



Diseño y desarrollo de una plataforma de gestión de recursos humanos.

Isaac Morales General
Grado de Ingeniería Informática

Consultor: Gregorio Robles Martínez

09/06/2021



Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada [3.0 España de Creative Commons](https://creativecommons.org/licenses/by-nc-nd/3.0/es/)

FICHA DEL TRABAJO FINAL

Título del trabajo:	Diseño y desarrollo de una plataforma de gestión de recursos humanos
Nombre del autor:	Isaac Morales General
Nombre del consultor:	Gregorio Robles Martínez
Fecha de entrega (mm/aaaa):	06/2021
Área del Trabajo Final:	Desarrollo web
Titulación:	<i>Grado de Ingeniería Informática</i>
Resumen del Trabajo (máximo 250 palabras):	
<p>En 2019, entró en vigor la normativa de control horario, según la cual, las empresas deben registrar el inicio y fin de la jornada laboral, además de conservar estos datos durante, al menos, cuatro años. La infracción de esta normativa puede acarrear sanciones graves.</p> <p>Además, debido a la relación que tiene el control de la jornada horaria con otras tareas asociadas a los recursos humanos de cualquier empresa como son la solicitud y registro de vacaciones, ausencias médicas y consulta de nóminas se plantea este TFG, el cual tiene como objetivo desarrollar una aplicación web que permita a las empresas realizar este cometido.</p> <p>La aplicación web se ha desarrollado siguiendo una arquitectura de microservicios, intentando modularizar la aplicación para poder otorgar mayor escalabilidad e independencia. Además, se trata de una web SPA, es decir, recarga y muestra su contenido en respuesta a acciones propias de la navegación sin tener que realizar nuevas peticiones al servidor.</p>	

Abstract (in English, 250 words or less):

In 2019, the time control regulations came into force, according to which companies must record the start and end of the working day, in addition to keeping this data for at least four years. Violation of this regulation can lead to serious penalties.

In addition, due to the relationship that control of the working day has with other tasks associated with the human resources of any company such as the request and registration of vacations, medical absences and payroll consultation, this TFG is proposed, which aims develop a web application that allows companies to carry out this task.

The web application has been developed following a microservices architecture, trying to modularize the application in order to grant greater scalability and independence. In addition, it is a SPA (Single Page Application) website, which means that it recharges and displays its content in response to browsing actions without having to make new requests to the server.

Palabras clave (entre 4 y 8):

Registro horario, desarrollo web, recursos humanos, jornada diaria.

Índice

1. Introducción.....	1
1.1 Contexto y justificación del Trabajo.....	1
1.2 Objetivos del Trabajo.....	2
1.3 Enfoque y método seguido.....	3
1.4 Planificación del Trabajo.....	5
1.5 Breve resumen de productos obtenidos.....	7
1.6 Breve descripción de los otros capítulos de la memoria.....	7
2. Especificación de requisitos.....	9
2.1 Tabla de resumen de requisitos.....	9
2.2 Descripción de requisitos funcionales.....	10
2.3 Descripción de requisitos no funcionales.....	11
3. Análisis funcional.....	13
3.1 Actores del sistema.....	13
3.2 Tabla de resumen de casos de uso.....	14
3.3 Descripción de casos de uso.....	14
3.3 Diagrama UML de casos de uso.....	18
4. Diseño del producto.....	19
4.1 Arquitectura general de la aplicación.....	19
4.2 Seguridad de la aplicación.....	20
4.2 Back-end de la aplicación.....	21
4.3 Base de datos.....	24
4.4 Front-end de la aplicación.....	28
4.5 Herramientas de apoyo.....	29
5. Validación del producto.....	30
5.1 Fases en el desarrollo de los test funcionales.....	30
5.2 Test de aceptación.....	31
5.3 Batería de pruebas.....	32
6. Contenidos.....	38
6.1 Login.....	38
6.2 Registro de administradores y empresas.....	38
6.3 Home.....	41
6.4 Clock In.....	42
6.5 Absences.....	42
6.6 Pay Slips.....	43
6.7 Messages.....	45
6.8 Employees.....	45
6.9 Search.....	48
7. Conclusiones.....	49
7.1 Lecciones aprendidas.....	49
7.2 Consecución de objetivos planteados.....	49
7.3 Seguimiento de la planificación.....	49
7.4 Líneas futuras de trabajo.....	50
8. Glosario.....	51
9. Bibliografía.....	52
10. Anexo.....	55
10.1 Manual de instalación.....	55

Lista de figuras

Ilustración 1. Relación etapas TFG vs modelo en cascada.	4
Ilustración 2. Diagrama de Gantt con la estimación de tiempos para la realización del TFG.	6
Ilustración 3. Definición de casos de uso.	18
Ilustración 4. Arquitectura general de la aplicación.	19
Ilustración 5. Ciclo de vida de un token JWT.	21
Ilustración 6. Arquitectura de microservicios.	23
Ilustración 7. Modelo de base de datos.	24
Ilustración 8. Pantalla login.	38
Ilustración 9. Pantalla registro administrador.	39
Ilustración 10. Pantalla registro administrador y empresa.	40
Ilustración 11. Pantalla home.	41
Ilustración 12. Pantalla clockin.	42
Ilustración 13. Pantalla absences.	43
Ilustración 14. Pantalla payslips.	44
Ilustración 15. Pantalla nómina.	44
Ilustración 16. Pantalla messages.	45
Ilustración 17. Pantalla employees.	46
Ilustración 18. Pantalla de empleado.	47
Ilustración 19. Pantalla search.	48

1. Introducción

1.1 Contexto y justificación del Trabajo

Debido a la normativa vigente de control horario dispuesta en el Real Decreto-Ley 8/2019 de 8 de marzo [1], las empresas se han visto obligadas a implantar diferentes tecnologías para poder registrar la jornada diaria de sus trabajadores. Concretamente, el control horario entró en vigor el 12 de mayo de 2019, dando, de esta forma, un margen de tiempo de dos meses a las empresas para adaptarse y cumplir con la obligación del control horario.

La normativa impone a las empresas registrar tanto el inicio como la finalización de la jornada laboral y, además, estas deben conservar estos registros por un periodo de tiempo de cuatro años y tenerlos a disposición de los trabajadores, sindicatos e Inspección de Trabajo y Seguridad Social. El incumplimiento de esta normativa constituye una infracción grave y la multa asociada a la misma tendrá una cuantía que estará entre los 626 y los 6250 euros. Por lo tanto, el desarrollo de una aplicación web que lleve a cabo esta gestión unida a la de vacaciones y nóminas otorga un gran valor a los departamentos de recursos humanos de las empresas.

Este TFG consistirá en el desarrollo de una aplicación web que permita la gestión llevada a cabo por un departamento de recursos humanos de cualquier empresa. Esta aplicación permitirá la gestión de vacaciones, nóminas y control horario. Además, no se limitará a una empresa concreta, sino que un administrador/gestor podrá dar de alta una nueva empresa y llevar el control de sus empleados desde ella.

1.2 Objetivos del Trabajo

Se establecen los siguientes objetivos generales:

- I. Diseñar y desarrollar una parte back-end que permita realizar mediante una arquitectura de microservicios las distintas funciones mencionadas.
- II. Diseñar y desarrollar una parte front-end que permita la interacción de los usuarios con la aplicación.
- III. Implementar una base de datos que almacena toda la información necesaria para llevar a cabo el desarrollo de la aplicación.

Se establecen los siguientes objetivos específicos:

Objetivo 1: Permitir el registro de usuarios y administradores.

- i. Permitir el registro de un administrador que a su vez se le asigna a una empresa existente o nueva.
- ii. Permitir el registro de un empleado al que se le asigna a una empresa existente.
- iii. Implementar una página web de registro de empleados y administradores.
- iv. Implementar una página web de login de usuarios y administradores.
- v. Diseñar la parte correspondiente en la base de datos para el almacenamiento de usuarios y administradores.

Objetivo 2: Gestión de control horario.

- i. Permitir a los empleados indicar cuando comienza y finalizan una jornada (se podrán marcar varios inicios y finales a lo largo del día).
- ii. Implementar la vista de administrador para comprobar las horas trabajadas de los usuarios.
- iii. Diseñar la parte correspondiente en la base de datos para el almacenamiento de las jornadas diarias.

Objetivo 3: Gestión de las vacaciones.

- i. Permitir a los empleados solicitar vacaciones indicando cuándo comenzarían y finalizarían.
- ii. Implementar la vista de administrador para aprobar o denegar las vacaciones de los empleados.
- iii. Diseñar la parte correspondiente en la base de datos para el almacenamiento de las vacaciones.

Objetivo 4: Gestión de las ausencias.

- i. Permitir a los empleados introducir ausencias justificadas indicando cuándo comenzarían y finalizarían.
- ii. Implementar la vista de administrador para comprobar las ausencias de los empleados.
- iii. Diseñar la parte correspondiente en la base de datos para el almacenamiento de las ausencias.

Objetivo 5: Gestión de nóminas.

- i. Permitir a los empleados consultar y descargar en PDF sus nóminas.

1.3 Enfoque y método seguido

Debido a que el desarrollo del proyecto se ha llevado a cabo de manera secuencial, con la entrega de diferentes PECs, la metodología llevada a cabo ha sido la metodología waterfall, también conocida como modelo de desarrollo en cascada [2]. La primera mención a realizar un modelo en fases se debe a Winson Royce [3], sin embargo, su uso comienza a extenderse a partir del artículo de Bell y Thayer [4]. Esta metodología implica ordenar las distintas etapas del desarrollo, es decir, una nueva etapa no comenzará hasta que finalice completamente la anterior. Una etapa se considera finalizada cuando se ha obtenido el entregable correspondiente a ella. La secuencia de etapas que sigue este modelo puede reducirse a la siguiente:

1. Análisis de requisitos
2. Diseño del software
3. Implementación
4. Verificación de errores
5. Despliegue del producto
6. Mantenimiento

Entre las principales ventajas de esta metodología se encuentran:

- Es un modelo útil para llevar un orden y organizar el trabajo.
- Es sencillo y fácil de seguir.
- Se detecta fácilmente la etapa en la que surge el problema para poder solucionarlo rápidamente.
- Es más fácil de medir el progreso.

A su vez, se encuentran las siguientes desventajas:

- Poco margen para realizar ajustes a lo largo del proyecto debido cambios en los requisitos.
- En ocasiones, los fallos solo se detectan una vez finalizado el proceso de implementación.

Etapa	Entregable
Análisis	PEC 1
Diseño	PEC 2
Implementación y verificación	PEC 3
Despliegue y mantenimiento	PEC 4

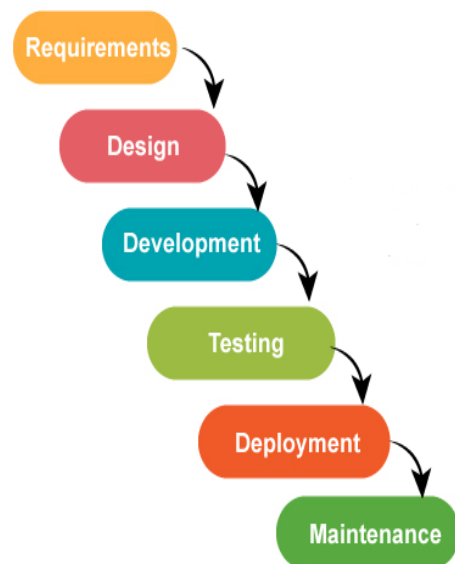


Ilustración 1. Relación etapas TFG vs modelo en cascada.

