



UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



TFG del Grado en Ingeniería  
Informática

Generador de cuestionarios y  
escenarios de test sobre soft  
skills



Presentado por Daniel Fernández Barrientos  
en Universidad de Burgos — 11 de junio  
de 2024

Tutor: Raúl Marticorena Sánchez







UNIVERSIDAD DE BURGOS  
ESCUELA POLITÉCNICA SUPERIOR  
Grado en Ingeniería Informática



D. Raúl Marticorena Sánchez, profesor del departamento de Ingeniería Informática, área de Lenguajes y Sistemas informáticos.

Expone:

Que el alumno D. Daniel Fernández Barrientos, con DNI 51079809-Y, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado Generador de cuestionarios y escenarios de test sobre soft skills.

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

En Burgos, 11 de junio de 2024

Vº. Bº. del Tutor:

D. Raúl Marticorena Sánchez





## Resumen

Las habilidades blandas o “soft skills” [20] son las habilidades personales y sociales que complementan las habilidades técnicas en el entorno laboral.

En los entornos de trabajo actuales, donde la mayoría de los trabajadores están altamente cualificados o sobrecualificados en algunos casos, las capacidades personales para interactuar con compañeros y responsables, así como para afrontar diversas tareas o situaciones de alto estrés, son cruciales para diferenciar entre un trabajador competente y uno ideal para un puesto de trabajo.

Analizar o contrastar datos sobre la evolución del estado de ánimo de los trabajadores o sobre el nivel de empatía entre personas o departamentos es una herramienta muy potente para las direcciones de recursos humanos y para las áreas de psicología de las empresas. Esto permite conocer de antemano las capacidades y debilidades de la plantilla, y anticiparse a situaciones predecibles.

El proyecto está dirigido a proporcionar a todas las empresas, tanto PYMES como grandes corporaciones, una herramienta accesible, fácilmente personalizable y que permite la gestión de un gran número de cuestionarios para analizar las habilidades necesarias.

## Descriptores

Habilidades blandas, cuestionario, spring boot, aplicación web, docker.

## **Abstract**

Soft skills [20] are personal and social abilities that complement technical skills in the workplace.

In today's work environments, where most workers are highly qualified or even overqualified in some cases, personal abilities to interact with colleagues and supervisors, as well as to handle various tasks or high-stress situations, are crucial to differentiate between a competent worker and an ideal candidate for a job.

Analyzing or comparing data on the evolution of employees' moods or the level of empathy between people or departments is a powerful tool for HR departments and company psychology teams. This allows them to understand in advance the strengths and weaknesses of the staff and to anticipate predictable situations.

The project aims to provide all companies, both SMEs and large corporations, with an accessible, easily customizable tool that allows the management of a large number of questionnaires to analyze the necessary skills.

## **Keywords**

Soft skills, questionnaire, spring boot, web app, docker.



---

# Índice general

---

<b>Índice general</b>	<b>iii</b>
<b>Índice de figuras</b>	<b>v</b>
<b>Índice de tablas</b>	<b>vi</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Estructura de la memoria . . . . .	2
1.2. Materiales adjuntos . . . . .	3
<b>2. Objetivos del proyecto</b>	<b>5</b>
2.1. Objetivos generales . . . . .	5
2.2. Objetivos técnicos . . . . .	5
2.3. Objetivos personales . . . . .	6
<b>3. Conceptos teóricos</b>	<b>7</b>
<b>4. Técnicas y herramientas</b>	<b>9</b>
4.1. Spring Tool Suite for Eclipse . . . . .	10
4.2. L <sup>A</sup> T <sub>E</sub> X . . . . .	10
4.3. MikT <sub>E</sub> X . . . . .	10
4.4. Spring Boot . . . . .	10
4.5. Thymeleaf . . . . .	12
4.6. HTML . . . . .	12
4.7. MySQL . . . . .	13
4.8. Docker . . . . .	13
4.9. GitHub . . . . .	14

4.10. JSON . . . . .	14
4.11. Zube.io . . . . .	14
4.12. GitPod . . . . .	15
<b>5. Aspectos relevantes del desarrollo del proyecto</b>	<b>17</b>
5.1. Inicio del proyecto . . . . .	17
5.2. Metodologías . . . . .	17
5.3. Formación . . . . .	18
5.4. Desarrollo de la aplicación web . . . . .	19
5.5. Documentación . . . . .	22
5.6. Publicación . . . . .	23
<b>6. Trabajos relacionados</b>	<b>25</b>
6.1. Web RiskReal.eu . . . . .	25
<b>7. Conclusiones y Líneas de trabajo futuras</b>	<b>29</b>
7.1. Conclusiones . . . . .	29
7.2. Líneas de trabajo futuras . . . . .	30
<b>8. Agradecimientos</b>	<b>31</b>
<b>Bibliografía</b>	<b>33</b>

---

## Índice de figuras

---

5.1. Despliegue de la aplicación en la plataforma web de GitPod. . . . .	23
5.2. Funcionamiento de la aplicación web a través de GitPod. . . . .	23
6.1. Datos de registro de prueba. . . . .	26
6.2. Error tras enviar el formulario de <i>Ir a la zona para empresas</i> . . . . .	27

---

# Índice de tablas

---

4.1. Herramientas y tecnologías utilizadas en cada parte del proyecto	9
---	---

---

# 1. Introducción

---

Las habilidades blandas, conocidas también como *soft skills*, desempeñan un papel importante en la mayoría de las áreas de la vida, tanto en el ámbito profesional como en las relaciones interpersonales.

