***GestY***

***Por: Javier Garcia Ruiz***

[1. ÁNALISIS DEL PROBLEMA 3](#_Toc134729310)

[1.1 INTRODUCCION 3](#_Toc134729311)

[1.2 OBJETIVOS 3](#_Toc134729312)

[1.3 FUNCIONES Y RENDIMIENTOS DESEADOS 3](#_Toc134729313)

[1.4 PLANTEAMIENTO Y EVALUACION DE DIVERSAS SOLUCIONES 4](#_Toc134729314)

[1.5 JUSTIFICACION DE LA SOLUCION ELEGIDA 4](#_Toc134729315)

[1.6 MODEELADO DE LA SOLUCION 4](#_Toc134729316)

[1.7 PLANIFICACION TEMPORAL 5](#_Toc134729317)

[2. DISEÑO E IMPLEMENTACION DEL PROYECTO 5](#_Toc134729318)

[3. FASE DE PRUEBAS 6](#_Toc134729319)

[4. DOCUMENTACION DE LA APLICACIÓN 6](#_Toc134729320)

[4.1 INTRODUCCIÓN A LA APLICACIÓN 6](#_Toc134729321)

[4.2 MANUAL DE INSTALACIÓN 6](#_Toc134729322)

[4.3 MANUAL DE USUARIO 6](#_Toc134729323)

[4.4 MANUAL DE ADMINSTRACIÓN 6](#_Toc134729324)

[5. CONCLUSIONES FINALES 6](#_Toc134729325)

[6. BIBLIOGRAFIA 6](#_Toc134729326)

# ÁNALISIS DEL PROBLEMA

## INTRODUCCION

***GestY*** es una aplicación de gestión de empresas, donde podrás gestionar todo aquello necesario para organización de los quehaceres de un empleado dentro de su empresa. En *GestY* podrás gestionar: *Clientes, Proyectos, Listas de Tareas, Tareas y Empleados*, así como un apartado de mensajería instantánea entre empleados de la misma empresa, integrado dentro de la misma aplicación. También cuenta con un panel administrativo situado en la página oficial de *GestY.*

## OBJETIVOS

Con *GestY* buscamos la integración de todas las herramientas de gestión necesarias con el fin de facilitar el trabajo y la gestión de los empleados de las empresas asociadas a *GestY*. La integración de la mensajería instantánea dentro de la aplicación aporta un valor añadido a la aplicación, gracias a la falta de necesidad de alternar de aplicaciones para la comunicación con los empleados de la empresa.

## FUNCIONES Y RENDIMIENTOS DESEADOS

*GestY* cuenta con dos utilidades principales:

* Servidor: Situado en <https://gesty.devf6.es>, *GestY* cuenta con un panel administrativo desarrollado en el Framework web *Laravel* usando *PHP*, en el cual podremos encontrar a la empresa a la que pertenecemos, asi como un listado de tareas, empleados y la configuración de correo electrónico. Situado en el servidor *cloud* de la empresa *HiveMQ,* <https://www.hivemq.com/mqtt-cloud-broker/>, está alojado el servicio ***Mosquitto***(MQTT), servicio encargado de la gestión de la mensajería instantánea, concretamente en la *URL:*

<e7fa393ea4af4647a2482dffccd1d654.s2.eu.hivemq.cloud>

* Cliente: *GestY* cuenta con una versión para Ordenadores, desarrollado en *Java* con una interfaz grafica en Java Swift, así como una aplicación para Móviles Android, desarrollado en el Framework *Flutter* usando el lenguaje de programación *Dart.* Los clientes se conectan con una *API REST* situada en el servidor web de *GestY*, concretamente <https://gesty.devf6.es/api>.

## PLANTEAMIENTO Y EVALUACION DE DIVERSAS SOLUCIONES

A la hora de plantear el desarrollo de la aplicación, se buscó la forma de ampliar el rango de posibles problemas a solucionar, haciendo asi una más potente aplicación/infraestructura la cual presentase una fácil implementación en las empresas asociadas y fuera *User-Friendly* a la hora del uso de cualquiera de sus integraciones.

Se buscó una forma de centralización de servicios y datos, para realizar una aplicación con posible uso en grupos con datos disponibles para los usuarios de la aplicación.

