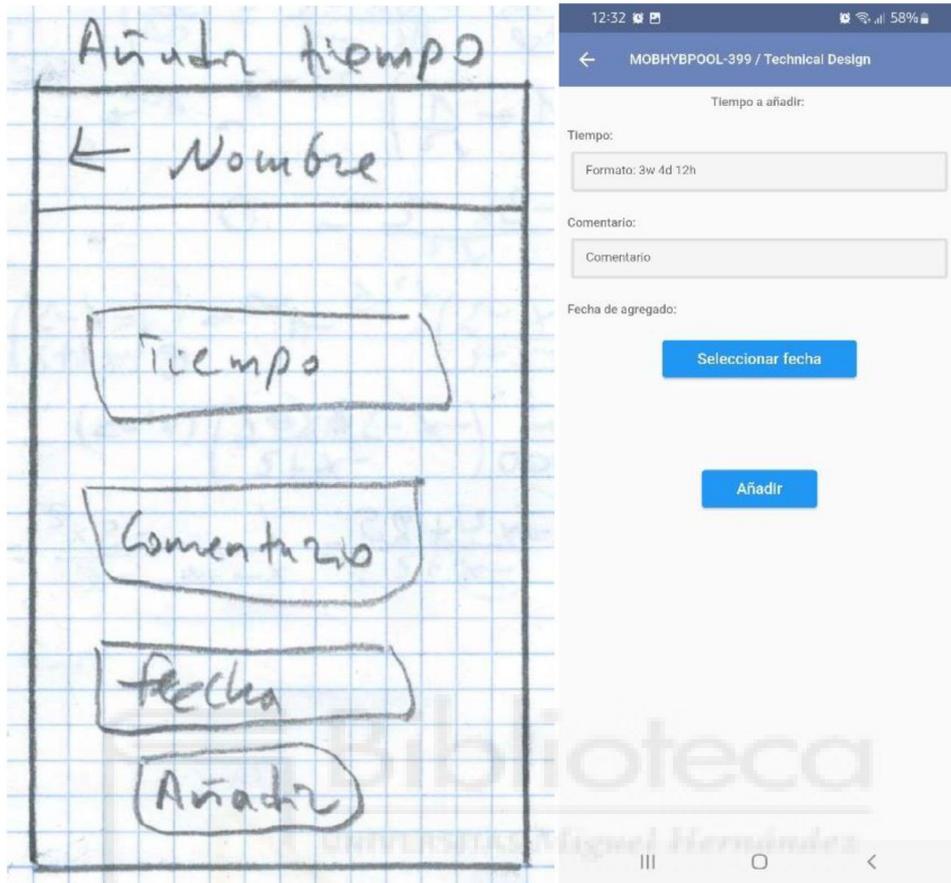


Ilustración 53 - Pantalla "Añadir Tiempo"