En el mundo laboral actual se están empezando a valorar más este tipo de habilidades, llegando a poder ser excluyentes.

La capacidad de trabajar en equipo, una comunación efectiva o la ética del trabajo son solo algunos ejemplos de este tipo de habilidades, altamente demandadas por los responsables en los procesos de contratación, sobretudo en un entorno laboral donde la tipología de trabajo híbrido o 100 % presencial aparece en la mayoría de ofertas de empleo.

A nivel de contratación, las empresas no solo buscan a personas que estén cualificadas, sino que también se interesan por saber qué actitudes o cualidades tienen.

Poder encontrar un perfil que encaje con el entorno en el que va a trabajar, es crucial para el éxito de todas las partes, por eso no solo se deben evaluar este tipo de habilidades en nuevos procesos, sino que se tiene que realizar un proceso continuo de evaluación dentro de las empresas.

Conocer por qué en un grupo hay mucha rotación, o por qué en otros la mitad de los compañeros no se hablan entre sí es también fundamental para un crecimiento adecuado tanto de la empresa como de las personas, un ambiente de trabajo agradable estimula la productividad y el espíritu de pertenencia que tanto se ha perdido últimamente.

Poder generar cuestionarios personalizados para estudiar una o varias habilidades de forma simultánea es una gran ventaja para el análisis de conductas y actitudes.

RiskRealApp es una aplicación web, sencilla de desplegar y altamente personalizable, donde se asigna un valor a cada pregunta y donde los evaluadores pueden obtener un fichero de *CSV* con toda la información, separada por puntuación elegida a nivel de pregunta como a nivel total, por *ID* de cuestionario o por fecha de realización, para poder generar tantas estadísticas como se necesiten.

El uso de tecnologías como *docker* facilitan la implantación de esta aplicación, siendo portable, escalable e instalable en los principales sistemas operativos del mercado: *Windows*, *Linux* y *macOS*.

## 1.1. Estructura de la memoria

La memoria tiene la siguiente estructura:

- **Introducción:** Breve descripción de las habilidades blandas, y la importancia de poder medirla de forma continua.
- **Objetivos del proyecto:** Exposición de los objetivos que persigue el proyecto.
- **Conceptos teóricos:** Explicación de los conceptos teóricos analizados para comprender la solución propuesta.
- **Técnicas y herramientas:** Técnicas y herramientas utilizadas en el desarrollo del proyecto.
- **Aspectos relevantes del desarrollo del proyecto:** Aspectos más destacables que han tenido lugar durante la realización del proyecto.
- **Trabajos relacionados:** Aplicación RiskReal[21] actual y páginas web similares.
- **Conclusiones y líneas de trabajo futuras:** Conclusiones obtenidas tras la finalización del proyecto y futuras mejoras que se podría aplicar.

De forma complementaria a la memoria, se proporcionan los siguientes anexos:

- **Plan de proyecto software:** Planificación temporal y estudio de viabilidad del proyecto.

- **Especificación de requisitos:** Se definen los objetivos generales, los requisitos funcionales y no funcionales, y los diferentes casos de uso.
- **Especificación de diseño:** Se describen las diferentes fases de diseño del proyecto.
- **Documentación técnica de programación:** Se describen los aspectos relacionados con el entorno de programación más relevantes, como la estructura de directorio o la forma de compilar una nueva versión de la aplicación.
- **Documentación de usuario:** Guía para una correcta instalación y manejo de la aplicación por parte de los usuarios.
- **Anexo de sostenibilidad curricular:** Aspectos relevantes de la sostenibilidad aplicados en el proyecto.

## 1.2. Materiales adjuntos

Los materiales que se adjuntan con la memoria son:

- Aplicación java RiskRealApp.
- Cuestionarios de prueba.
- JavaDoc.

Los siguientes materiales están disponibles a través de internet:

- Repositorio web del proyecto [16].
- Imagen de docker de la aplicación [15].





---

## 2. Objetivos del proyecto

---

A continuación, se detalla los diferentes objetivos que han impulsado la realización del proyecto.

### 2.1. Objetivos generales

- Desarrollar una aplicación web que permita la realización de cualquier tipo de cuestionario sobre habilidades blandas.
- Crear una aplicación sencilla y portable, fácilmente utilizable, y que permita cierta configuración adicional de base.
- Almacenar los resultados en un archivo estructurado y sencillo de analizar.
- Permitir que la aplicación pueda llegar a la mayoría de los países del entorno, estando disponible en varios idiomas.

### 2.2. Objetivos técnicos

- Utilizar un *framework* de Java para el desarrollo de la aplicación.
- Convertir la aplicación web en lo más portable posible, utilizando tecnologías como *docker*.
- Aplicar la arquitectura *modelo-vista-controlador*. separando la parte del código que se encarga de presentar la imagen de la que se encarga de obtener los datos.

- Incluir un pool de conexiones a una base de datos para almacenar los datos de los usuarios de forma persistente.
- Utilizar GitHub como plataforma de control de versiones.
- Utilizar *maven* como herramienta de automatización en la construcción de la aplicación y la imagen de *docker*.