## JUSTIFICACION DE LA SOLUCION ELEGIDA

Se valoró la idea de hacer en Java/Android la aplicación movil, pero se descartó definitivamente al ser Flutter una tecnología más actual y, a la vez, más potente con un desarrollo menor.

Para la aplicación de escritorio, se valoró la idea de desarrollarla junto a Flutter, gracias a que dispone de desarrollo multiplataforma, pero acabó desechándose por ser Flutter un lenguaje orientado a móviles, aunque disponga de desarrollo multiplataforma.

La idea inicial de centralizar todos los servicios en un servidor externo usando la tecnología Docker, creando contenedores propios y realizando una clusterización con Kubernetes (K8s), ha sido desechada para Producción de la aplicación ya que podría generar ralentización y fallas de seguridad de carácter grave o críticas. Sin embargo, se ha mantenido una dockerización los servicios necesarios para desarrollo y realización de pruebas en un entorno local, juntando PHP, MySQL y MQTT.

## MODEELADO DE LA SOLUCION

##### 1.6.1 RECURSOS HUMANOS

##### 1.6.2 RECURSOS HARDWARE

##### 1.6.3 RECURSOS SOFTWARE

## PLANIFICACION TEMPORAL

# DISEÑO E IMPLEMENTACION DEL PROYECTO

***Introducción a GestY***

*GestY* está montado sobre una ***infraestructura técnica robusta y muy potente***, buscando la ***solidez*** del traspaso de datos y almacenamientos de los mismos, asi como una ***usabilidad lo más sencilla pero potente posible***.

La infraestructura consta de:

* ***Servidor central / WebService***:

El servidor central es donde está corriendo el servidor web junto a la base de datos, siendo este servidor la pieza maestra de la infraestructura. Está montado sobre el dominio ***devf6.es*** gracias a que la empresa ***Área F5***, donde estoy cursando la Formación en Centros de Trabajo me ha cedido su espacio de desarrollo para el alojamiento de los servicios principales.

Dentro de este servidor estan corriendo los siguientes servicios:

* + ***Apache2 + PHP*** como Servidor Web:
  + ***MariaDB*** (v10.3.38) como Base de datos:

El servicio web está ejecutando una ***plataforma desarrollada en Laravel***, ya que Laravel tiene un funcionamiento muy robusto y potente a la vez, asi como una gran diversidad de librerías y utilidades que lo hacen un gran gestor de contenidos web (CMS). ***Laravel trabaja excepcionalmente con APIs***, tanto de cliente como de servidor.

El ***uso principal de Laravel*** en este proyecto ***es hacer de intermediario entre la aplicación cliente y la base de datos***; obteniendo, ordenando y preparando la información necesaria para la aplicación cliente.

Una de las principales ventajas de usar Laravel es su ***sistema de migraciones***. Este permite crear, de forma sencilla, la base de datos.

Laravel ***también hace de gestor de correos***, permitiendo crear, preparar y enviar correos gracias a su integración con el servicio SMTP y la facilidad de configuración del mismo.

* ***Servidor cloud MQTT***:

***MQTT*** es un ***protocolo de mensajería*** basado en estándares, o un conjunto de reglas, la que se utiliza para comunicación de un equipo a otro. ***MQTT*** ***admite la mensajería entre dispositivos*** a la nube y la nube al dispositivo.

El servicio ***MQTT*** he decidido alojarlo en un ***servidor cloud*** por la ***facilidad de acceso y configuración.*** A su vez, es mejor para el proyecto ya que *MQTT* ***requiere de grandes recursos de almacenamiento*** y es recomendable mantenerlo en un entorno adecuado para un correcto funcionamiento.

* ***Cliente***:

La aplicación cliente consta de dos versiones:

* + PC:

La aplicación para Ordenadores está basada en Java, usando como interfaz gráfica Java Swing.

Esta aplicación será la usada por los empleados, la cual constará con un inicio de sesión, asi como con un registro de empresa en el caso de que no tengas ninguna asignada con anterioridad.