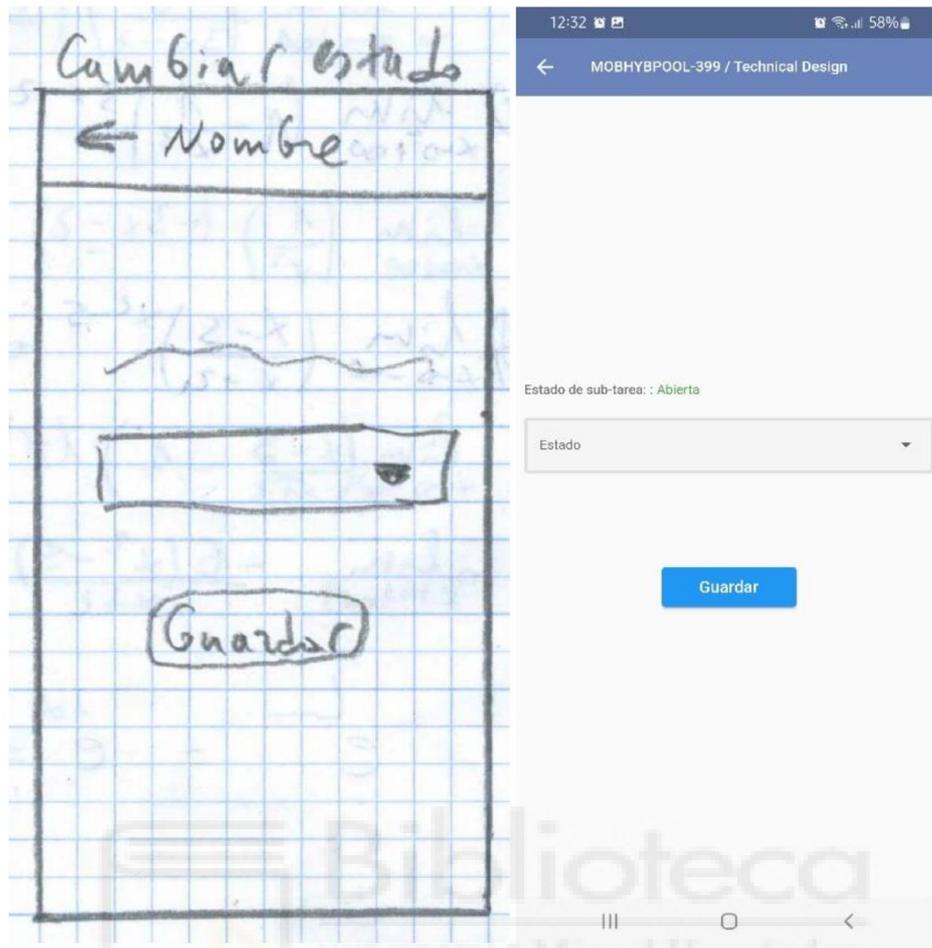


Ilustración 54 - Pantalla "Cambiar Estado"

## 5.5 PROBLEMAS

Los problemas a la hora de desarrollar la aplicación que fueron encontrados son dos y no fue fácil resolverlos ya que llevaron un tiempo.

- **API:** La API que hay que usar es propia de la empresa y no está bien elaborada como por ejemplo BBDD de Google, Firebase [44]. Había que hacer muchísimas pruebas en Postman para poder conseguir un resultado útil y enviar datos a la página. La estructura del Json al parecer o no está acabada o no se entienden los fines ya que hay muchos apartados únicos con nombres propios y sin sentido ya que casi todos están vacíos. Muchos

apartados únicos obligan a que cada apartado sea una propia clase sin atributos, ni métodos, etc. Para resolver este problema se ha optado la opción de navegar de manera "específica". Es decir, en vez de almacenar todas las clases "innecesarias". Tener una o varias clases que necesito y usar atributos necesarios.

- **Token API:** Primer problema real ha surgido a la hora del inicio de sesión del usuario final que en este caso sólo puede ser un empleado de la empresa. Normalmente para poder entrar al sistema se requiere un correo electrónico y la contraseña asignada en la plataforma. En nuestro caso sólo se puede entrar mediante un nombre de usuario que solamente se puede averiguar entrando a la página al apartado del perfil. Es decir, antes de poder usar la aplicación por primera vez, hay que haber entrado al perfil para poder averiguar el nombre del usuario. Segundo criterio para entrar es la "API Key". API Key es una clave única que se puede generar también desde el apartado de perfil de la plataforma. A la clave se puede asignar una fecha de caducidad y dos tipos de permisos, uno de sólo lectura y otro de lectura y escritura. Para poder solicitar dicha clave, previamente es necesario pedir los permisos a la empresa para que te aparezca el apartado de las claves API. Es decir, es un problema gordo para el usuario. Hubiesemos podido solicitar un procedimiento a la empresa para que crea nueva manera de acceder con e-mail y la contraseña. Pero el gerente ha optado por la opción de dejarlo tal como está por ahora.

## 5.6 EJEMPLO FUNCIONAL

En este apartado se van a mostrar algunas pantallas de la aplicación funcional con la descripción correspondiente.