1.4 Planificación del Trabajo

Para la planificación del trabajo y teniendo en cuenta los objetivos descritos en el apartado 1.2 se enumeran las siguientes tareas a realizar:

1. Diseñar e implementar la base de datos.
2. Diseñar e implementar el microservicio encargado de la gestión de usuarios.
3. Diseñar e implementar el microservicio encargado de la gestión del control horario.
4. Diseñar e implementar el microservicio encargado de la gestión de las vacaciones y ausencias.
5. Diseñar e implementar el microservicio encargado de la gestión de las nóminas
6. Diseñar e implementar el componente front de registro de usuarios.
7. Diseñar e implementar el componente front de login.
8. Diseñar e implementar el componente front de gestión del control horario.
9. Diseñar e implementar el componente front de gestión de las vacaciones.
10. Diseñar e implementar el componente front de gestión de las ausencias.
11. Diseñar e implementar el componente front de gestión de las nóminas.
12. Realizar test unitarios y funcionales para comprobar el correcto funcionamiento de la aplicación.
13. Terminar memoria del TFG y preparar la presentación.

Se implementa un esquema que represente las tareas pendientes de realizar en función del tiempo en el que se estima que se completarán.

ACTIVITY	February	March	April	May	June
PEC 1	4 Days				
PEC 2		30 Days			
Análisis y diseño de la base de datos		8 Days			
Implementación de la base de datos		4 Days			
Análisis y diseño de la arquitectura de microservicios		8 Days			
Implementación microservicio Eureka		2 Days			
Implementación microservicio Gateway		2 Days			
Implementación microservicio Spring Cloud Config		2 Days			
Implementación microservicio registro horario		4 Days			
PEC 3			30 Days	20 Days	
Implementación microservicio registro ausencias			4 Days		
Implementación microservicio usuarios y administradores			4 Days		
Implementación microservicio gestión empresas			4 Days		
Implementación microservicio gestión nóminas			4 Days		
Realizar test unitarios y funcionales			6 Days		
Análisis y diseño parte front-end			8 days	2 Days	
Implementación parte front-end				15 Days	
Despliegue de aplicación en entorno				3 Days	
PEC 4				8 Days	8 Days
Realizar memoria				8 Days	
Realizar presentación					8 Days

Ilustración 2. Diagrama de Gantt con la estimación de tiempos para la realización del TFG.

1.5 Breve resumen de productos obtenidos

- Código fuente de los distintos microservicios.
- Código fuente de la librería con clases comunes a los microservicios.
- Código fuente de la aplicación web.
- Script de la base de datos.
- Script con datos falseados para poblar la base de datos.
- Documentación de referencia para el desarrollo de la aplicación.
 - Especificación de requisitos.
 - Diseño de la arquitectura.
- Memoria del Trabajo de Fin de Grado.

1.6 Breve descripción de los otros capítulos de la memoria

- Capítulo 2: Especificación de requisitos. Se describe el comportamiento del sistema desarrollado.
- Capítulo 3: Análisis funcional. Se especifica de manera concreta y detallada qué tiene que hacer la aplicación y cómo tiene que hacerlo.
- Capítulo 4: Diseño del producto. Se define cómo va a ser el software. Es decir, se describe cada uno de los componentes que formarán parte de la aplicación.
- Capítulo 5: Validación del producto. Se muestran las distintas pruebas que se realizan para comprobar que el sistema cumple con las funciones designadas.
- Capítulo 6: Contenidos. Se exponen las distintas páginas de la aplicación web.
- Capítulo 7: Conclusiones. Se argumentan las lecciones aprendidas y se muestran las posibles líneas de trabajo futuras.

- Capítulo 8: Glosario. Catálogo de palabras y acrónimos junto con su definición.
- Capítulo 9: Bibliografía. Referencias a las publicaciones consultadas.
- Capítulo 10. Anexos. Manual de descarga e instalación.

2. Especificación de requisitos

En este apartado se describe cada uno de los requisitos que la aplicación debe cumplir [5]. Los requisitos se van a dividir en funcionales (RF) y no funcionales (RNF):

- Requisitos funcionales (RF): Aquellos que describen el comportamiento del sistema en función de una acción desarrollada por el usuario.
- Requisitos no funcionales (RNF): Aquellos que se enfocan en cuestiones de diseño e implementación.

2.1 Tabla de resumen de requisitos

REQUISITOS FUNCIONALES	
RF.01	Registro de nuevos administradores
RF.02	Registro de nuevos usuarios
RF.03	Registro de nuevas empresas
RF.04	Autenticación de usuarios
RF.05	Solicitud de vacaciones
RF.06	Registro de la jornada laboral
RF.07	Aceptación/denegación de vacaciones por parte del administrador
RF.08	Consulta de nóminas mensuales
RF.09	Consulta de jornada registrada de un usuario
RF.10	Registro de ausencias

REQUISITOS NO FUNCIONALES	
RNF.01	Frontal
RNF.02	Base de datos
RNF.03	Arquitectura de microservicios
RNF.04	Disponibilidad y carga del sistema
RNF.05	Manejo y seguridad de credenciales

2.2 Descripción de requisitos funcionales

Nombre	Registro de nuevos administradores
Código	RF.01
Tipo	Conducta
Descripción	El sistema debe permitir el registro de administradores y asociarlos a una empresa existente o nueva en ese momento.

Nombre	Registro de nuevos usuarios
Código	RF.02
Tipo	Conducta
Descripción	El sistema debe permitir el registro de usuarios y asociarlos a una empresa existente.

Nombre	Registro de nuevas empresas
Código	RF.03
Tipo	Conducta
Descripción	El sistema debe permitir el registro de nuevas empresas en el momento en el que se registra un administrador.

Nombre	Autenticación de usuarios
Código	RF.04
Tipo	Seguridad
Descripción	El sistema debe autenticar a los usuarios mediante credenciales (usuario y contraseña) para proteger los datos de los usuarios y limitar el acceso a los recursos disponibles.

Nombre	Solicitud de vacaciones
Código	RF.05
Tipo	Conducta
Descripción	El sistema debe permitir la solicitud de vacaciones por parte de los usuarios.

Nombre	Registro de la jornada laboral
Código	RF.06
Tipo	Conducta
Descripción	El sistema debe permitir el registro horario por parte de los usuarios.

Nombre	Aceptación/denegación de vacaciones por parte del administrador
Código	RF.07
Tipo	Conducta
Descripción	El sistema debe permitir la aceptación/denegación de las vacaciones de los usuarios por parte de un administrador de la empresa correspondiente.

Nombre	Consulta de nóminas mensuales
Código	RF.08
Tipo	Conducta
Descripción	El sistema debe permitir que el usuario consulte sus nóminas.

Nombre	Consulta de jornada registrada de un usuario
Código	RF.09
Tipo	Conducta
Descripción	El sistema debe permitir que el administrador consulte el tiempo trabajado por parte de un usuario concreto que pertenezca a su empresa.

Nombre	Registro de ausencias
Código	RF.10
Tipo	Conducta
Descripción	El sistema debe permitir el registro de ausencias justificadas por parte del usuario.

2.3 Descripción de requisitos no funcionales

Nombre	Frontal
Código	RNF.01
Tipo	Organizacional
Descripción	El sistema debe contar con un frontend desarrollado en Angular. Esta aplicación será la encargada de interactuar con el backend del sistema y con los usuarios.

Nombre	Base de datos
Código	RNF.02
Tipo	Organizacional
Descripción	El sistema debe contar con una base de datos PostgreSQL que almacene la información necesaria.

Nombre	Arquitectura de microservicios
Código	RNF.03
Tipo	Organizacional
Descripción	El sistema debe contar con un backend desarrollado en Java y el framework SpringBoot cuya misión será, a través de una serie de microservicios, dotar al frontend de la información que este solicite. Además, se encargará de recoger y almacenar la información necesaria en la base de datos

Nombre	Disponibilidad y carga del sistema
Código	RNF.04
Tipo	Negocio
Descripción	El sistema debe poder funcionar 24x7 sin interrupciones y con una carga de uso adaptada a la demanda.

Nombre	Manejo y seguridad de credenciales
Código	RNF.05
Tipo	Seguridad
Descripción	Las contraseñas de los usuarios se almacenarán encriptadas en la base de datos.

3. Análisis funcional

En este apartado se definen los actores del sistema y los casos de uso que describirán las actividades que deben llevar a cabo los actores para poder ejercer las acciones requeridas en los requisitos funcionales.

La definición de casos de usos es una técnica utilizada para capturar los requerimientos funcionales de un sistema [6]. Describen las interacciones entre los actores del sistema y el sistema en sí mismo, dotando una narrativa de cómo el sistema es utilizado.

3.1 Actores del sistema

Actor	Usuario no registrado
Código	ACT.01
Descripción	Usuario que no está registrado en el sistema.

Actor	Usuario no autenticado
Código	ACT.02
Descripción	Usuario que no se ha autenticado mediante credenciales de acceso.

Actor	Administrador autenticado
Código	ACT.03
Descripción	Usuario que se ha autenticado mediante credenciales de acceso tiene rol de administrador.

Actor	Empleado autenticado
Código	ACT.04
Descripción	Usuario que se ha autenticado mediante credenciales de acceso tiene rol de empleado.