## 2.3. Objetivos personales

- Adentrarme en el *framework de Spring Boot*, algo totalmente desconocido para mí hasta el momento, y permitirme ver lo sencillo que es la creación de una aplicación web, una vez que se conocen todas sus bondades.
- Explorar el ecosistema completo de la programación web, tanto por lo anterior como por el uso de motores de plantillas y de generación de código *HTML* de forma dinámica con *Thymeleaf*.
- Profundizar en *docker* y su ecosistema, una tecnología con gran auge en el mercado laboral actual.
- La creación de una aplicación que tenga claramente diferenciado el *front-end* (páginas *HTML*) del *back-end* (código en java) y con persistencia fuera de la aplicación (base de datos y almacenamiento local o centralizado en el equipo anfitrión).

---

## 3. Conceptos teóricos

---

La parte del proyecto que me ha supuesto mayor complejidad teórica ha sido la relacionada con la tecnología de *Spring Boot* [8]. Entender cómo funciona *docker* [1] tampoco ha sido una tarea sencilla, pero el sí necesaria, tanto para hacer más accesible la aplicación como para potenciar mis habilidades técnicas.



---

## 4. Técnicas y herramientas

---

En esta parte de la memoria se van a presentar las distintas herramientas y técnicas utilizadas en el desarrollo del proyecto. Se incluirán todas las que se hayan valorado o utilizado, incluso aquellas que tras ciertas pruebas o pasos, se hayan descartado por haber encontrado una alternativa más funcional. A continuación, una tabla resumen con las herramientas utilizadas en cada parte del proyecto:

Herramientas	App Web	BBDD	Memoria
HTML5	X		
CSS	X		
BOOTSTRAP	X		
Java	X		
Spring Boot	X		
Eclipse	X		
Thymeleaf	X		
Docker	X	X	
JSON	X	X	
CSV	X	X	
MySQL		X	X
GitHub	X	X	X
Zube.io	X	X	X
MikTeX			X
L <sup>A</sup> T <sub>E</sub> X			X

Tabla 4.1: Herramientas y tecnologías utilizadas en cada parte del proyecto

## 4.1. Spring Tool Suite for Eclipse

Herramienta de desarrollo (IDE) que se utilizará para llevar a cabo el proyecto. A pesar de haber otras similares, como IntelliJ, he preferido utilizar la *suite* de *Spring Boot* para Eclipse al estar más familiarizado con el entorno de trabajo, ya que ha sido la herramienta utilizada en otras asignaturas del grado.

## 4.2. L<sup>A</sup>T<sub>E</sub>X

El editor de texto para generar toda la documentación relacionada con el trabajo de fin de grado será L<sup>A</sup>T<sub>E</sub>X. Aún estando acostumbrado a utilizar Microsoft Word para este tipo de tareas, veo en utilizar L<sup>A</sup>T<sub>E</sub>X una oportunidad de aprender un nuevo generador de documentos. La curva de aprendizaje es más grande que con Word (en mi caso), pero el resultado merece la pena porque permite aplicar formatos de forma sencilla, un aspecto en el que Word para documentos muy grandes, está en clara desventaja.

## 4.3. MikT<sub>E</sub>X

Para trabajar con plantillas de L<sup>A</sup>T<sub>E</sub>X en entornos windows existen varias alternativas, en mi caso he preferido utilizar el programa MikT<sub>E</sub>X. Es un programa sencillo y tiene por separado tanto la parte de edición como la de presentación, para ir viendo los cambios una vez se compila la plantilla completa.

## 4.4. Spring Boot

Spring Boot [23] es la evolución del clásico *framework* de creación de aplicaciones web.

La construcción de una aplicación web con un *framework* antiguo consistía en los siguientes pasos:

1. Crear un proyecto de java.
2. Importar las dependencias necesarias.
3. Preparar todos los archivos de configuración necesarios para que funcione la aplicación.

4. Desplegar la aplicación en un servidor web.

Este mismo proceso con Spring Boot consta de los siguientes pasos:

1. Acceder a *Spring Initializr* [12].
2. Indicar el tipo de proyecto, el lenguaje de programación y los *starters* [14] de Spring necesarios.
3. Generar el proyecto.

El paso 2 de la lista anterior engloba todos los pasos de la primera lista, incluso va más allá, porque el proyecto que genera se puede arrancar desde la propia *suite* de Spring y tendrías tu aplicación web funcionando en muy poco tiempo.

## **Starters Utilizados**

A continuación, se van a comentar ciertos aspectos de los *starters* que se han utilizado en la construcción de la aplicación.

### **Spring Web**

Establece la configuración del proyecto para su funcionamiento como una aplicación web. Al incluir un servidor *Apache Tomcat* no es necesario generar el archivo *JAR* o *WAR* y desplegarlo en un servidor, solo con levantar la aplicación esta ya se lanza en el servidor web embebido.

### **Spring Security**

Establece la configuración de seguridad por defecto en la aplicación web. Sin más configuración adicional, genera una contraseña por defecto de administrador con cada inicio de la aplicación, ya que securiza todas las páginas web de forma predeterminada.

Con ciertas clases de java, Spring Security [19] permite personalizar la configuración de la aplicación.

## Spring JPA

Establece toda la integración para usar clases relacionadas con tablas de una base de datos. Elimina la necesidad de crear patrones DAO [17] para nuestras clases y simplifica este proceso de forma considerable.

1. Creamos nuestra clase de java.
2. Incluimos las anotaciones [22] en la clase.
3. Creamos una interfaz que extienda de *JPA repository* [11] o de *CRUD repository* [9] asociada a nuestra clase.
4. Ya disponemos de todo lo necesario para actualizar, borrar, crear o recuperar objetos de la base de datos.

## 4.5. Thymeleaf

Es un motor de generación de plantillas y de generación dinámica de código *HTML*.