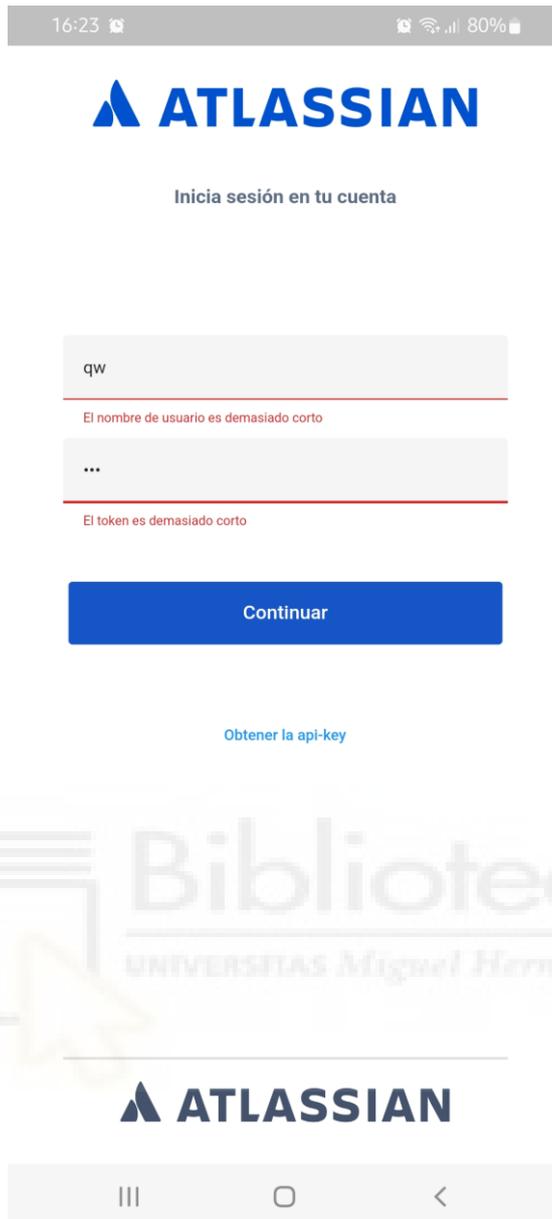


Ilustración 55 - Captura de pantalla de la pantalla de inicio de sesión

La pantalla de inicio de sesión obliga a que el usuario siga un formato de nombre y del token, el nombre tiene que contener al menos 5 caracteres y el token por lo menos 15. Si el usuario no sigue el formato, no se va a lanzar una petición HTTP de acceso a la aplicación.



Ilustración 56 - Captura de pantalla "Proyecto"

La pantalla de proyecto es similar a la pantalla de tarea. Básicamente por primera vez que acceder a esta pantalla, salen tareas asignadas a ti y tienes las posibilidad de cambiar el filtro para que aparezcan todas las tareas del proyecto actual. Arriba aparece el mensaje indicativo de la lista de tareas actual, en este caso sólo las tareas asignadas al empleado actual.

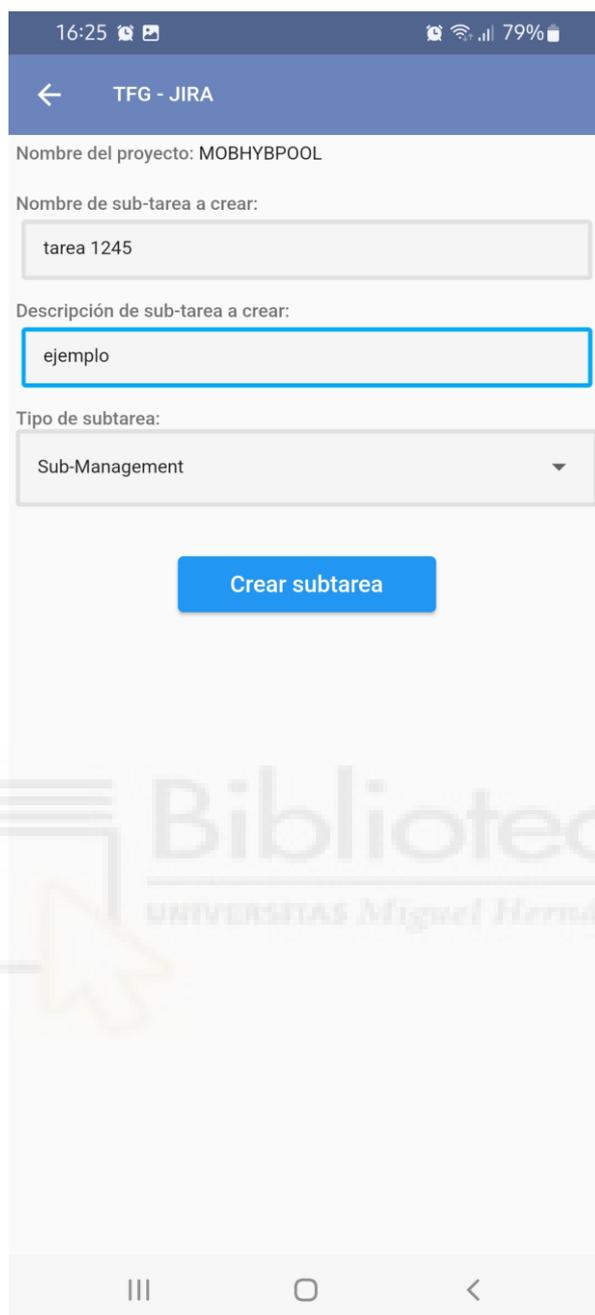


Ilustración 57 - Captura de la pantalla de nueva subtarea

En la pantalla de nueva subtarea el usuario tiene que introducir el nombre de subtarea por obligación, la descripción y seleccionar el tipo de subtarea de una lista desplegable.