3.2 Tabla de resumen de casos de uso

CASOS DE USO		
Código	Descripción	Requisito asociado
CU.01	Registro de nuevos administradores en empresas existentes	RF.01
CU.02	Registro de nuevos administradores en empresas no existente	RF.01, RF.03
CU.03	Registro de nuevos empleados	RF.02
CU.04	Login erróneo	RF.04
CU.05	Login satisfactorio	RF.04
CU.06	Solicitud de vacaciones	RF.05
CU.07	Aceptación/denegación de vacaciones por parte del administrador	RF.07
CU.08	Consulta de nóminas mensuales	RF.08
CU.09	Consulta de número de horas trabajadas de un empleado por parte del administrador	RF.09
CU.10	Registro de ausencias por parte de empleado	RF.10
CU.11	Registro de jornada diaria por parte de empleado	RF.06

3.3 Descripción de casos de uso

Nombre	Registro de nuevos administradores en empresas existentes
Actor	Usuario no registrado
Código	CU.01
Requisito	RF.01
Descripción	Se registrará un nuevo usuario con rol de administrador como parte de una empresa que ya está registrada en el sistema.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Pulsa sobre el enlace "Don't you have an account?" 3. Introduce todos los datos del formulario y selecciona una empresa del desplegable. 4. Pulsa sobre el botón "Create"

Nombre	Registro de nuevos administradores en empresas no existentes
Actor	Usuario no registrado
Código	CU.02
Requisito	RF.01, RF.03
Descripción	Se registrará un nuevo usuario con rol de administrador como parte de una empresa que no está registrada en el sistema, es decir, también se creará la empresa a su vez.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Pulsa sobre el enlace "Don't you have an account?" 3. Introduce todos los datos del formulario y pulsa sobre el botón "Create Company" 4. Rellena los datos del formulario de New Company 5. Pulsa sobre el botón "Create"

Nombre	Registro de nuevos empleados
Actor	Administrador autenticado
Código	CU.03
Requisito	RF.02
Descripción	Se registrará un nuevo usuario con rol de empleado por parte del administrador.
Pasos realizados	<ol style="list-style-type: none"> 1. El administrador accede a la aplicación 2. Introduce sus credenciales y hace el login 3. Pulsa sobre la pestaña "Employees – Employees" 4. Completa el formulario para añadir un nuevo empleado 5. Pulsa sobre el botón "Add new employee"

Nombre	Login erróneo
Actor	Usuario no autenticado, Usuario no registrado
Código	CU.04
Requisito	RF.04
Descripción	Un usuario registrado o no intentará autenticarse con credenciales falsas.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Introduce unas credenciales erróneas 3. Pulsa sobre el botón "Enter" 4. Se obtiene un mensaje de error

Nombre	Login satisfactorio
Actor	Usuario no autenticado
Código	CU.05
Requisito	RF.04
Descripción	Un usuario registrado intentará autenticarse con credenciales auténticas.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Introduce sus credenciales 3. Pulsa sobre el botón "Enter" 4. Accede a la aplicación

Nombre	Solicitud de vacaciones
Actor	Empleado autenticado, Administrador autenticado
Código	CU.06
Requisito	RF.05
Descripción	Un usuario autenticado (empleado o administrador) realiza una solicitud de vacaciones.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Introduce sus credenciales y hace el login 3. Pulsa sobre la pestaña "Absences" 4. Selecciona el rango de fechas y en el desplegable de tipos de ausencias selecciona "Holidays" 5. Pulsa sobre el botón "Request"

Nombre	Aceptación/denegación de vacaciones por parte del administrador
Actor	Administrador autenticado
Código	CU.07
Requisito	RF.07
Descripción	Un administrador autenticado (empleado o administrador) acepta o deniega una solicitud de vacaciones.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Introduce sus credenciales y hace el login 3. Pulsa sobre la pestaña o sobre el botón "Messages" 4. Pulsa sobre el botón "Validate" para aceptarlas o sobre "Decline" para rechazarlas.

Nombre	Consulta de nóminas mensuales
Actor	Empleado autenticado, Administrador autenticado
Código	CU.08
Requisito	RF.08
Descripción	Un usuario autenticado (empleado o administrador) consulta la nómina de un mes concreto.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Introduce sus credenciales y hace el login 3. Pulsa sobre la pestaña "Pay Slips" 4. Selecciona el mes y año del cual quiere obtener la nómina 5. Pulsa sobre el botón "Request"

Nombre	Consulta de número de horas trabajadas de un empleado por parte del administrador
Actor	Administrador autenticado
Código	CU.09
Requisito	RF.09
Descripción	Un administrador autenticado consulta el número de horas trabajadas por un empleado durante el último mes.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Introduce sus credenciales y hace el login 3. Pulsa sobre la pestaña "Employees – Employees" 4. Pulsa sobre la ficha del usuario que quiere analizar

Nombre	Registro de ausencias por parte de empleado
Actor	Empleado autenticado, Administrador autenticado
Código	CU.10
Requisito	RF.10
Descripción	Un usuario autenticado (empleado o administrador) registra una ausencia justificada.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Introduce sus credenciales y hace el login 3. Pulsa sobre la pestaña "Absences" 4. Selecciona el rango de fechas y en el desplegable de tipos de ausencias selecciona "Medical Leave" o "Personal Matters" 5. Pulsa sobre el botón "Request"

Nombre	Registro de jornada diaria por parte de empleado
Actor	Empleado autenticado, Administrador autenticado
Código	CU.11
Requisito	RF.06
Descripción	Un usuario autenticado (empleado o administrador) registra su jornada diaria.
Pasos realizados	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación 2. Introduce sus credenciales y hace el login 3. Pulsa sobre la pestaña o sobre el botón "Clock In" 4. Pulsa sobre el botón "Clockin" para marcar la entrada y sobre "Clockout" para la salida

3.3 Diagrama UML de casos de uso

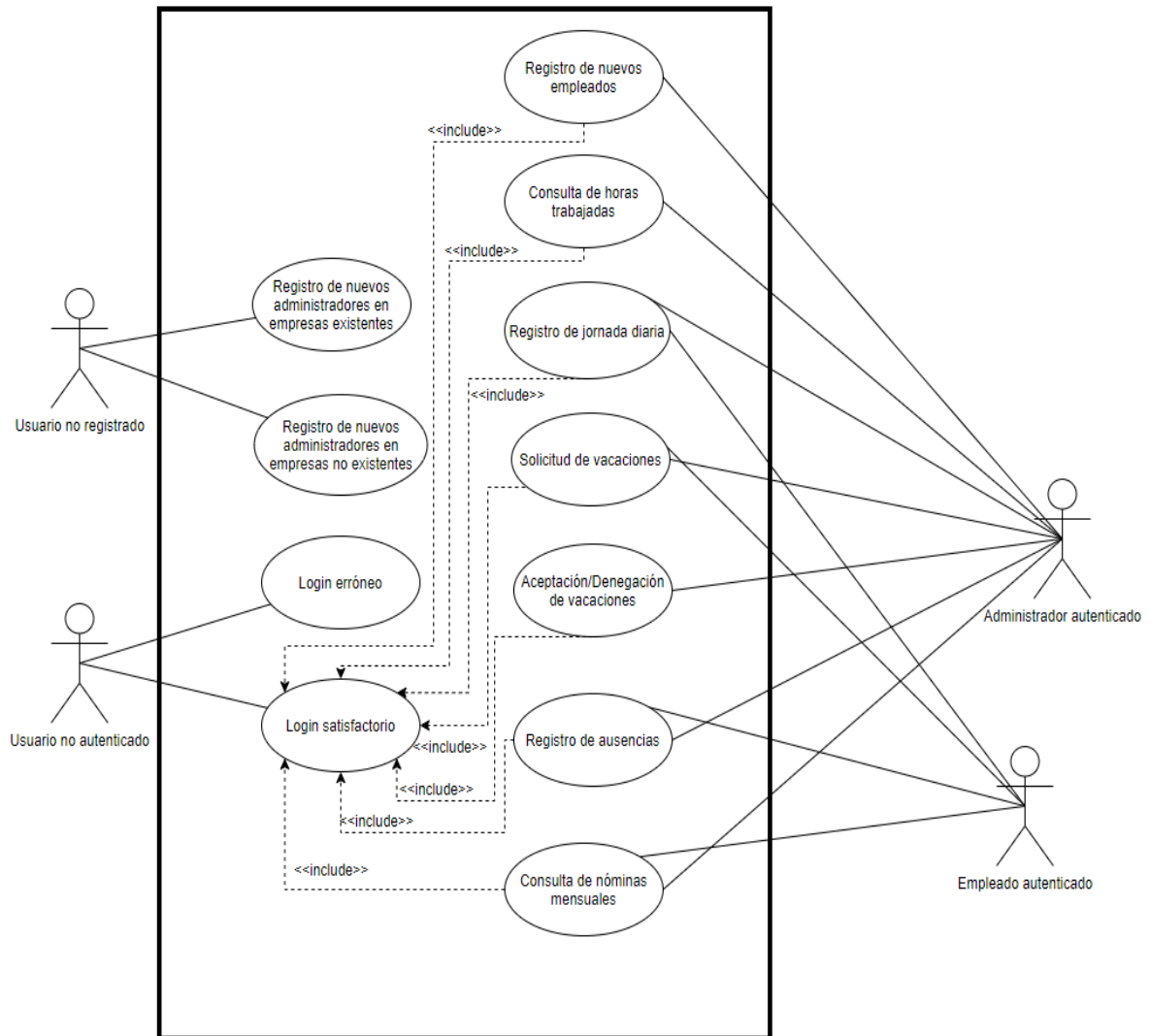


Ilustración 3. Definición de casos de uso.

4. Diseño del producto

A continuación, se describen los detalles del producto desarrollado para poder obtener los casos de uso descritos anteriormente.

4.1 Arquitectura general de la aplicación

La aplicación se divide en una parte back-end, la cual basa en una arquitectura de microservicios desarrollada con Spring Boot y PostgreSQL como sistema de base de datos relacional y para la parte front-end se utiliza HTML5 como lenguaje para el desarrollo y creación de las páginas web en conjunto con CSS3 para definir los estilos de esa páginas y Angular 11 como framework de desarrollo web que ofrece la base para el desarrollo de aplicaciones robustas, escalables y optimizadas.

Además de las tecnologías antes mencionadas, la aplicación va a utilizar Git para almacenar los ficheros de configuración de los que van a hacer uso los distintos microservicios a través del servidor de configuración.

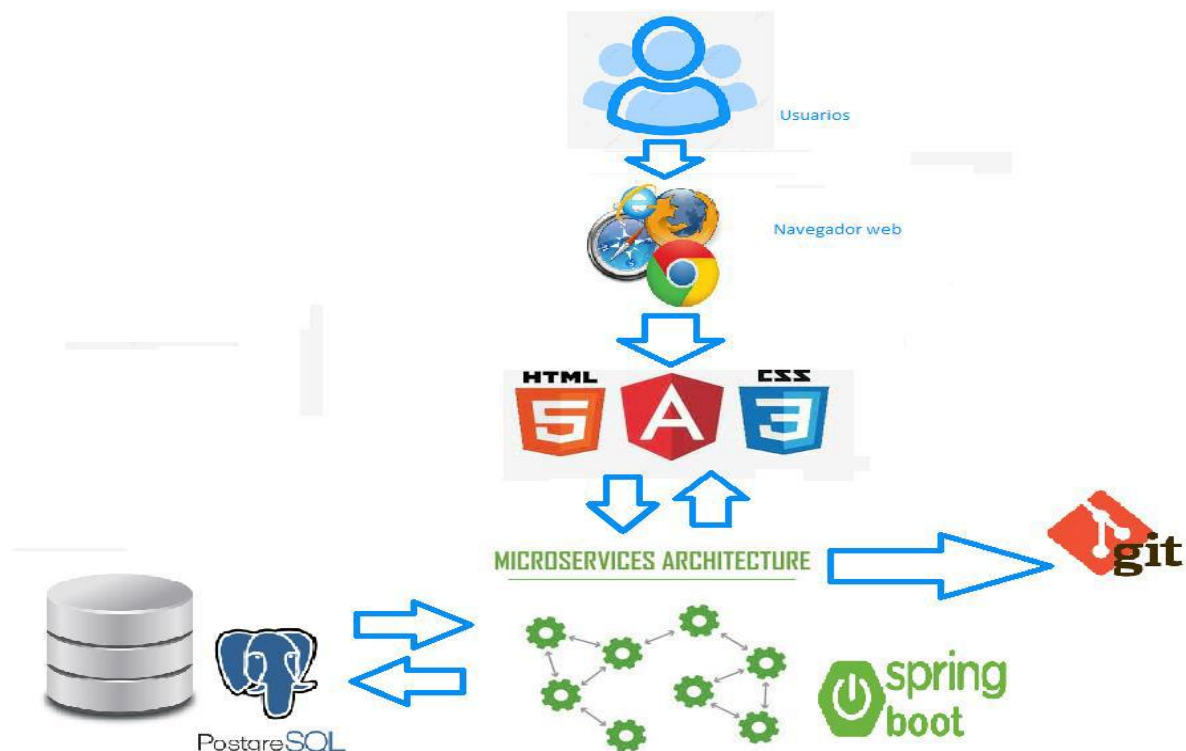


Ilustración 4. Arquitectura general de la aplicación.