Permite incluir cierta lógica en las etiquetas *HTML* de nuestras páginas web para mostrar u ocultar opciones, iterar sobre un objeto o lista para crear tantas filas en una tabla como valores.

Permite el poder enviar variables desde nuestro controlador de Java a la página web, reconocer esas variables y mostrar su contenido.

También incluye el soporte para los archivos de personalización, pudiendo acceder a los archivos de propiedades en función del idioma seleccionado en la web.

El uso de los *fragments* es un gran aliado para ahorrarnos líneas de código.

## 4.6. HTML

Lenguaje utilizado en la creación de las páginas web, en particular HTML5, que tiene algunas funcionalidades básicas ya implementadas, como por ejemplo el uso de “required” en campos de formulario para que sea obligatorio rellenar los campos sin necesidad de utilizar JavaScript para el control.



## 4.7. MySQL

Se utiliza MySQL como base de datos para almacenar los usuarios y roles. Se ha decidido usar una base de datos porque su integración con Spring Security es muy sencilla y funcional. A través de los repositorios CRUD que nos proporciona Spring JPA podemos gestionar las tablas y registros de forma ágil.

## 4.8. Docker

Un *docker* o contenedor, es una imagen de un sistema operativo básico con una o varias aplicaciones instaladas que al arrancarse, permite el acceder a esta aplicación sin necesidad de dedicar una gran fuente de recursos a su funcionamiento.

Es realmente mucho más sencillo que todo eso, se podría simplificar con lo siguiente: **Docker es un servidor corriendo como un microservicio.**

Lo mejor de Docker es que las imágenes son multiplataforma, con tener un servicio de docker instalado en el equipo, puede correr cualquiera de las imágenes disponible en su repositorio.

Facilita el despliegue y portabilidad de nuevas aplicaciones, a la vez que disminuye el tamaño de la infraestructura necesaria para su funcionamiento.

Se ha utilizado docker tanto para el servidor de MySQL como para la propia aplicación Web, de esta forma con 2 simples contenedores, un docker-compose (fichero de configuración de dockers que permite descargarse, crear, parar o lanzar dockers) podemos tener toda la aplicación completa corriendo en cualquier entorno, ya que docker funciona en los principales sistemas operativos del mercado: Linux, Windows y macOS.

### Docker Compose

Esta herramienta complementaria de docker, *docker compose* [13] permite definir y gestionar aplicaciones que requieran más de un contenedor, de forma conjunta y sencilla. Se ha utilizado como base para la instalación del proyecto, explicando sus funcionalidades y permitiendo ciertas personalizaciones en el despliegue de la aplicación.

## 4.9. GitHub

Se va utiliza GitHub para dar visibilidad al trabajo diario y constante en el proyecto, ya que permite realizar aportaciones incrementales de código, documentación, etc del TFG completo.

Es la herramienta por excelencia para llevar un control de versiones en cualquier proyecto, sea software o no.

### Repository

Se ha creado un repositorio [16] donde se subirá todo el TFG al completo, tanto la documentación de la memoria, como la aplicación de java.

### Releases

En esta sección dentro del repositorio se han ido subiendo las diferentes versiones funcionales de la aplicación, para tener un lugar central desde el que descargarse la última versión “liberada”.

### Packages

En esta parte de GitHub se han subido las imágenes de docker funcionales de la aplicación, para permitir un despliegue sencillo desde una ubicación centralizada. Con un simple comando de *docker pull url de la imagen* puede descargarte la imagen para empezar a utilizarla.

## 4.10. JSON

Se van a utilizar ficheros con la estructura de un json para almacenar los cuestionarios completos, a los que se accederá desde la aplicación web de Java.

## 4.11. Zube.io

Se va a utilizar la versión gratuita de Zube.io, que permite generar Sprints, y gráficos de seguimiento, como *Burndown* o *Burnup*.

Gracias a la sencilla y completa integración con Github ofrece una versión más completa de la parte de Project de Github, por lo que he decidido

utilizar esta plataforma para el seguimiento de las tareas en vez de la que ofrecía Github.

La versión gratuita permite integrar lo necesario del repositorio y del proyecto, por lo que no es necesario ampliar a versiones más completas.

Aunque se puede compartir con otros usuarios y colaboradores del proyecto, no se permite el acceso público.

## 4.12. GitPod

Se ha estudiado la posibilidad de usar *gitpod* para alojar una versión funcional de la aplicación, ya que diferentes alternativas como *Heroku* o *Render* ni permitían un fácil despliegue con `docker-compose.yml` ni daban la opción de una prueba real de forma gratuita.

A pesar de haber vinculado *gitpod* con *GitHub*, se limitaba a 50 horas el despliegue de una aplicación, pero eso no se incluye como parte del proyecto.



---

## 5. Aspectos relevantes del desarrollo del proyecto

---

Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, incluyendo las diferentes decisiones que se han ido tomando y los problemas que se han tenido que afrontar.

### 5.1. Inicio del proyecto

La selección del proyecto suponía un doble reto personal:

- Nunca había programado aplicaciones web ni conocía la tecnología de Spring Boot.
- No había un proyecto base, documentado, como inicio del proyecto, sino que era crear algo nuevo desde cero.

Tras la primera reunión con el tutor, donde me expuso el camino que se quería seguir con el proyecto, me puse a recabar información sobre ello.

### 5.2. Metodologías

Desde el inicio del proyecto se ha dedicado tiempo y esfuerzo en que la realización del proyecto fuese de la manera más profesional posible.