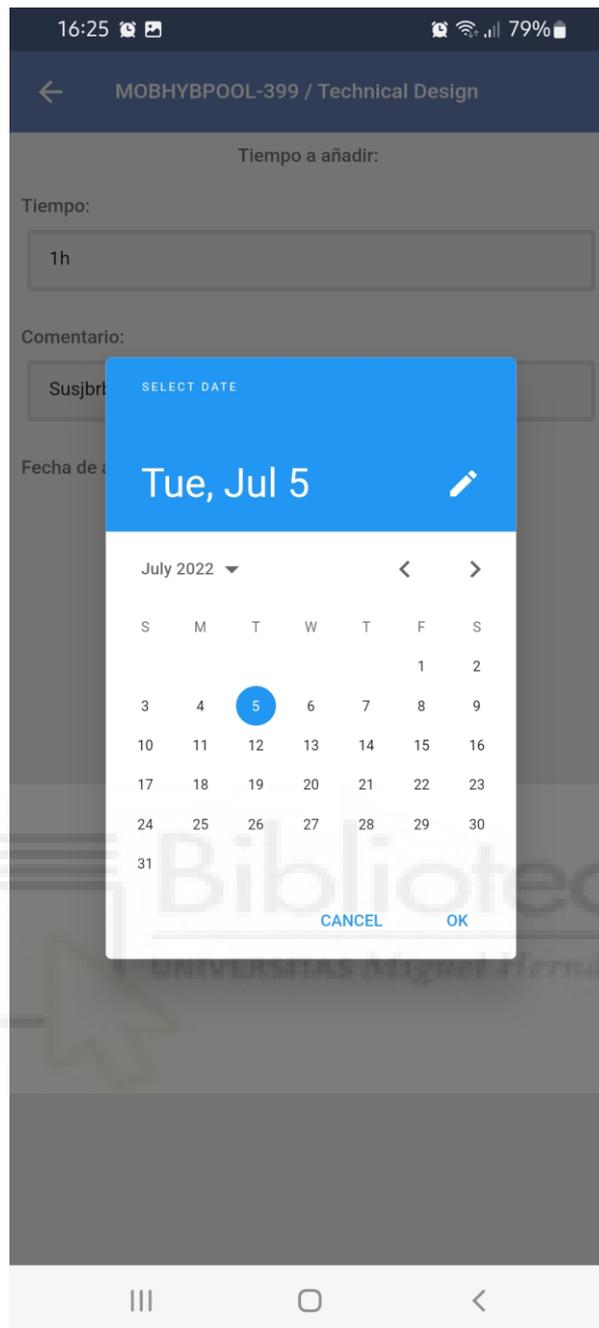


Ilustración 58 - Captura de la pantalla de agregar el tiempo

En la pantalla de agregar el tiempo a una subtarea seleccionada el empleado tiene que indicar el tiempo a añadir, un comentario y seleccionar el día en que quiere agregar el tiempo, por defecto si no se marca ningún día, se envía el día actual.



# CAPÍTULO 6

## CONCLUSIONES Y PROPUESTAS

---

### 6.1 CONCLUSIONES

Durante los últimos cuatro meses se ha desarrollado una aplicación para dispositivos móviles cuyo sistema operativo es Android con una tecnología híbrida siguiendo los principios de la ingeniería del software. La aplicación ha sido terminada con éxito ya que ha cumplido todos los requisitos propuestos por la empresa. Además, ha recibido múltiples valoraciones positivas por parte del equipo que ha estado siguiendo el desarrollo del proyecto durante todo el tiempo.

Al ser una tecnología híbrida, no llevaría nada de tiempo implementar esta aplicación para otros tipos de dispositivos como iOS, Windows Phone o Web. De esta manera, todos los empleados podrían darle el uso a la aplicación.