4.2 Seguridad de la aplicación

La aplicación va a ser securizada utilizando Oauth 2.0 y JWT (JSON Web Token) para el acceso a los recursos de los microservicios. Oauth 2.0 es una framework de autorización que proporciona una guía en la que se definen flujos de autorización específicos para acceder a los datos del usuario desde diferentes tipos de aplicaciones consumidoras utilizando diferentes tipos de tokens, entre ellos el JWT [7]. Por otra parte, JWT es un estándar dentro del documento creado por parte del IETF llamado RFC 7519 [8] [9] en el cual se define un mecanismo para propagar entre dos partes la identidad de un determinado usuario y los roles o privilegios que posee. Estos privilegios se incrustan en objetos JSON dentro del cuerpo de un mensaje firmado digitalmente. El token consiste en una cadena de texto formada por tres partes codificadas en Base64:

- Header: encabezado dónde se indica el algoritmo y el tipo de token.
- Payload: donde aparecen los datos de usuario y privilegios, así como toda la información que queramos añadir.
- Signature: una firma que nos permite verificar si el token es válido.

La web oficial de JWT nos permite codificar y decodificar tokens [10].

Para encriptar las contraseñas de los usuarios almacenadas en la base de datos se utilizará Bcrypt [11]. Consiste en una función que genera un hash a partir de una cadena de caracteres. Un hash es un algoritmo matemático que transforma cualquier bloque arbitrario de datos en una nueva serie de caracteres con una longitud fija. Independientemente de la longitud de los datos de entrada, el valor hash de salida tendrá siempre la misma longitud.

La encriptación de las contraseñas del sistema se realizará utilizando Spring Boot Security, en concreto, la clase BcryptPasswordEncoder [12] [13].



Ilustración 5. Ciclo de vida de un token JWT. [<https://openwebinars.net/blog/que-es-json-web-token-y-como-funciona/>]

4.2 Back-end de la aplicación

El proyecto se ha desarrollado en Java 8 [14], utilizando el framework Spring [15], en concreto se ha optado por utilizar Spring Boot [16], ya que posee todas las características generales de Spring, añadiendo una mayor facilidad a la hora de realizar las configuraciones necesarias. También se he elegido Maven [17] como herramienta para la gestión y construcción del proyecto.

Spring es una de las soluciones más populares para el desarrollo de aplicaciones empresariales en Java. Entre las principales características podemos encontrar:

- Inyección de dependencias (DI): es un patrón de diseño, en el que se proporcionan objetos a una clase en lugar de ser la propia clase la que cree dichos objetos.
- Programación orientada a aspectos (AOP): es un paradigma de programación que permite una mejor separación de responsabilidades, de forma que se encapsulan diferentes conceptos que componen una aplicación en entidades bien definidas.

Maven es una herramienta de software para la gestión y construcción de proyectos Java creada por Jason van Zyl, de Sonatype, en 2002 [18].

Para este propósito utiliza un Project Object Model (POM) en el que se describe el proyecto de software que se va a construir, sus dependencias de otros módulos y componentes externos, y el orden de construcción de los elementos [19].

Para el desarrollo del proyecto se ha decidido seguir una arquitectura basada en microservicios, cuya idea principal es dividir el software en servicios que sean lo más independientes entre ellos que sea posible y, de esta forma, sean fácilmente reemplazables y actualizables [20] [21].

En concreto, la aplicación se compone de diversos microservicios (absences, companies, dailyentries, oauth y users). Además de estos microservicios, se ha implantado un servidor Eureka [22], un API Gateway con Zuul [23] y un servidor de configuración con Spring Cloud Config. También se ha desarrollado una librería que contenga las clases comunes a todos los microservicios para implementarlas una sola vez y poder utilizarlas desde todos los servicios con la importación de la librería.

Eureka es un servicio REST que se comporta como un servidor, su objetivo es registrar información sobre el resto de microservicios (nombre, IP, estado, ...) Además, facilita el balanceo de carga y la tolerancia a fallos. Todos los microservicios deben definirse como clientes de Eureka, de esta forma cuando arranquen se comunicarán con el servidor Eureka para notificarle que están disponibles para ser consumidos. El servidor Eureka mantiene la información de todos los microservicios registrados y su estado. Cada microservicio le notificará su estado cada 30 segundos.

Zuul actúa como proxy inverso, proporcionando un punto de acceso único (gateway) a todos los microservicios que forman parte del sistema.

Spring Cloud Config permite exteriorizar y centralizar la configuración de los microservicios en un único lugar, facilitando la administración de la configuración del sistema.

Además, se utiliza Git [24] como repositorio para centralizar la configuración de cada uno de los servicios.

La arquitectura de microservicios seguirá el siguiente esquema:

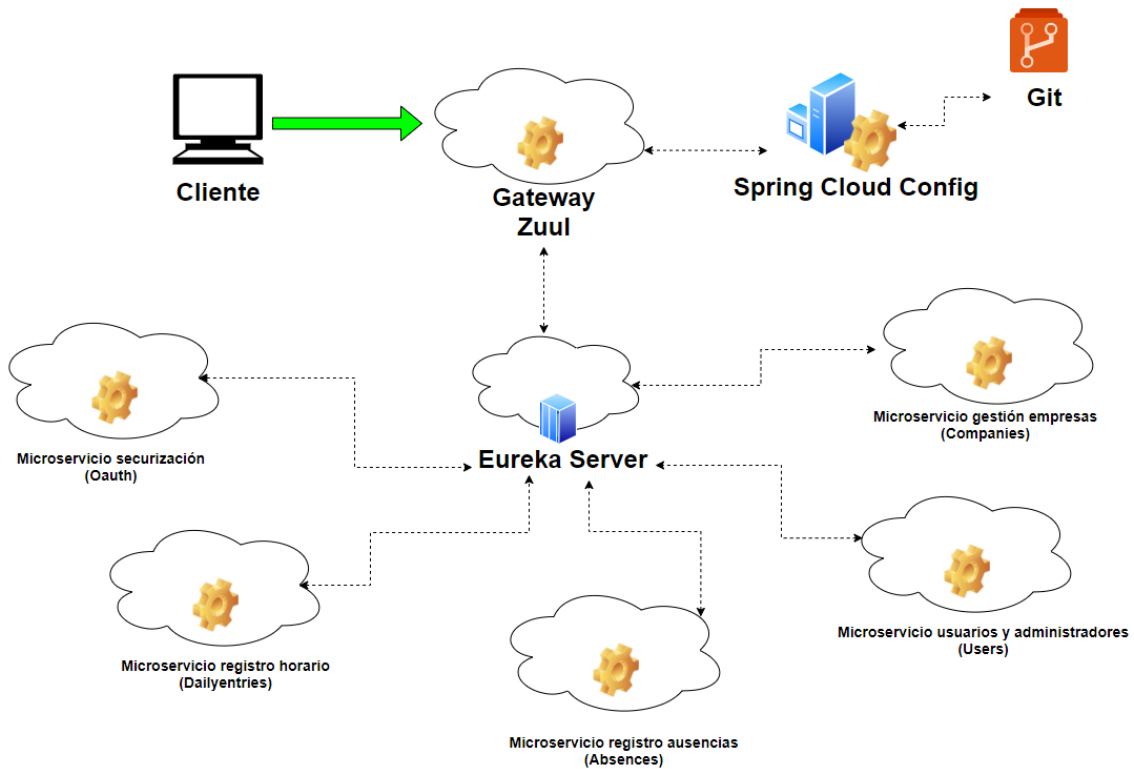


Ilustración 6. Arquitectura de microservicios.

A continuación, se proporciona una pequeña descripción de cada uno de los microservicios desarrollados:

- Servicio de gestión de usuarios y administradores (Users): permite registrar usuarios y administradores. También, permite obtener la información necesaria de los usuarios para generar las nóminas y obtener todos los usuarios que pertenecen a una compañía determinada.
- Servicio de gestión de empresas (Companies): permite registrar empresas.
- Servicio de registro horario (Dailyentries): permite registrar la jornada de los usuarios.
- Servicio de registro de ausencias (Absences): permite registrar las ausencias de los usuarios.
- Servicio de securización (Oauth): permite la autenticación de los usuarios y genera un JWT.

4.3 Base de datos

Se utiliza PostgreSQL como sistema gestor de la base de datos de la aplicación. PostgreSQL [25] [26] es un sistema para gestionar bases de datos de alto nivel, de software libre y con licencia BSD, compatible con cualquier uso, ya sea personal o comercial.

Teniendo en cuenta el diseño de la aplicación, se han generado las siguientes tablas:

- Employee: contiene la información de los usuarios de la aplicación.
- Company: contiene la información de las compañías registradas en la aplicación.
- Daily_entry: contiene la información de las jornadas registradas por los usuarios.
- Absence: contiene la información de las ausencias registradas por los usuarios.
- Role: contiene los distintos tipos de roles que pueden tener los usuarios de la aplicación.
- User_role: contiene la relación entre usuarios y roles.

En este punto se describe de forma esquemática y tabular la estructura de la base de datos del sistema:

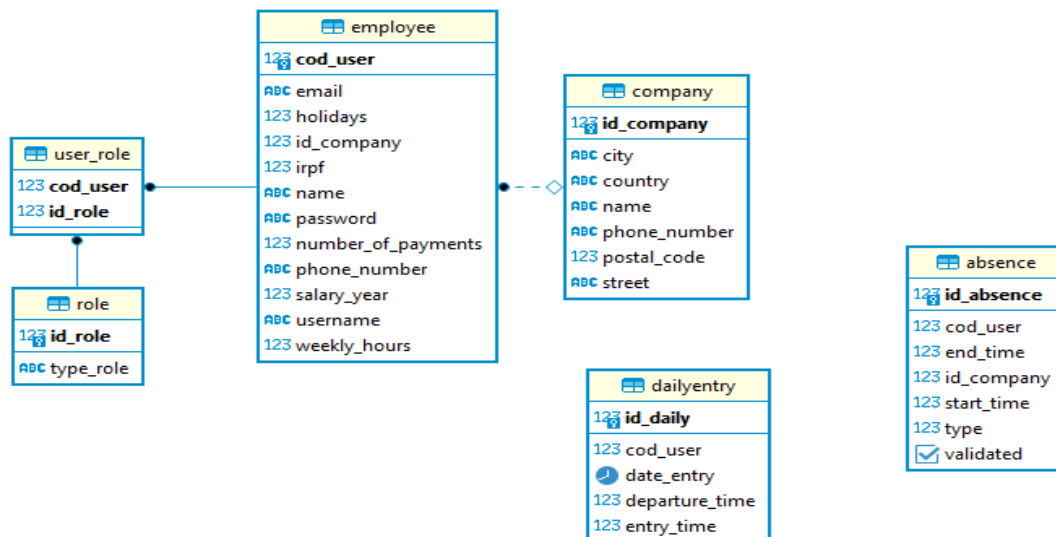


Ilustración 7. Modelo de base de datos.