La gestión del proyecto se ha realizado teniendo como base la metodología ágil de *Scrum* [7].

Al tratarse de un Trabajo de Fin de Grado, que se realiza de forma individual, no se siguió de forma completa, porque el grupo de desarrolladores estaba formado solo por un miembro.

Cómo se ha aplicado la metodología:

- Reunión inicial de *kick-off* [6] del proyecto, donde se presentó el proyecto y se establecieron ciertos objetivos generales.
- Se utilizó una estrategia de desarrollo incremental basada en *sprints*, a través de iteraciones y revisiones.
- La duración media de los *sprints* fue de dos semanas, excepto los últimos del proyecto, que han sido de una semana.
- En la finalización de cada *sprints* se entregaba un incremento del proyecto.
- Se realizaba una reunión de revisión del *sprint* que servía también como reunión de planificación del nuevo *sprint*.
- Se definía en el tablero kanban de *Zube.io* una lista de tareas y se asociaban al nuevo *sprint*.
- Para monitorizar el progreso se han utilizado gráficos *burnup*.

### 5.3. Formación

La principal tecnología en el desarrollo del proyecto ha sido *Spring Boot*, sobre la que no se tenía ningún conocimiento previo, lo que me ha llevado a tener que formarme por diferentes medios:

- Spring Boot 3. Aplicaciones web y REST APIs con Spring MVC (Udemy) [18].
- Cómo convertir en imagen de docker una aplicación de Spring Boot [10].
- Cómo aplicar estilos de personalización de *bootstrap* y *CSS* en documentos *HTML*.

Me gustaría destacar la web [Baeldung](#), al contener mucha información sobre Spring Boot en general.

Aunque en menor medida, también he tenido que formarme en otras de las tecnologías utilizadas:

- Uso y configuración de un archivo *docker compose* [\[13\]](#).
- Cómo se utiliza  $\text{\LaTeX}$  para la generación de documentos.

## 5.4. Desarrollo de la aplicación web

Se va a dividir el desarrollo de la aplicación web en varios apartados, para poder detallar mejor el proceso de construcción de la misma.

### Código de la aplicación

El desarrollo del código de la aplicación supuso un primer reto, pues en el inicio del proyecto todavía no terminaba de comprender cómo iba a crear una aplicación web con Java, ya que no era lo que estaba acostumbrado a crear con este lenguaje de programación.

Una vez se comprendió cómo funcionaba, de forma básica, *Spring Web* como *starter* del proyecto, este paso se convirtió en algo trivial.

La primera parte de la creación de la aplicación consistía en poder “leer” un archivo *JSON* y convertirlo a objetos en Java con los que se pudiese trabajar de forma nativa.

Se crearon varias clases modelo para agrupar el contenido del *JSON*:

- Clase *Quiz*: Contiene la descripción básica del cuestionario y un listado de preguntas.
- Clase *Questions*: Contiene la información de cada pregunta y un listado de respuestas.
- Clase *Answers*: Contiene la información de cada respuesta.

Una vez conseguido lo anterior, la base del proyecto ya estaba construida.

Para la presentación de contenidos en las páginas web, se usaron clases *controller* en Java, que sirven de enlace entre la propia aplicación y el código *HTML* de la página web.

A través de estas clases, se pasaban los cuestionarios completos al código *HTML* de la página web y en ella, con *Thymeleaf* se mostraba la información.

Ya se tenía la forma de ver el contenido del *JSON* en las páginas web.

Hasta aquí tendríamos una base más “robusta” de la aplicación, teniendo todo lo necesario dentro del propio proyecto de Java.

Los primeros problemas los encontramos al intentar actualizar los archivos de propiedades, ya que al ser internos de la aplicación, necesitaban un reinicio de la misma cada vez que había un cambio, algo que no era funcional para un entorno de producción.

A raíz de lo anterior, se empezó a trabajar con archivos *JSON* externos a la aplicación, utilizando rutas relativas para poder encontrar los archivos siempre, independientemente de la ubicación raíz del proyecto.

Finalmente, se incluyeron varios tipos de clases de Java adicionales:

- Clases de configuración: Se utilizaron ciertas clases para modificar la configuración por defecto de Spring Boot. También se ha utilizado para acceder a un archivo de propiedades personalizado, donde se definen algunos elementos de configuración y personalización de la aplicación.
- Clases de servicio: Contienen los métodos a los que se tienen que acceder desde otras clases, evitando duplicidad en el código y teniendo una clase centralizada para la gestión de la mayoría de los procesos, siendo más sencillo comprender su código y ahorrando métodos adicionales en otras clases.

## Código de las páginas web

La estética y el diseño de las páginas web ha sido lo más complejo en cuanto a presentación final del proyecto.

Ayudado de *bootstrap* y personalización a través de plantillas *CSS* se ha conseguido aplicar un estilo homogéneo a toda la aplicación.

El uso de los fragmentos de *Thymeleaf* ha ayudado a que los ficheros *HTML* que componen la página web sean más legibles, evitando tener que escribir las mismas líneas de forma repetitiva, simplificándolo todo a una simple línea que enlaza con el *fragmento* indicado.

Definitivamente, implementar las etiquetas *HTML* ha sido lo más costoso en cuanto a programación se refiere, porque conseguir cuadrar y encajar todos los apartados en su sitio ha sido un trabajo de prueba y error constante.