Se ha desarrollado una aplicación funcional que al finalizar el desarrollo va a utilizarse entre los empleados de la empresa. La van a poder gozar los usuarios que dispongan de un dispositivo móvil con el sistema operativo de Android.

En cuanto a como meta persona, he logrado desarrollar un proyecto de mucha carga casi en solitario. El framework como Flutter, nunca lo había usado anteriormente, sin embargo, se parece mucho a Java y a C++, y de éstos sí que tengo un dominio. Estoy muy satisfecho de finalizar el trabajo con éxito y de

poder estar incorporado en una empresa real y obtener una experiencia valiosa que me servirá en el ámbito profesional.

## **6.2 PROPUESTAS DE FUTURO**

La aplicación cumple con todos los objetivos que ha propuesto la empresa en cuanto a la funcionalidad y el diseño. Sin embargo, se puede hacer muchas mejoras en el futuro ya que quedan muchas operaciones pendientes agregar.

Posibles mejoras podrían ser añadir opción de modificar más campos de una subtarea, poder crear una tarea, aunque para cada tipo de tarea se requiere diferentes campos obligatorios y eso llevaría muchísimo tiempo. Otra mejora sería asignar a un empleado a una tarea, es decir, saldría una lista de empleados posibles a asignar. Y la última idea que podría ser es poder buscar una tarea o subtarea por nombre, aunque la API no ofrece dicha opción.

El proyecto que he estado creando le pertenece a la empresa por completo, por lo tanto, en el futuro, se le asignará dicho proyecto a uno o varios empleados para mejorarlo. Los empleados podrán partir de la base y eso ahorraría un costo y el tiempo que es un recurso muy valioso. Como se ha desarrollado con una tecnología híbrida, futuros desarrolladores podrán añadir código para que pueda ejecutarse en SO de iOS, aunque en este caso no era el objetivo el desarrollo para otros sistemas operativos.



# CAPÍTULO 7

## BIBLIOGRAFÍA

---

### 7.1 FUENTES BIBLIOGRÁFICAS

#### Bibliografía

- [1] «Udemy,» [En línea]. Available: <https://www.udemy.com/es/>. [Último acceso: 28 Junio 2022].
- [2] «Atlassian - Jira software,» [En línea]. Available: <https://www.atlassian.com/software/jira>. [Último acceso: 2022 Junio 2022].
- [3] NTT Data, «NTT Data,» [En línea]. Available: <https://es.nttdata.com/about-us>. [Último acceso: 28 Junio 2022].
- [4] «Flutter,» [En línea]. Available: <https://flutter.dev/>. [Último acceso: 28 Junio 2022].
- [5] Ditrendia, «Todas las estadísticas sobre móviles que deberías conocer,» [En línea]. Available: <https://mktefa.ditrendia.es/blog/todas-las-estad%C3%ADsticas-sobre-m%C3%B3viles-que-deber%C3%ADas-conocer-mwc19#:~:text=El%2068%25%20de%20la%20poblaci%C3%B3n,penetraci%C3%B3n%20tan%20solo%20del%2053%25..> [Último acceso: 25 mayo 2022].
- [6] Atlassian, «Atlassian jira mobile app,» [En línea]. Available: <https://www.atlassian.com/software/jira/mobile-app>. [Último acceso: 28 Junio 2022].
- [7] Atlassian, «Google Play - Jira Data Center and Server,» 22 Julio 2019. [En línea]. Available: <https://play.google.com/store/apps/details?id=com.atlassian.jira.server>. [Último acceso: 28 Junio 2022].