Employee			
Descripción	Usuarios de la aplicación		
Atributo	Descripción	Tipo	Nullable
Cod_user	Clave primaria, columna incremental	Integer	No
Email	Correo electrónico	Varchar	No
Holidays	Número de días de vacaciones al año	Integer	No
Id_company	Identificador único de la compañía	Integer	No
Irfp	Irfp aplicable	Float	No
Name	Nombre del usuario	Varchar	No
Password	Password del usuario	Varchar	No
Number_of_payments	Número de pagas anuales	Float	No
Phone_number	Número de teléfono	Varchar	No
Salary_year	Salario anual	Float	No
Username	Alias del usuario	Varchar	No
Weekly_hours	Horas semanales de trabajo	Float	No

Company			
Descripción		Compañías registradas en la aplicación	
Atributo	Descripción	Tipo	Nullable
Id_company	Clave primaria, columna incremental	Integer	No
City	Ciudad de la compañía	Varchar	No
Country	País de la compañía	Varchar	No
Name	Nombre de la compañía	Varchar	No
Phone_number	Número de teléfono	Varchar	No
Postal_code	Código postal	Integer	No
Street	Dirección de la compañía	Varchar	No

Daily entry			
Descripción		Jornadas registradas en la aplicación	
Atributo	Descripción	Tipo	Nullable
Id_daily	Clave primaria, columna incremental	Integer	No
Cod_user	Identificador del usuario	Integer	No
Date_entry	Fecha del registro de la jornada	Date	No
Departure_time	Epoch del registro de entrada	Integer	Si
Entry_time	Epoch del registro de salida	Integer	No

Absence			
Descripción		Ausencias registradas en la aplicación	
Atributo	Descripción	Tipo	Nullable
Id_absence	Clave primaria, columna incremental	Integer	No
Cod_user	Identificador del usuario	Integer	No
Id_company	Fecha del registro de la jornada	Integer	No
Start_time	Epoch del registro de entrada	Integer	No
End_time	Epoch del registro de salida	Integer	No
Type	Código del tipo de ausencia	Integer	No
Validated	Validación de la ausencia por parte del administrador	Boolean	Si

Role			
Descripción		Tipos de roles disponibles en la aplicación	
Atributo	Descripción	Tipo	Nullable
Id_role	Clave primaria, identificador rol	Integer	No
Type_role	Descripción del rol	Varchar	No

User_role			
Descripción		Asociación entre rol y empleado	
Atributo	Descripción	Tipo	Nullable
Cod_user	Identificador del usuario	Integer	No
Id_role	Clave primaria, identificador rol	Integer	No

4.4 Front-end de la aplicación

El desarrollo de la parte front se realiza mediante el framework de typescript Angular 11 [27]. Utilizando este framework se ha podido dividir el desarrollo en una serie de componentes (modularizando la funcionalidad) que son acoplados posteriormente. La aplicación se carga en una sola página, es decir, es una aplicación SPA (Single Page Application). Una SPA es un sitio web que recarga y muestra su contenido en respuesta a acciones propias de la navegación sin tener que realizar nuevas peticiones al servidor para volver a cargar más código HTML, ya que se carga al completo al inicializar la aplicación.

También se hace uso de Bootstrap [28] como framework de interfaz de usuario, que contiene todo tipo de plantillas de diseño basadas en HTML y CSS y facilita el administrar las plantillas y crear sitios responsive, es decir, permite que la interfaz de usuario del sitio web funcione con igual rendimiento y calidad en cualquier tamaño de pantalla.

En el contenido web se pueden diferenciar tres secciones:

- Header: Se muestra el logo y nombre de la empresa, un botón para hacer el logout y, además, el menú de navegación para las distintas funcionalidades o pantallas. En función de los roles asignados al usuario que se ha logado podrá ver en la barra de navegación la sección de empleados de la compañía (solo los administradores tienen acceso a esta vista).
- Body: En esta sección se muestran los componentes necesarios que reflejan las funcionalidades implementadas en la aplicación.
- Footer: Se generan enlaces de acceso a las redes sociales y se muestra información sobre el copyright, autoría, política de cookies y textos legales.

4.5 Herramientas de apoyo

La principal herramienta que se ha utilizado como apoyo ha sido Postman [29]. Se trata de un cliente que permite realizar peticiones sobre APIs de manera muy sencilla y, de esta forma, poder testarlas. Las principales características de esta aplicación son:

- Permite crear peticiones http a servicios REST.
- Se pueden definir colecciones. Mediante Postman podemos agrupar APIs en colecciones.
- Permite generar código de invocación, es decir, dado un API se puede generar el código necesario para hacer las llamadas en diferentes lenguajes de programación.
- Permite crear un servidor de mockups para poder testar las APIs antes de que estén desarrolladas.

5. Validación del producto

Desde el comienzo del desarrollo de software, una gran parte de los recursos se ha dedicado a la resolución de errores aparecidos tras la entrega del proyecto. Por este motivo se han generado diversas normas para cuantificar y medir la calidad del software, como el conjunto de normas ISO 9000-2015, establecidas por la Organización Internacional de Normalización (ISO) [30].

Existen una amplia variedad de tipos de test [31], en función del modo en el que se analiza el comportamiento del sistema. Sin embargo, podemos diferenciar dos grandes bloques:

- Test de caja blanca: se analiza el código y la estructura del producto que se va a probar.
- Test de caja negra: se analiza el producto teniendo en cuenta únicamente las entradas que recibe y las salidas o respuestas que produce. No analiza el comportamiento interno.

Para el presente TFG los test que voy a desarrollar van a ser funcionales, que un tipo de test de caja negra y se basan en las especificaciones y requisitos del sistema desarrollado. Concretamente, van a ser test de aceptación (TA), cuyo objetivo es validar el funcionamiento del producto, comprobando que cumple con los requisitos esperados y acordados.

5.1 Fases en el desarrollo de los test funcionales

Las pruebas funcionales se dividen en las siguientes fases:

1. Identificación y análisis de requisitos: se necesita identificar las funciones que se espera que el producto sea capaz de llevar a cabo. Para ello hay que obtener toda la información posible de la aplicación.
2. Diseño del plan de pruebas: se identifican y especifican aquellos requisitos y funcionalidades que van a ser testadas. También

debe especificarse qué entradas se van a utilizar y cuáles son las salidas esperadas.

3. Ejecución de test: se ejecutan los test diseñados anteriormente. Esta ejecución puede realizarse de forma manual o automática.
4. Gestión de incidencias: si el resultado obtenido en el test no es el esperado se abre una incidencia en la que se describen todos los detalles del error para que el equipo o persona encargada pueda reproducirlo y subsanarlo.

5.2 Test de aceptación

El objetivo de estos test es confirmar el correcto funcionamiento del producto antes de realizar la entrega [32]. Para ello se comprueba que cumpla los requisitos acordados. Las principales ventajas de estos test son:

- Las funciones y las características que se van a probar son conocidas.
- Los detalles de las pruebas se conocen y se pueden medir.
- Las pruebas se pueden automatizar, lo que permite realizar pruebas de regresión.
- Los criterios de aceptabilidad son conocidos.

Entre las principales desventajas se incluyen:

- Requiere una planificación y recursos significativos.
- Es posible que las pruebas no revelen defectos profundos, ya que sólo busca defectos que espera encontrar.

5.3 Batería de pruebas

Escenario	Registro de nuevo administrador en empresa existente		
Código	TA.01		
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • La empresa ya está registrada 		
Pasos	Resultados esperados		
1. El usuario accede a la aplicación	Se muestra la pantalla de login		✓
2. Pulsa sobre el enlace "Don't you have an account?"	Se muestra el formulario de registro		✓
3. Introduce todos los datos del formulario y selecciona una empresa del desplegable.			
4. Pulsa sobre el botón "Create"	Se crea el nuevo usuario y se inserta en la base de datos		✓

Escenario	Registro de nuevo administrador y empresa		
Código	TA.02		
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado 		
Pasos	Resultados esperados		
1. El usuario accede a la aplicación	Se muestra la pantalla de login		✓
2. Pulsa sobre el enlace "Don't you have an account?"	Se muestra el formulario de registro		✓
3. Introduce todos los datos del formulario y pulsa sobre el botón "Create Company"	Se muestra el formu de registro de empresas		✓
4. Rellena los datos del formulario de New Company			
5. Pulsa sobre el botón "Create"	Se crea el nuevo usuario, la nueva empresa y se insertan en la base de datos		✓

Escenario	Registro de nuevo empleado	
Código	TA.03	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El administrador está registrado 	
Pasos	Resultados esperados	
1. El administrador accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales y hace el login	Se accede a la pantalla principal de la aplicación	✓
3. Pulsa sobre la pestaña “Employees – Employees”	Se muestra la pantalla de empleados	✓
4. Completa el formulario para añadir un nuevo empleado		
5. Pulsa sobre el botón “Add new employee”	Se crea el nuevo usuario y se inserta en la base de datos	✓

Escenario	Login erróneo	
Código	TA.04	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El usuario no está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce unas credenciales erróneas		
3. Pulsa sobre el botón “Enter”		
4. Se obtiene un mensaje de error	Aparece un mensaje de advertencia indicando el error	✓

Escenario	Login correcto	
Código	TA.05	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El usuario está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales		
3. Pulsa sobre el botón “Enter”		
4. Accede a la aplicación	Se accede a la pantalla principal de la aplicación	✓

Escenario	Solicitud de vacaciones correcta	
Código	TA.06	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El usuario está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales y hace el login	Se accede a la pantalla principal de la aplicación	✓
3. Pulsa sobre la pestaña "Absences"	Se accede a la pantalla de ausencias	✓
4. Selecciona el rango de fechas y en el desplegable de tipos de ausencias selecciona "Holidays"		
5. Pulsa sobre el botón "Request"	Se almacena en BD la solicitud y queda pendiente de la aprobación del administrador	✓

Escenario	Aceptación de vacaciones	
Código	TA.07	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El administrador está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales y hace el login	Se accede a la pantalla principal de la aplicación	✓
3. Pulsa sobre la pestaña o sobre el botón "Messages"	Se accede a la pantalla de mensajes	✓
4. Pulsa sobre el botón "Validate" para aceptarlas.	Se actualiza en la BD la solicitud de vacaciones	✓

Escenario	Denegación de vacaciones	
Código	TA.08	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El administrador está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales y hace el login	Se accede a la pantalla principal de la aplicación	✓
3. Pulsa sobre la pestaña o sobre el botón "Messages"	Se accede a la pantalla de mensajes	✓
4. Pulsa sobre el botón "Decline" para rechazarlas.	Se actualiza en la BD la solicitud de vacaciones	✓