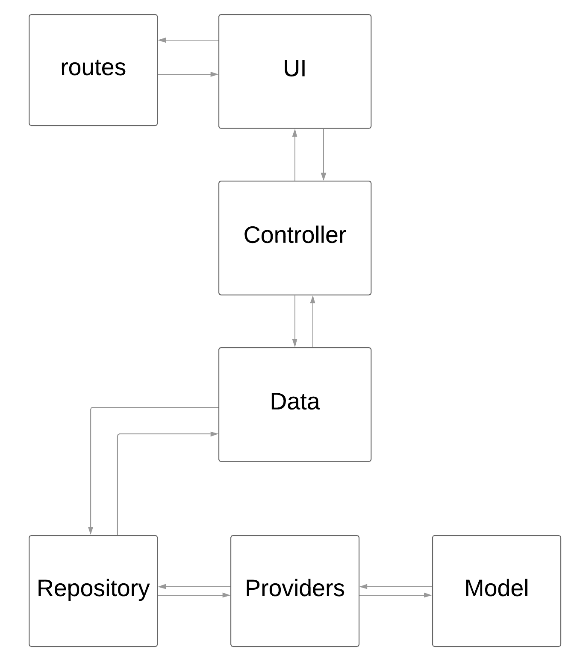
La funcionalidad de la aplicación cliente en Java consta de una serie de peticiones HTTP, usando la librería ***Apache HttpClient*** a una API REST montada en el [***WebService***](#_DISEÑO_E_IMPLEMENTACION) la cual hará de intermediario entre la base de datos y la aplicación, ocupándose de obtener, organizar y preparar la información, devolviéndola en formato JSON, la cual será formateada de vuelta, usando la librería ***Google Gson***.

* + Android:

La aplicación para móviles Android está basada en Dart, usando su propio Framework, Flutter, ambos desarrollados por ***Google***.

Esta aplicación está orientada a la administración de la misma, asi como para los jefes de la empresa. En ella podremos encontrar el listado de empresas, proyectos y tareas, asi como el listado de empleados que trabajan dentro de la misma.

La funcionalidad es similar a la aplicación Java, usando peticiones HTTP, usando la librería ***HTTP*** integrada en Flutter, a la API REST de nuestro [***WebService***](#_DISEÑO_E_IMPLEMENTACION) a través del cual obtendremos un listado de información en formato JSON, el cual formatearemos dentro de la aplicación usando el flujo de trabajo de Flutter.



Usando la librería ***GetX*** de Dart dispondremos de una serie de ***Providers, Consumers y Controllers***, los cuales nos facilitaran el desarrollo de la aplicación, asi como la obtención y utilización de la información obtenida a través de las peticiones a nuestro WebService.

# FASE DE PRUEBAS

# DOCUMENTACION DE LA APLICACIÓN

## INTRODUCCIÓN A LA APLICACIÓN

## MANUAL DE INSTALACIÓN

## 4.3 MANUAL DE USUARIO

## 4.4 MANUAL DE ADMINSTRACIÓN

# CONCLUSIONES FINALES

# BIBLIOGRAFIA

1. <https://trello.com/b/L432pNQ3/gesty> -------> Pizarra digital de quehaceres del proyecto
2. <https://laravel.com/docs/9.x/>
3. <https://pub.dev/packages/mqtt_client>
4. <https://www.emqx.com/en/blog/how-to-use-mqtt-in-java>
5. <https://community.jitsi.org/t/integrate-jitsi-with-a-java-application/20351>
6. <https://pub.dev/packages/jitsi_meet>
7. <https://hub.docker.com/_/eclipse-mosquitto>
8. <https://www.php.net/manual/es/>
9. <https://stackoverflow.com/>
10. <https://chat.openai.com/>
11. <https://pub.dev/packages/get>
12. <https://console.hivemq.cloud/clients/java-hivemq?uuid=e7fa393ea4af4647a2482dffccd1d654>
13. <https://aws.amazon.com/es/what-is/mqtt/>
14. <https://es.wikipedia.org/wiki/Flutter_(software)>
15. <https://tailwindcss.com/docs/installation>
16. <https://www.udemy.com/course-dashboard-redirect/?course_id=2311106>
17. <https://www.udemy.com/course-dashboard-redirect/?course_id=2306140>
18. <https://www.udemy.com/course-dashboard-redirect/?course_id=1813098>
19. <https://dart.dev/>
20. <https://hc.apache.org/httpcomponents-client-5.2.x/>
21. <https://laracasts.com/>
22. <https://medium.com/>