- [8] flutter.dev, «pub.dev,» [En línea]. Available: [https://pub.dev/packages/cupertino\\_icons](https://pub.dev/packages/cupertino_icons). [Último acceso: 13 Abril 2022].
- [9] V. Diví, «inLab FIB,» [En línea]. Available: <https://inlab.fib.upc.edu/es/blog/que-es-el-lenguaje-de-programacion-dart>. [Último acceso: 20 Mayo 2022].
- [10] Altexsoft, «The Good and the Bad of Flutter App Development,» 5 Mayo 2021. [En línea]. Available: <https://www.altexsoft.com/blog/engineering/pros-and-cons-of-flutter-app-development/>. [Último acceso: 27 Mayo 2022].
- [11] Dart, «Sound null safety,» [En línea]. Available: <https://dart.dev/null-safety>. [Último acceso: 29 Abril 2022].
- [12] Dart & Flutter, «pub.dev,» [En línea]. Available: <https://pub.dev/>. [Último acceso: 28 Junio 2022].
- [13] pub.dev, «Provider,» [En línea]. Available: <https://pub.dev/packages/provider>. [Último acceso: 22 Abril 2022].
- [14] A. Alonso, «Flutter y el patrón Provider,» [En línea]. Available: <https://dev.to/hextiandro/flutter-y-el-patron-provider-33c3>. [Último acceso: 20 Abril 2022].
- [15] Flutter, «Flutter Secure Storage,» [En línea]. Available: [https://pub.dev/packages/flutter\\_secure\\_storage](https://pub.dev/packages/flutter_secure_storage). [Último acceso: 5 Mayo 2022].
- [16] flutterchina.club, «Dio,» [En línea]. Available: <https://pub.dev/packages/dio>. [Último acceso: 29 Abril 2022].
- [17] Flutter, «URL Launcher,» [En línea]. Available: [https://pub.dev/packages/url\\_launcher](https://pub.dev/packages/url_launcher). [Último acceso: 25 Junio 2022].
- [18] waterydesert.com, «pub.dev,» [En línea]. Available: [https://pub.dev/packages/loading\\_animation\\_widget](https://pub.dev/packages/loading_animation_widget). [Último acceso: 30 Junio 2022].
- [19] Google Fuchsia, «Fuchsia,» [En línea]. Available: <https://fuchsia.dev/>. [Último acceso: 29 Junio 2022].
- [20] K. Bircan, «Cross-Platform App Development with Flutter, Xamarin, React Native,» 6 Agosto 2017. [En línea]. Available:

- <https://www.slideshare.net/KorhanBircan/crossplatform-app-development-with-flutter-xamarin-react-native>. [Último acceso: 29 Junio 2022].
- [21] Altexsoft, «Flutter vs Xamarin Cross-Platform Mobile Development Compared,» 26 Septiembre 2018. [En línea]. Available: <https://www.altexsoft.com/blog/engineering/flutter-vs-xamarin-cross-platform-mobile-development-compared/>. [Último acceso: 29 Junio 2022].
- [22] Altexsoft, «Flutter vs Xamarin Cross-Platform Mobile Development Compared,» [En línea]. Available: <https://www.altexsoft.com/blog/engineering/flutter-vs-xamarin-cross-platform-mobile-development-compared/>. [Último acceso: 29 Junio 2022].
- [23] Surf Dev, «Flutter VS Apache Cordova: Choosing the Best Technology for Cross-Platform Development,» [En línea]. Available: <https://surf.dev/flutter-vs-apache-cordova/>. [Último acceso: 29 Junio 2022].
- [24] Fireart, «React Native vs Flutter: Which one is better for 2022?,» 14 Abril 2022. [En línea]. Available: <https://fireart.studio/blog/flutter-vs-react-native-what-app-developers-should-know-about-cross-platform-mobile-development/#:~:text=Flutter%20is%20easier%20to%20use,appear%20in%20the%20launch%20app..> [Último acceso: 29 Junio 2022].
- [25] Lucid, «Lucidchart,» Lucid, [En línea]. Available: <https://www.lucidchart.com/pages/es>. [Último acceso: 13 Abril 2022].
- [26] Y. FERNÁNDEZ, «Xataka,» Xataka Basics, 23 Agosto 2019. [En línea]. Available: <https://www.xataka.com/basics/api-que-sirve>. [Último acceso: 3 Mayo 2022].
- [27] D. A., «¿Qué es JSON?,» 7 Marzo 2022. [En línea]. Available: <https://www.hostinger.es/tutoriales/que-es-json>. [Último acceso: 6 Abril 2022].
- [28] G. Romero, «Cómo realizar pruebas automatizadas con Postman,» [En línea]. Available: <https://www.encora.com/es/blog/como-realizar-pruebas-automatizadas-con-postman#:~:text=Postman%20es%20una%20aplicaci%C3%B3n%20que%20nos%20permite%20testear%20APIs%20a,tiempo%20de%20ejecuci%C3%B3n%20de%20pruebas..> [Último acceso: 21 Mayo 2022].