Escenario	Consulta de nómina mensual	
Código	TA.09	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El usuario está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales y hace el login	Se accede a la pantalla principal de la aplicación	✓
3. Pulsa sobre la pestaña "Pay Slips"	Se accede a la pantalla de las nóminas	✓
4. Selecciona el mes y año del cual quiere obtener la nómina		
5. Pulsa sobre el botón "Request"	Se muestra la nómina del mes seleccionado	✓

Escenario	Consulta de horas trabajadas de un empleado	
Código	TA.10	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El administrador está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales y hace el login	Se accede a la pantalla principal de la aplicación	✓
3. Pulsa sobre la pestaña “Employees – Employees”	Se accede a la pantalla de los empleados	✓
4. Pulsa sobre la ficha del usuario que quiere analizar	Se muestra la pantalla del empleado con las horas trabajadas en los últimos 30 días	✓

Escenario	Registro de ausencias	
Código	TA.11	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El usuario está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales y hace el login	Se accede a la pantalla principal de la aplicación	✓
3. Pulsa sobre la pestaña “Absences”	Se accede a la pantalla de ausencias	✓
4. Selecciona el rango de fechas y en el desplegable de tipos de ausencias selecciona “Medical Leave” o “Personal Matters”		
5. Pulsa sobre el botón “Request”	Se almacena en BD la solicitud y queda pendiente de la aprobación del administrador	✓

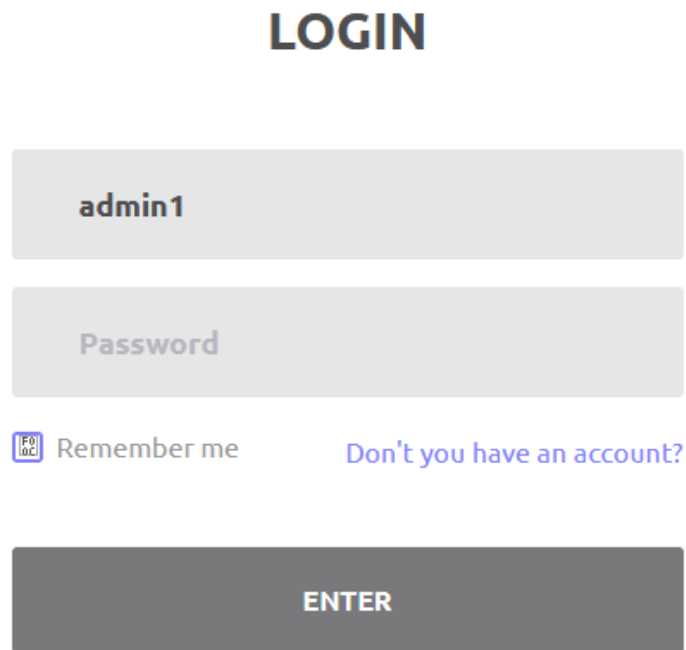
Escenario	Registro de jornada laboral	
Código	TA.12	
Precondiciones	<ul style="list-style-type: none"> • La aplicación se ha iniciado • El usuario está registrado 	
Pasos	Resultados esperados	
1. El usuario accede a la aplicación	Se muestra la pantalla de login	✓
2. Introduce sus credenciales y hace el login	Se accede a la pantalla principal de la aplicación	✓
3. Pulsa sobre la pestaña o sobre el botón "Clock In"	Se accede a la pantalla de registro horario	✓
4. Pulsa sobre el botón "Clockin" para marcar la entrada y sobre "Clockout" para la salida	Se marca la hora de entrada/salida y se almacena en la base de datos.	✓

6. Contenidos

A continuación, describo las páginas junto con los componentes que se incluyen en el cuerpo de todas ellas.

6.1 Login

Página inicial que permite autenticarse a los usuarios registrados.



LOGIN

admin1

Password

Remember me [Don't you have an account?](#)

ENTER

Ilustración 8. Pantalla login.

6.2 Registro de administradores y empresas

Página que muestra un formulario para el registro de administradores. Además, si la empresa que se quiere registrar no está disponible en el desplegable, se genera un formulario que permite registrar una nueva.

CREATE A NEW ADMIN

Username	
Name	
Email	
Password	Repeat it
Salary Year	Holidays
Payments	Weekly Hrs
IRPF	Phone
<input type="text"/>	
CREATE COMPANY	
<input type="checkbox"/> Remember me	Do you have an account? / Login
CREATE	

[f](#) [t](#) [G](#) [@](#) [in](#) [v](#)

Isaac Morales General

[UDC](#)
[Contact](#)
[Terms and Conditions](#)
[Privacy Policy](#)
[Cookies](#)

© 2021 Copyright: Clock-In.com

Ilustración 9. Pantalla registro administrador.

CREATE A NEW ADMIN

Username	
Name	
Email	
Password	Repeat it
Salary Year	Holidays
Payments	Weekly Hrs
IRPF	Phone

NEW COMPANY

Name	Phone
Country	City
Street	Postal code

Remember me [Do you have an account? / Login](#)

CREATE

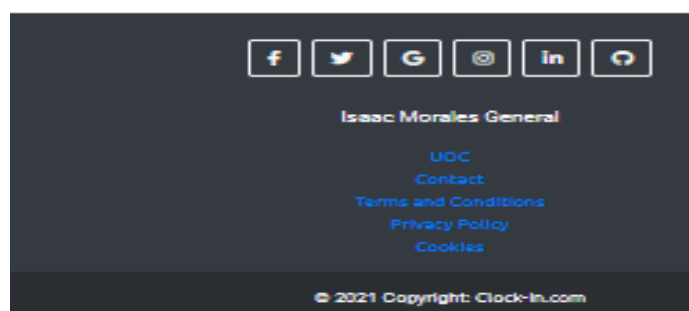


Ilustración 10. Pantalla registro administrador y empresa.

6.3 Home

Pantalla a la que se accede inmediatamente después de logarse el usuario. En ella se muestra información sobre el usuario, la empresa, las jornadas registradas en el día actual y un resumen de mensajes en el que se pueden ver las últimas ausencias solicitadas, validadas o pendientes de validar (en caso de un usuario con rol de administrador).

- Info-user: información del usuario logado.
- Info-company: información de la compañía del usuario.
- Daily-entry: registro de entradas del día actual.
- Message-summary: resumen de mensajes con las ausencias del usuario.

The screenshot displays the 'Home' page of the Clock-In application. At the top, a navigation bar includes the 'Clock-In' logo and menu items: Home, Clock In, Absences, Pay Slips, Messages, Employees, and Logout. The main content is divided into several sections:

- User details:** A table listing personal and account information for 'Administrator 1', including email, username, phone number, payments, salary, weekly hours, and holidays.
- Company details:** A table listing company information for 'companyX', including country, city, postal code, address, and phone number.
- Daily Entries:** A section for '6 jun. 2021' with a table for recording 'Entry' and 'Departure' times. A 'Clock In' button is visible below the table.
- Messages summary:** A section indicating 'You have 1 absences to validate' and 'You have 1 absences without validate'. It contains two tables: one for absences to be validated (showing user code 5, type HOLIDAYS, and dates 6-8 jun. 2021) and another for absences without validation (showing type HOLIDAYS and dates 6-8 jun. 2021). A 'Messages' button is located at the bottom right of this section.

At the bottom of the page, there is a footer with social media icons (Facebook, Twitter, Google+, Instagram, LinkedIn, RSS), the name 'Isaac Morales General', and links for 'UOC', 'Contact', 'Terms and Conditions', 'Privacy Policy', and 'Cookies'. The copyright notice '© 2021 Copyright: Clock-in.com' is at the very bottom.

Ilustración 11. Pantalla home.

6.4 Clock In

Página en la que se registra el inicio y el final de una jornada. Además, se muestra un gráfico con un histórico de 7 días en el que se observan las horas trabajadas por el usuario.

- Clockin-day: registra el inicio y el final de una jornada
- Graph-seven-days: gráfico con un histórico de 7 días en el que se observan las horas trabajadas por el usuario.

The screenshot displays the 'Clock-In' application interface. At the top, a navigation menu includes 'Clock-In', 'Home', 'Clock In', 'Absences', 'Pay Slips', 'Messages', 'Employees', and 'Logout'. The main content area is divided into two panels. The left panel, titled 'Clockin your working day', shows the date '6 jun. 2021' and a table with 'Entry' at '18:29:20 p. m.' and a 'Departure' field. A 'Clockout' button is visible. The right panel, titled 'Worked hours in last 7 days', features a bar chart with a y-axis from 0 to 0.0100 and an x-axis for days of the week (Mon-Sun). A single bar for Saturday indicates hours worked. The footer contains social media icons, the user name 'Isaac Morales General', links for 'UOC', 'Contact', 'Terms and Conditions', 'Privacy Policy', and 'Cookies', and a copyright notice '© 2021 Copyright: Clock-In.com'.

Ilustración 12. Pantalla clockin.

6.5 Absences

Pantalla en la que se solicitan vacaciones o se declaran ausencias por motivos médicos. También se muestra información sobre el usuario, la empresa y una lista de las últimas ausencias.

- Info-user: información del usuario logado.
- Info-company: información de la compañía del usuario.

- Holidays: formulario para solicitar vacaciones o declarar ausencias.
- Absences-list: lista de las últimas ausencias

The screenshot displays the 'Absences' page in the Clock-In application. At the top, a navigation bar includes 'Clock-In', 'Home', 'Clock In', 'Absences', 'Pay Slips', 'Messages', 'Employees', and 'Logout'. The main content is divided into several sections:

- User details:** A table listing user information: Name (Administrator 1), Email (admin@test1.com), Username (admin1), Phone number (123456123), Payments (12), Salary (12,00 €), Weekly Hours (12), and Holidays (12).
- Company details:** A table listing company information: Name (companyX), Country (Spain), City (Madrid), Postal code (28703), Address (calle), and Phone number (698745632).
- Holidays:** A section titled 'You have 10 free days yet!!' with a date range input field and a 'Request' button.
- Absences list:** A table showing one absence: 'HOLIDAYS' from '6 jun. 2021' to '8 jun. 2021'.

At the bottom, there are social media icons, contact information for 'Isaac Morales General' (UOC), and links for 'Contact', 'Terms and Conditions', 'Privacy Policy', and 'Cookies'. A copyright notice '© 2021 Copyright: Clock-In.com' is visible at the very bottom.

Ilustración 13. Pantalla absences.

6.6 Pay Slips

Pantalla en la que se solicita la nómina de un mes y año concreto. Esto lleva a una pantalla secundaria en la que se muestra esa nómina y aparece un botón para poder descargarla en PDF. También se muestra información sobre el usuario y la empresa.

- Info-user: información del usuario logado.
- Info-company: información de la compañía del usuario.
- Payslip-month: muestra la nómina y permite descargarla en PDF.

User details	
Name	Administrator 1
Email	admin@test1.com
Username	admin1
Phone number	123456123
Payments	12
Salary	12,00 €
Weekly Hours	12
Holidays	12

Company details	
Name	companyX
Country	Spain
City	Madrid
Postal code	28703
Address	calle
Phone number	698745632

Year Month

Ilustración 14. Pantalla payslips.