## Integración de servicios y dependencias

Los servicios o dependencias más destacables que implementa la aplicación son los siguientes:

### Servicio de envío de correos electrónicos

Durante el proceso de desarrollo de un servicio de correo electrónico que enviase a los usuarios una nueva contraseña en caso de que no recordasen la suya, se probaron configuraciones con varios servidores de correo, pero finalmente, por simplicidad y acceso a documentación, se utilizó *Gmail* que permite la creación de cuentas gratuitas y el uso de sus servidores de correo desde cualquier *API* siempre que se utilice un *token* [3].

La configuración completa del servidor de correo, así como, sus credenciales, se independizaron de la aplicación en un archivo de propiedades, para permitir su modificación por cualquier otro servicio de correo, dando libertad al programador para cambiarlo sin necesidad de compilar de nuevo el proyecto.

### Seguridad con Spring Security y Spring JPA

En primer lugar, tras aplicar *Spring Security*, la mayoría de las páginas dejaron de funcionar, todas aquellas que eran de tipo *POST*.

Tras investigar descubrí que esto sucedía porque la primera medida de seguridad que se aplica, es la necesidad de recibir un *CSRF token* (*Cross-Site Request Forgery*) [2] para evitar ataques *man in the middle* a los formularios de las aplicaciones web, al requerir un *token* de autenticación, para comprobar la autenticidad de la comunicación.

Debido a esto, todos los formularios tienen configurado un campo “oculto” que envía este *token* al controlador de la aplicación, para permitir el acceso.

Se decidió utilizar una base de datos *mysql* que se levanta cada vez que se inicia la aplicación, almacena los datos en una carpeta externa a la misma, para tener persistencia, y que está vinculada con el la seguridad de la aplicación, ya que se definieron los campos de nuestras tablas de usuarios y perfiles que se iban a usar tanto para iniciar sesión, como para comprobar a qué parte de la aplicación tiene cada uno.

## 5.5. Documentación

Para generar la documentación del proyecto, se ha utilizado una plantilla de  $\text{\LaTeX}$ , que permite mantener una estructura uniforme y el mismo estilo durante la documentación de todo el proyecto, aportando un aura de profesionalidad y sin los problemas y complicaciones de formatos que nos podemos encontrar en otros programas comerciales.

La documentación de la aplicación se ha generado desde el propio editor de código, *Spring Tool Suite 4 for Eclipse*, siguiendo el formato *javadoc* [5].

Este formato permite la visualización de todas las clases utilizadas en formato *HTML*, pudiendo navegar por los diferentes métodos y obteniendo una breve descripción de cada uno, así como de los parámetros de entrada que esperan o del retorno de cada método.

## 5.6. Publicación

Se ha intentado realizar la publicación de la aplicación en la plataforma en la nube de *GitPod* [4], porque permitía la integración de un repositorio de *GitHub* y poder desplegar tus aplicaciones con un archivo *docker-compose.yml*, pero las limitaciones del plan gratuito en cuanto a horas mensuales de ejecución no han permitido tenerla disponible de forma continua, sí pudiendo comprobarse su correcto despliegue y funcionamiento.

Despliegue de la aplicación con “docker compose up -d” en *GitPod*:

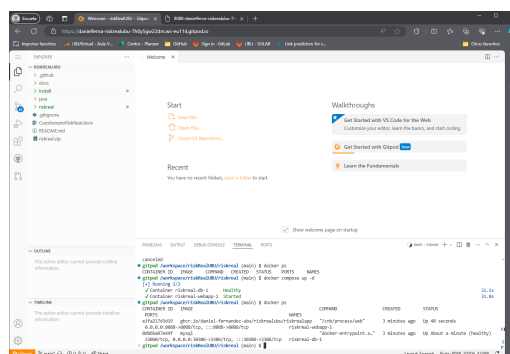


Figura 5.1: Despliegue de la aplicación en la plataforma web de *GitPod*.

Acceso a la aplicación web a través de la dirección web que genera *GitPod*:

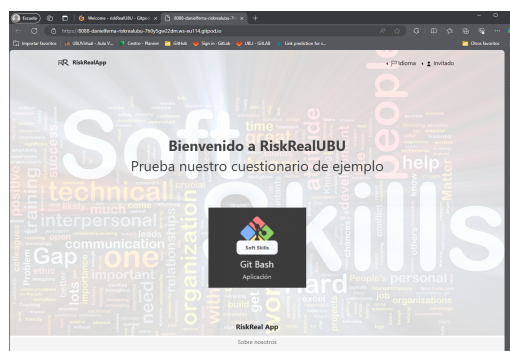


Figura 5.2: Funcionamiento de la aplicación web a través de *GitPod*.



---

## 6. Trabajos relacionados

---

Sobre este proyecto en particular, solo hay un trabajo relacionado, la aplicación web App.RiskReal.eu, la cuál ha servido de base de requisitos para la realización del proyecto actual.

### 6.1. Web RiskReal.eu

Es la página web [21] que se ha utilizado como base para la creación del nuevo proyecto. La aplicación se basa en la utilización de diferentes cuestionarios para evaluar “soft skills” de trabajadores.

#### Descripción general

Desde la página principal se ofrece una descripción básica de lo que se puede hacer. Indica que con diversos escenarios de test, se pueden evaluar de forma eficiente las diferentes “soft skills”.

Consta de dos páginas base, *Inicio*, que consideraría la parte *abierta* de la web, y *Cursos*, con acceso restringido solo para usuarios registrados.