- [29] Quicktype, «Quicktype.io,» [En línea]. Available: <https://quicktype.io/>. [Último acceso: 11 Abril 2022].
- [30] ROSEPAC, «Mejores EDITORES de TEXTO Gratis para PROGRAMAR,» [En línea]. Available: <https://ciberninjas.com/mejores-editores-texto/>. [Último acceso: 7 Abril 2022].
- [31] Atlassian, «Qué es Git,» [En línea]. Available: <https://www.atlassian.com/es/git/tutorials/what-is-git>. [Último acceso: 15 Abril 2022].
- [32] Cleventy, «EcuRed,» 5 Marzo 2020. [En línea]. Available: <https://cleventy.com/que-es-git-flow-y-como-funciona/>. [Último acceso: 26 Abril 2022].
- [33] rchavez, «Tutorial: Uso de SourceTree para manejar Github,» 2019 Abril 2019. [En línea]. Available: <https://soporte.code-fu.net.ni/sourcetree-tutorial/#:~:text=%C2%BFQu%C3%A9%20es%20SourceTree%3F,usuario%20que%20provee%20la%20aplicaci%C3%B3n..> [Último acceso: 27 Abril 2022].
- [34] Android Studio, «Create and manage virtual devices,» [En línea]. Available: <https://developer.android.com/studio/run/managing-avds>. [Último acceso: 06 Abril 2022].
- [35] Backmarket, «MacBook Pro 13" (2017) - QWERTY - Español,» Apple, [En línea]. Available: [https://www.backmarket.es/es-es/p/macbook-pro-13-retina-2017-core-i5-23-ghz-ssd-120-gb-8gb-teclado-espanol/bfb5093c-5e15-40c2-ab90-9ab998b2e57f?shopping=gmc&gclid=CjwKCAjwk\\_WVBhBZEiwAUHQcmW60QDo5Xt9YSkQnD6KtAgbMD9XQpErVIcBsK7wIWp4gZfUuaITUthoCn84QAvD\\_BwE#](https://www.backmarket.es/es-es/p/macbook-pro-13-retina-2017-core-i5-23-ghz-ssd-120-gb-8gb-teclado-espanol/bfb5093c-5e15-40c2-ab90-9ab998b2e57f?shopping=gmc&gclid=CjwKCAjwk_WVBhBZEiwAUHQcmW60QDo5Xt9YSkQnD6KtAgbMD9XQpErVIcBsK7wIWp4gZfUuaITUthoCn84QAvD_BwE#). [Último acceso: 30 Junio 2022].
- [36] Samsung, «Galaxy A31,» Samsung, [En línea]. Available: <https://www.samsung.com/es/business/smartphones/galaxy-a/galaxy-a31-sm-a315gzkueub/>. [Último acceso: 30 Junio 2022].
- [37] Smart-gsm, «Google Pixel 5,» Google, [En línea]. Available: <https://www.smart-gsm.com/moviles/google-pixel-5>. [Último acceso: 30 Junio 2022].
- [38] J. J. R. Sala, «Temario de la asignatura "Introducción a la Ingeniería del Software",» [En línea]. Available:

[https://docs.google.com/presentation/d/1p\\_bf\\_y6xIzuHzJ6W\\_H38PppKrNpNqEOR07OzHdJto1M/edit#slide=id.i0](https://docs.google.com/presentation/d/1p_bf_y6xIzuHzJ6W_H38PppKrNpNqEOR07OzHdJto1M/edit#slide=id.i0). [Último acceso: 30 Junio 2022].

- [39] Fortinet, «What Is an API Key?,» [En línea]. Available: <https://www.fortinet.com/resources/cyberglossary/api-key>. [Último acceso: 27 Abril 2022].
- [40] M. GRANT, «Gantt Chart,» Gantt, 26 Agosto 2021. [En línea]. Available: <https://www.investopedia.com/terms/g/gantt-chart.asp>. [Último acceso: 24 Abril 2022].
- [41] Canva, «Canva,» Canva, [En línea]. Available: [https://www.canva.com/es\\_es/](https://www.canva.com/es_es/). [Último acceso: 30 Junio 2022].
- [42] IBM, «Definición de casos de uso,» [En línea]. Available: <https://www.ibm.com/docs/es/elm/6.0.3?topic=requirements-defining-use-cases>. [Último acceso: 18 Abril 2022].
- [43] Experience ux, «What is wireframing?,» [En línea]. Available: <https://www.experienceux.co.uk/faqs/what-is-wireframing/>. [Último acceso: 10 Abril 2022].
- [44] Google, «Firebase,» [En línea]. Available: <https://firebase.google.com/>. [Último acceso: 6 Mayo 2022].