Payslip

Payslip -> Month: 5, Year: 2021

Description	Amount
Basic:	
Salary before taxes and discounts:	1,00 €
Worked hours:	-450.831,91
Absence hours:	0,00
Pluses:	
Discounts:	
Minus hours	-8.647,57 €
IRPF 12,0%	-1.037,59 €
Total:	
Salary after taxes and discounts:	-7.608,98 €

Ilustración 15. Pantalla nómina.

6.7 Messages

Página en la que se muestra las ausencias pendientes de validar por el administrador de la compañía, las últimas ausencias validadas y, en el caso de usuarios con rol de administrador, las ausencias de otros usuarios que debe validar o rechazar.

- Message: lista de ausencias.

Clock-In Home Clock In Absences Pay Slips Messages Employees Logout

Messages

1 absences to validate

User code	Type absence	Start Time	End Time	Validate
5	HOLIDAYS	6 Jun. 2021	8 Jun. 2021	Validate Decline

1 absences without validate for any admin

Type absence	Start Time	End Time
HOLIDAYS	6 Jun. 2021	8 Jun. 2021

Isaac Morales General

[UOC](#)
[Contact](#)
[Terms and Conditions](#)
[Privacy Policy](#)
[Cookies](#)

© 2021 Copyright: ClockIn.com

Ilustración 16. Pantalla messages.

6.8 Employees

(Solo visible para administradores) Página en la que se muestra información de la compañía, un formulario para registrar nuevos empleados y el listado de todos los usuarios que pertenecen a la compañía. Desde esta página clicando sobre uno de los usuarios se accede a otra página en la que se muestran detalles del usuario: la lista de ausencias, un gráfico con las horas trabajadas en los últimos 30 días y un formulario para editar los datos de ese usuario o eliminarlo.

- Info-company: información de la compañía del usuario.

- Register-employee: formulario para registrar empleados.
- Employee-details: muestra información detallada de un empleado de la compañía.
 - Absences-list: lista de las últimas ausencias
 - Edit-employee: formulario para editar información del usuario.
 - Graph-thirty.days: gráfico con un histórico de 30 días en el que se observan las horas trabajadas por el usuario.

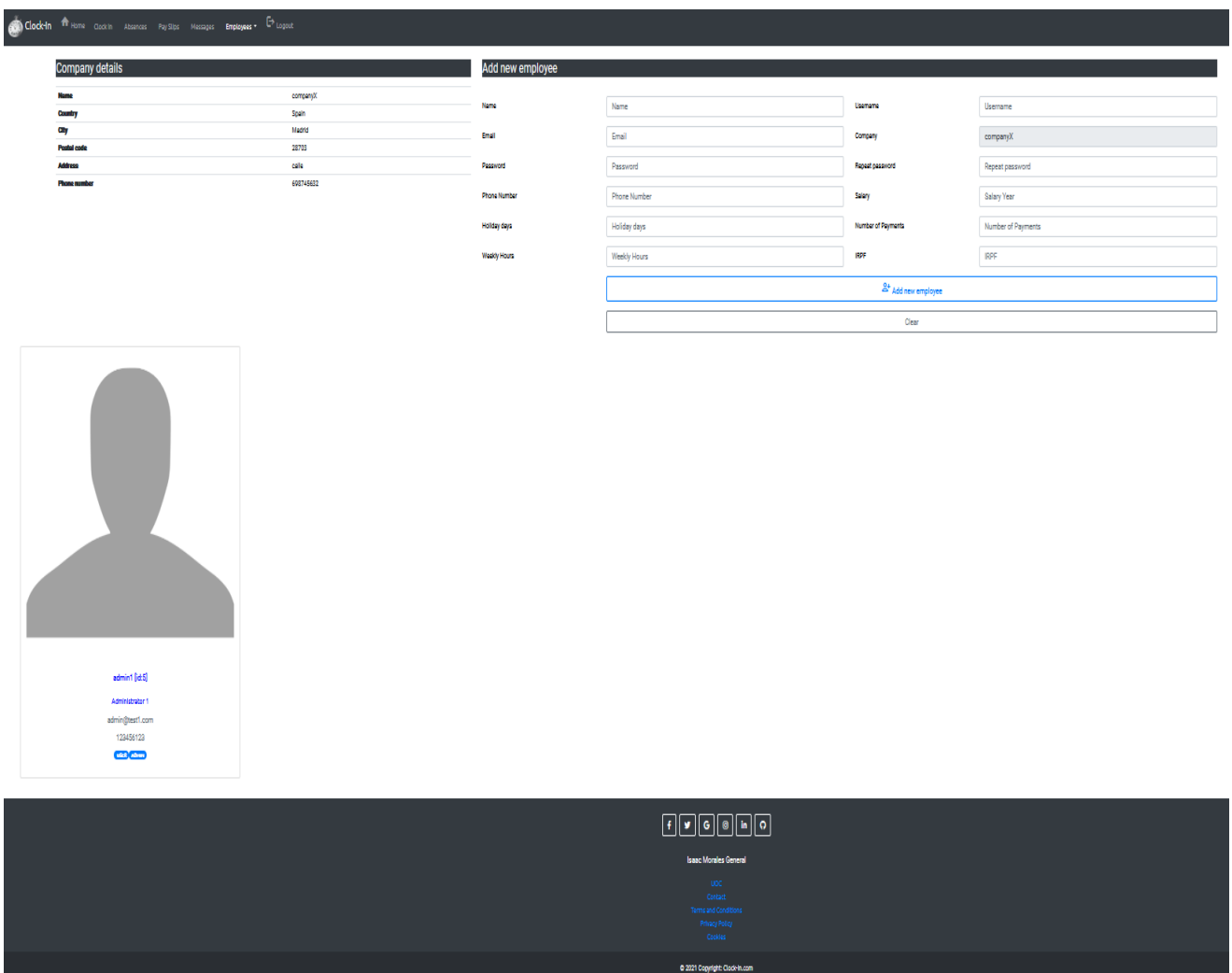


Ilustración 17. Pantalla employees.

Absences list

1 absences without validate for any admin

Type absence	Start Time	End Time
HOLIDAYS	6 jun. 2021	8 jun. 2021

[Messages](#)

Edit employee

Roles Admin Employee

ID employee ID company

Name Username

Email Company

Phone Number Salary

Holiday days Number of Payments

Weekly Hours IRPF

[Update employee](#)

[Reset](#)

[Remove employee](#)

Worked hours in last 30 days

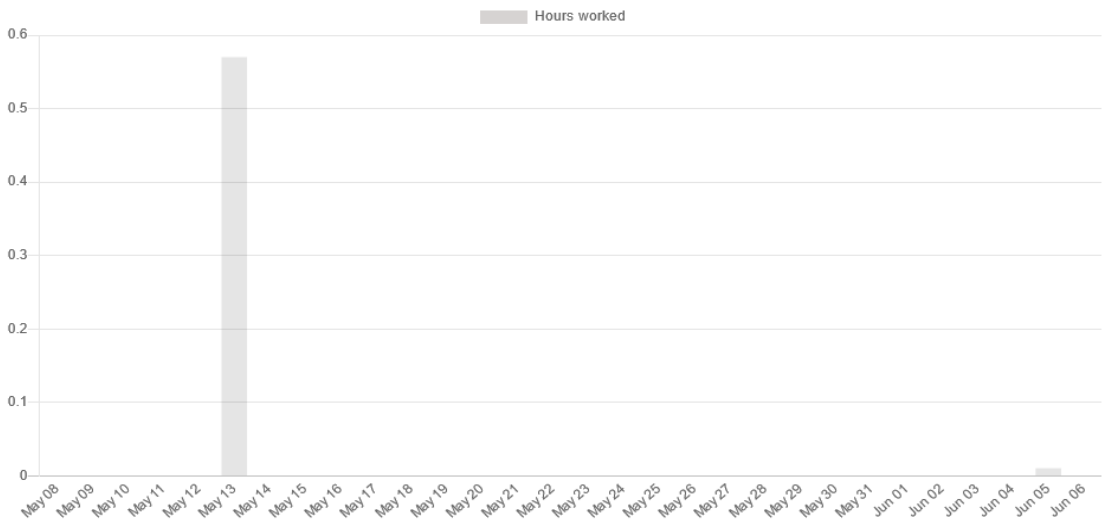


Ilustración 18. Pantalla de empleado.

6.9 Search

(Solo visible para administradores) Página en la que aparece un buscador que permite buscar empleados por nombre, email o username y los muestra en un listado y al clicar se puede acceder a la página de detalles de ese usuario.

- Search: buscador de usuarios.
- Employee-details: muestra información detallada de un empleado de la compañía.
 - Absences-list: lista de las últimas ausencias
 - Edit-employee: formulario para editar información del usuario.
 - Graph-thirty.days: gráfico con un histórico de 30 días en el que se observan las horas trabajadas por el usuario.

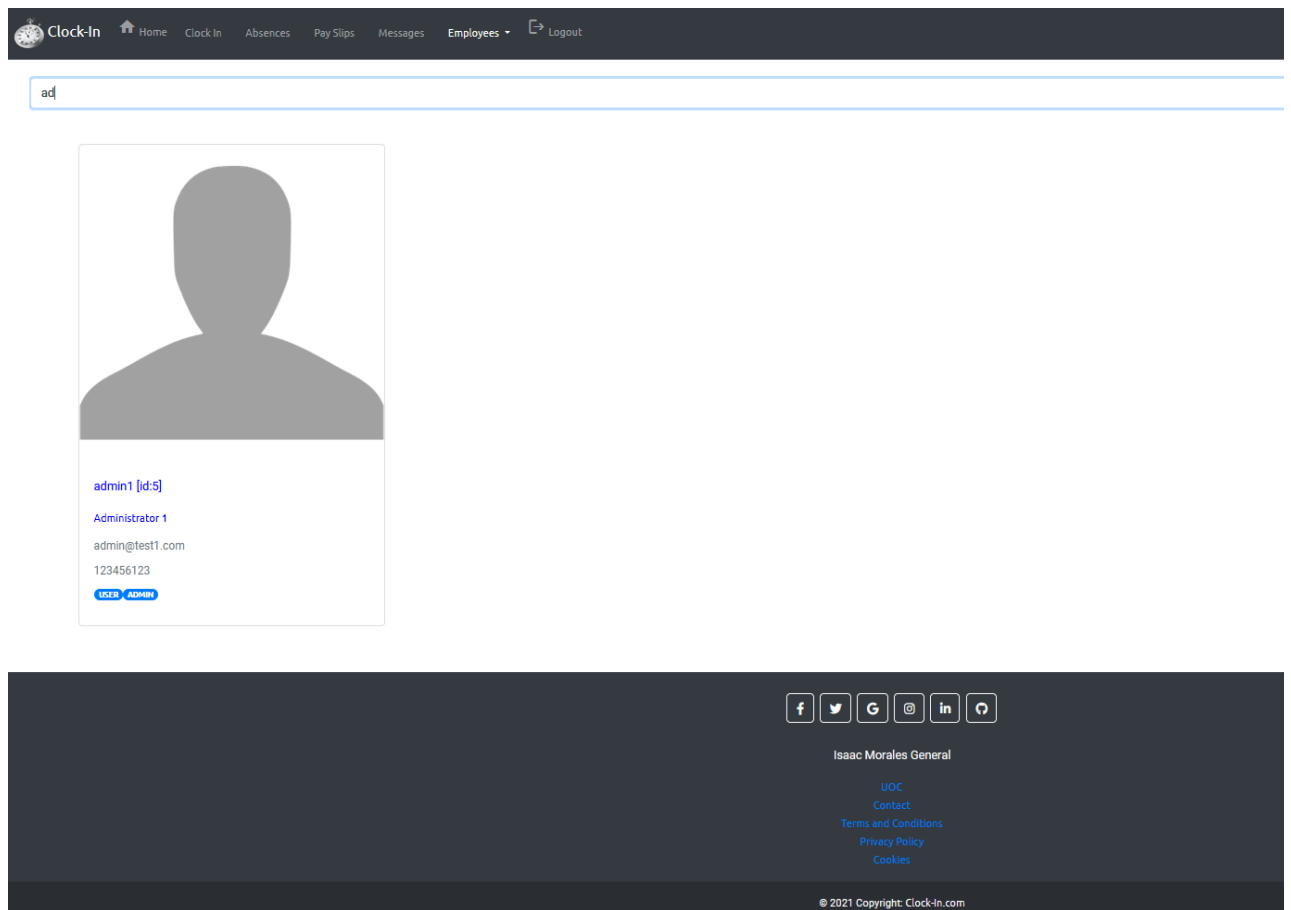


Ilustración 19. Pantalla search.