#### Parte abierta

Permite realizar un cuestionario propio, para obtener una evaluación orientativa de en qué estado nos encontramos en cuanto a habilidades blandas; y un test sobre un posible escenario, donde el “trabajador” va contestando en función de diferentes situaciones.

No permite retroceder en las preguntas, tan solo se puede ir avanzando.

El diseño de los tipos de cuestionario está diferenciado de inicio:

- Cuestionario tipo test: Una pregunta con varias posibles respuestas, bien categóricas o binarias.
- Cuestionario tipo escenario: Una pregunta con varias imágenes a modo de “situaciones” y varias respuestas, donde se tiene que elegir la situación que más se acerque a lo que cada uno haría.

El no poder mezclar todo tipo de preguntas me parece una limitación en cuanto a la variedad de cuestionarios a poder realizar.

Lo anterior se ha tenido en cuenta para que cada pregunta se gestione a nivel de pregunta, no a nivel de cuestionario completo, pudiendo mezclar los diferentes tipos de preguntas en el mismo cuestionario.

## Parte privada

Se realiza un intento de registro con los siguientes datos:

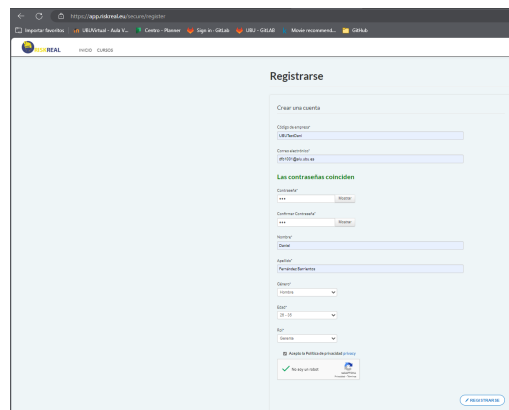
The image shows a web browser window with a registration form titled "Registrarse". The form is for creating a new account. It includes fields for "Correo electrónico" (Email), "Contraseña" (Password), "Confirmar Contraseña" (Confirm Password), "Nombre" (Name), "Apellido" (Surname), "Fecha de Nacimiento" (Date of Birth), "Sexo" (Gender), "País" (Country), and "Código Postal" (Postal Code). There are also checkboxes for "Acepto la Política de Privacidad y el Aviso" (I accept the Privacy Policy and the Notice) and "Recibir emails" (Receive emails). A green message "Las contraseñas coinciden" (The passwords match) is displayed above the password fields. The form is set against a light blue background with a white sidebar on the left.

Figura 6.1: Datos de registro de prueba.

Tras ello, no se recibe ningún correo de confirmación ni se consigue acceder a la propia aplicación. Debido al contratiempo anterior, no se puede analizar más en detalle como es la parte “privada” de la aplicación.

Tampoco se permite el registro en la zona de empresas, obteniendo el mensaje que se puede ver en la siguiente imagen:

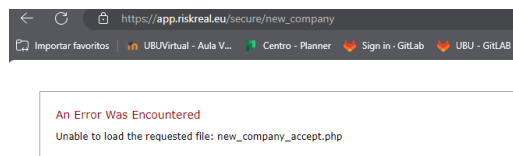


Figura 6.2: Error tras enviar el formulario de *Ir a la zona para empresas*.





---

## 7. Conclusiones y Líneas de trabajo futuras

---

En este capítulo se exponen las conclusiones derivadas del trabajo del proyecto, así como las líneas de trabajo futuras por las que se puede dar continuidad al proyecto.

### 7.1. Conclusiones

Tras el desarrollo del proyecto se obtienen las siguientes conclusiones:

- Se ha cumplido con el objetivo general del proyecto, teniendo ahora disponible una aplicación web que permite la realización y gestión de cuestionarios.
- Haber utilizado tecnologías como Spring Boot y Docker, le dan al proyecto un aura de modernidad, al ser tecnologías que están en expansión dentro del ámbito del desarrollo de aplicaciones.
- He podido aplicar conocimientos de gran parte de lo aprendido durante el grado, y aprender sobre muchos otros en los que no contaba con ninguna experiencia previa.
- Integrar la creación del proyecto con herramientas de control de versiones ha facilitado el desarrollo incremental del proyecto, así como un *know-how* de buenas prácticas para cualquier proyecto *software* que tenga que desarrollar en el futuro.

- Estimar el tiempo total dedicado a pruebas y prototipos de la aplicación es muy complicado, ya que en algunos momentos me he atascado con parte de la programación y he tenido que probar vías alternativas, pero el uso de la metodología ágil para la gestión de las tareas ha facilitado el seguimiento del progreso.

## 7.2. Líneas de trabajo futuras

La entrega del Trabajo de Fin de Grado solo es el inicio del proyecto, ya que su desarrollo continúa.

A continuación, se resume el camino a seguir del proyecto:

- Integrar en la página web la posibilidad de editar los cuestionarios ya cargados, sin tener que actualizar el archivo *JSON* y volver a cargarlo.
- Integrar la aplicación con una *API* de traducción que permita generar de forma automática los cuestionarios en otros idiomas.
- Integrar la posibilidad de crear cuestionarios desde cero desde la propia página web, contando con la ventaja de tener un modelo de datos ya definido.
- Se puede considerar la opción de migrar la aplicación a plataformas móviles, para no depender de la conexión a internet para acceder.

---

## 8. Agradecimientos

---

Quería utilizar un capítulo a parte para expresar mis agradecimientos personales, tanto por este Trabajo de Fin de Grado, como por estos años en la Universidad de Burgos.

Mi primera experiencia con una universidad fue hace más de 12 años, donde empecé por primera vez con Ingeniería Informática de forma presencial, pero no era el momento ni el lugar para llevar a cabo esa tarea.

Tengo que decir, que aún siendo en modalidad *online*, la cercanía que he sentido durante estos 4 años y medio que llevo en la universidad me ha hecho sentir por momentos que estaba en persona en las clases.

Las facilidades que he encontrado en esta universidad para poder quitarme la espinita de cursar Ingeniería Informática y llegar hasta 4º, tras mucho esfuerzo y tras mucho aprendizaje, me llenan de emoción y de orgullo de decir que he estudiado en la Universidad de Burgos.

El agradecimiento está dirigido a todo el personal de la universidad, ya que a pesar de la distancia, siempre he recibido un trato rápido y cercano.

Me llevo la experiencia y la suerte de haber tenido grandes docentes, dispuestos a responder a un correo electrónico un fin de semana, o fuera de su jornada laboral, de hacer tutorías cuando mejor nos venía a todos los alumnos, contando con que la mayoría trabajamos y acababan siendo a última hora de la tarde.

La profesionalidad y vocación del personal de la universidad de los más altos que he vivido, muchas gracias por todos estos años.

Por último, y no menos importante, quiero agradecer a Raúl Marticorena Sánchez, mi tutor en este Trabajo de Fin de Grado, y mi profesor en algunas

de las asignaturas del grado, su especial dedicación y compromiso con los alumnos.

Sin pretender extenderme mucho más, mi más sincero agradecimiento a la Universidad de Burgos.

---

## Bibliografía

---

- [1] Docker docs. <https://docs.docker.com/>. [Internet; Accedido junio de 2024].
- [2] Csrftoken. <https://stackoverflow.com/questions/5207160/what-is-a-csrf-token-what-is-its-importance-and-how-does-it-work>, 2016. [Internet; actualizado 2016].
- [3] Crear token en gmail. <https://soporteacc.com/hc/articles/136/crear-contrasena-de-aplicacion-para-gmail>, 2024. [Internet; accedido 2024].
- [4] Gitpod. <https://www.gitpod.io/>, 2024. [Internet; actualizado 2024].
- [5] Introduction to javadoc. <https://www.baeldung.com/javadoc>, 2024. [Internet; actualizado 2024].
- [6] Kick off. <https://blog.hubspot.es/marketing/reunion-kick-off>, 2024. [Internet; actualizado 2024].
- [7] Metodología scrum. <https://www.atlassian.com/es/agile/scrum>, 2024. [Internet; actualizado 2024].
- [8] Spring boot. <https://spring.io/projects/spring-boot>, 2024. [Internet; actualizado 2024].
- [9] Spring boot crud repository. <https://docs.spring.io/spring-data/commons/docs/current/api/org/springframework/data/repository/CrudRepository.html>, 2024. [Internet; actualizado 2024].

- [10] Spring boot docker información. <https://spring.io/guides/topicals/spring-boot-docker>, 2024. [Internet; actualizado 2024].
- [11] Spring boot jpa repository. <https://docs.spring.io/spring-data/jpa/docs/current/api/org/springframework/data/jpa/repository/JpaRepository.html>, 2024. [Internet; actualizado 2024].
- [12] Spring inicializr. <https://start.spring.io/>, 2024. [Internet; actualizado 2024].
- [13] Tutorial docker compose. <https://imaginaformacion.com/tutoriales/que-es-docker-compose>, 2024. [Internet; actualizado 2024].
- [14] Tom Hombergs Baeldung. Spring boot starterst. <https://www.baeldung.com/spring-boot-starters>, 2024. [Internet; actualizado 2024].
- [15] Daniel Fernández Barrientos. Imágen de docker de la aplicación riskrealapp. <https://github.com/Daniel-Fernandez-UBU/riskRealUBU/pkgs/container/riskrealubu/riskrealapp/227834130?tag=latest>, 2024. [Internet; actualizado 2024].
- [16] Daniel Fernández Barrientos. Repositorio riskrealubu — github. <https://github.com/Daniel-Fernandez-UBU/riskRealUBU>, 2024. [Internet; actualizado 2024].
- [17] Oscar Blancarte. Patrón dao. <https://www.oscarblancarteblog.com/2018/12/10/data-access-object-dao-pattern/>, 2018. [Internet; actualizado 2018].
- [18] Ivan Eliseo Tinajero Diaz. Spring boot 3. aplicaciones web y rest apis con spring mvc — udemy. <https://www.udemy.com/course/spring-framework-desarrollo-web-spring-mvc>, 2023. [Internet; actualizado noviembre-2023].
- [19] Jonathan Faber. Spring security. <https://adictosaltrabajo.com/2020/05/21/introduccion-a-spring-security/>, 2020. [Internet; actualizado 2020].
- [20] Kelly Main Monique Danao. Forbes - softskills. <https://www.forbes.com/advisor/in/business/soft-skills-examples/>, 2024. [Internet; actualizado 2024].

- [21] RiskReal. Soft skills detection. <https://app.riskreal.eu/>, 2022. [Internet; actualizado 2022].
- [22] Cecilio Álvarez Caules. Anotaciones jpa. <https://www.arquitecturajava.com/jpa-table-y-entity/>, 2023. [Internet; actualizado 2023].
- [23] Cecilio Álvarez Caules. Arquitectura spring boot. <https://www.arquitecturajava.com/que-es-spring-boot/>, 2023. [Internet; actualizado 2023].