7. Conclusiones

Para finalizar el TFG se extraen una serie de conclusiones del trabajo realizado.

7.1 Lecciones aprendidas

Los principales problemas a los que me he enfrentado realizando este TFG han sido aquellos derivados del desconocimiento de algunas de las tecnologías utilizadas. Sobre todo, las referidas a la parte front-end (desconocía Angular por completo) y a la parte de securización de las APIs. Sin embargo, gracias a esto he aprendido que la curva de aprendizaje de nuevas tecnologías se va haciendo menor a medida que conoces otras.

Por otro lado, he aprendido la importancia del diseño, el llevar a cabo un diseño desde las etapas más tempranas me han hecho avanzar muy rápidamente en el momento de implementarlas.

7.2 Consecución de objetivos planteados

Los objetivos que se plantearon al comienzo del proyecto han sido alcanzados en su totalidad. Sin embargo, como explicaré en el apartado de líneas futuras de trabajo, considero que hay posibilidades de ampliar los objetivos de este proyecto y hacerlo más ambicioso.

7.3 Seguimiento de la planificación

En cuanto al seguimiento de la planificación del trabajo, considero que se ha llevado a cabo adecuadamente, ya que no se ha excedido ninguno de los plazos propuestos para cada tarea. Sin embargo, la carga de trabajo que he tenido que llevar a cabo ha sido superior a lo que tenía planteado en un principio, sobre todo, debido a la necesidad de aprender a manejar tecnologías desconocidas por mi hasta ese momento.

7.4 Líneas futuras de trabajo

En este apartado se describen algunas líneas de trabajo y funcionalidades que podrían implementarse en iteraciones posteriores del proyecto:

- Rediseño del front-end teniendo en cuenta de forma mucho más detallada la experiencia de usuario, es decir, todos aquellos factores que construyen la percepción de un usuario al interactuar con el sistema.
- Desarrollo de sistema de monitorización de los microservicios. Ampliando el ecosistema back-end con algunas piezas ya consolidadas como Spring Cloud Sleuth (librería que implementa una solución de trazado distribuido, para identificar automáticamente las peticiones entre microservicios) y Zipkin (que amplía la funcionalidad de Sleuth, trazando todas las trazas y almacenándolas para su posterior visualización en una interfaz gráfica).
- Ampliar la validación del producto con test unitarios en cada una de las piezas.
- Publicar la aplicación en un servidor de aplicaciones para que cualquier usuario que desee probarla tenga un entorno donde realizar sus pruebas y, posteriormente, pueda descargarse la aplicación del repositorio Git y modificarla según sus objetivos.

8. Glosario

- i. **DI:** Inyección de dependencias. Patrón de diseño de software usado en la Programación Orientada a Objetos, que trata de solucionar las necesidades de creación de los objetos de una manera práctica, útil y escalable.
- ii. **AOP:** Programación orientada a aspectos. Paradigma de programación en el que se modularizan las aplicaciones, otorgando una división de responsabilidades.
- iii. **POM:** Project Object Model. Unidad fundamental de trabajo en Maven. Es un fichero XML que contiene información sobre el proyecto y detalles de configuración usados por Maven para construir el proyecto.
- iv. **IETF:** Internet Engineering Task Force. Es el consorcio de cooperación técnica más importante de Internet.
- v. **RF:** Requisito funcional. Requisito que describe una función del sistema de software.
- vi. **RNF:** Requisito no funcional. Requisito que impone el cliente al programa que necesita, referente al diseño o a la implementación.
- vii. **JWT:** Jason Web Token. Es un token basado en JSON que sirve para enviar datos entre aplicaciones o servicios y garantizar que sean válidos y seguros.
- viii. **SPA:** Single Page Application. Aplicación o sitio web que cabe en una sola página con el propósito de dar una experiencia más fluida a los usuarios, como si fuera una aplicación de escritorio.

9. Bibliografía

- [1] POLO SÁNCHEZ, María Cristina, et al. Real Decreto-Ley 8/2019, de 1 de marzo, de Medidas urgentes de protección social y de lucha contra la precariedad laboral en la jornada de trabajo [BOE n.º 57, de 7-III-2019]. 2019.
- [2] PRESSMAN, Roger S.; TROYA, Jose Maria. Ingeniería del software. 1988.
- [3] ROYCE, Winston W. Managing the development of large software systems: concepts and techniques. En Proceedings of the 9th international conference on Software Engineering. 1987. p. 328-338.
- [4] BELL, Thomas E.; THAYER, Thomas A. Software requirements: Are they really a problem?. En Proceedings of the 2nd international conference on Software engineering. 1976. p. 61-68.
- [5] BORJA BUESTÁN, Carlos Daniel; CUJI TORRES, Valeria Alexandra. Metodología para la especificación de requerimientos de software basado en el estándar IEEE 830-1998. 2013. Tesis de Licenciatura.
- [6] FOWLER, Martin. UML distilled: a brief guide to the standard object modeling language. Addison-Wesley Professional, 2004.
- [7] NGUYEN, Quy; BAKER, Oras F. Applying Spring Security Framework and OAuth2 To Protect Microservice Architecture API. JSW, 2019, vol. 14, no 6, p. 257-264.
- [8] JONES, Michael; CAMPBELL, Brain; MORTIMORE, Chuck. JSON Web Token (JWT) profile for OAuth 2.0 client authentication and authorization Grants. May-2015.{Online}. Available: <https://tools.ietf.org/html/rfc7523>, 2015.
- [9] SAKIMURA, N.; JWT, JSON Web Token. Internet Engineering Task Force (IETF) M. Jones Request for Comments: 7519 Microsoft Category: Standards Track J. Bradley. 2015.
- [10] JWT. 2021. "JSON Web Tokens". Jwt.io. 2021. <https://jwt.io/>.
- [11] PROVOS, Niels; MAZIERES, David. Bcrypt algorithm. En USENIX. 1999.
- [12] ÁLVAREZ, Cecilio. Uso de Spring properties y encriptación. Arquitectura Java. 2019. <https://www.arquitecturajava.com/uso-de-spring-properties-y-encriptacion/>.

- [13] SURYOTRISONGKO, Hatma; JAYANTO, Dedy Puji; TJAHYANTO, Aris. Design and development of backend application for public complaint systems using microservice spring boot. *Procedia Computer Science*, 2017, vol. 124, p. 736-743.
- [14] URMA, Raoul-Gabriel; FUSCO, Mario; MYCROFT, Alan. *Java 8 in action*. Manning publications, 2014.
- [15] JOHNSON, Rod. *Expert one-on-one J2EE design and development*. John Wiley & Sons, 2004.
- [16] WALLS, Craig. *Spring Boot in action*. Manning Publications, 2016.
- [17] MILLER, Frederic P.; VANDOME, Agnes F.; MCBREWSTER, John. *Apache Maven*. Alpha Press, 2010.
- [18] DOMÍNGUEZ VARGAS, Karen Stehania, et al. *Sistema Integrado de Gestión Operativa (SIGO) empresa Ventas y Servicios SA*. 2013. Tesis de Licenciatura.
- [19] BHARATHAN, Raghuram. *Apache Maven Cookbook*. Packt Publishing Ltd, 2015.
- [20] LEWIS, James; FOWLER, Martin. *Microservices: a definition of this new architectural term*. MartinFowler. com, 2014, vol. 25, p. 14-26.
- [21] NEWMAN, Sam. *Building microservices: designing fine-grained systems*. " O'Reilly Media, Inc.", 2015.
- [22] NETFLIX. Eureka. <https://github.com/Netflix/eureka>.
- [23] NETFLIX. Zuul. <https://github.com/Netflix/zuul>.
- [24] GIT. Repositorio Git. <https://git.kernel.org/pub/scm/git/git.git/tree/>.
- [25] GINESTÀ, Marc Gibert; MORA, Oscar Pérez. *Bases de datos en PostgreSQL*. SI:[sn], 2012.
- [26] DOUGLAS, Korry; DOUGLAS, Susan. *PostgreSQL: a comprehensive guide to building, programming, and administering PostgreSQL databases*. SAMS publishing, 2003.
- [27] FREEMAN, Adam. *Pro Angular*. Apress, 2017.
- [28] SPURLOCK, Jake. *Bootstrap: Responsive Web Development*. " O'Reilly Media, Inc.", 2013.
- [29] CHRISTUDAS, Binildas. *cURL and Postman*. En *Practical Microservices Architectural Patterns*. Apress, Berkeley, CA, 2019. p. 847-855.

[30] MEDINA, Fanny Liliana Cruz; DÍAZ, Andrea del Pilar López; CARDENAS, Consuelo Ruiz. Sistema de gestión ISO 9001-2015: técnicas y herramientas de ingeniería de calidad para su implementación. Ingeniería Investigación y Desarrollo: I2+ D, 2017, vol. 17, no 1, p. 59-69.

[31] NIDHRA, Srinivas; DONDETI, Jagruthi. Black box and white box testing techniques-a literature review. International Journal of Embedded Systems and Applications (IJESA), 2012, vol. 2, no 2, p. 29-50.

[32] DAVIS, Fred D.; VENKATESH, Viswanath. Toward preprototype user acceptance testing of new information systems: implications for software project management. IEEE Transactions on Engineering management, 2004, vol. 51, no 1, p. 31-46.

10. Anexo

10.1 Manual de instalación

El código se puede descargar del repositorio de Github:
<https://github.com/imoralesgeneral/Clockin>.

En primer lugar, se debería ejecutar el script de base de datos para poder crear el modelo. A continuación, se deben arrancar los distintos microservicios, comenzando por Eureka y Config-server.

Por último, se descargan los módulos necesarios para la parte front y se arranca desde consola de comandos con “npm start”.