



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**RiskReal: Desarrollo de un
portal web para comprobar
las soft skills
Documentación Técnica**



Presentado por Marcos Ubierna Fernández
en Universidad de Burgos — 9 de julio de 2024
Tutor: Raúl Marticorena Sánchez

Índice general

Índice general	i
Índice de figuras	iii
Índice de tablas	v
Apéndice A Plan de Proyecto Software	1
A.1. Introducción	1
A.2. Planificación temporal	1
A.3. Estudio de viabilidad	3
Apéndice B Especificación de Requisitos	9
B.1. Introducción	9
B.2. Objetivos generales	9
B.3. Catálogo de requisitos	10
B.4. Especificación de requisitos	12
Apéndice C Especificación de diseño	21
C.1. Introducción	21
C.2. Diseño de datos	21
C.3. Diseño procedimental	25
C.4. Diseño arquitectónico	27
Apéndice D Documentación técnica de programación	29
D.1. Introducción	29
D.2. Estructura de directorios	29
D.3. Manual del programador	30

D.4. Compilación, instalación y ejecución del proyecto	31
D.5. Pruebas del sistema	35
Apéndice E Documentación de usuario	37
E.1. Introducción	37
E.2. Requisitos de usuarios	37
E.3. Instalación	37
E.4. Manual del usuario	39
Apéndice F Anexo de sostenibilización curricular	51
F.1. Introducción	51
Bibliografía	55

Índice de figuras

A.1. Flujo de commits	2
B.1. Diagrama de Casos de Uso	13
C.1. Diagrama Entidad relacion	23
C.2. Estructura del test	24
C.3. Estructura del cuestionario	24
C.4. Diagramas de secuencia: Registro de Usuario	25
C.5. Diagramas de secuencia: Inicio de Sesión	26
C.6. Diagramas de secuencia: Exportación de Resultados	26
C.7. Diagramas de actividad: Realización de test/cuestionario	27
C.8. Patrón MVC	28
D.1. Comando para clonar el repositorio	31
D.2. Creación del fichero .env	32
D.3. Creación del entorno virtual	32
D.4. Activación del entorno virtual	32
D.5. Instalación de las librerías	32
D.6. Vinculación del fichero main.py con Flask	33
D.7. Inicialización de migrate	33
D.8. Instancia de las tablas	33
D.9. Generación de las tablas	33
D.10.Lanzamiento de la aplicación	34
D.11.Creación de la imagen de docker	34
D.12.Creación y lanzamiento del contenedor	34
D.13.Contenedor generado en Docker Desktop	35
E.1. Comando para importar la imagen a Docker	38
E.2. Creación del contenedor de Docker	38

E.3. Ejecución del contenedor	39
E.4. Pestaña de inicio sin logearse	40
E.5. Pestaña de inicio como administrador	41
E.6. Pestaña de registro	41
E.7. Pestaña de login	42
E.8. Pestaña de inserción de los datos del invitado	43
E.9. Pestaña del test en la primera pregunta	44
E.10. Pestaña del test en una pregunta intermedia	44
E.11. Pestaña del test en la última pregunta	45
E.12. Pestaña del cuestionario en la primera pregunta	45
E.13. Pestaña del cuestionario en una pregunta intermedia	46
E.14. Pestaña del cuestionario en la última pregunta	46
E.15. Pestaña del resultado tras evaluarse	47
E.16. Tabla de los usuarios en la primera página	47
E.17. Tabla de los usuarios en la segunda página	48
E.18. Tabla de los resultados en la primera página	48
E.19. Tabla de los resultados en la segunda página	49
E.20. Archivo ccv abierto desde excel	49
F.1. Objetivos de Desarrollo Sostenible	52

Índice de tablas

A.1. Coste anual por recursos hardware y software.	4
A.2. Coste anual por personal e infraestructura.	5
A.3. Coste anual total	6
A.4. Librerías del proyecto con su versión y licencia	7
B.1. CU-1 Registro de Usuario.	14
B.2. CU-2 Inicio de Sesión.	15
B.3. CU-3 Realización de Tests/Cuestionarios.	16
B.4. CU-4 Visualización de Usuarios Registrados.	17
B.5. CU-5 Visualización de los resultados.	18
B.6. CU-6 Exportación de Resultados.	19

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este anexo se va a exponer la planificación temporal junto con la viabilidad legal y económica del proyecto.

A.2. Planificación temporal

Introducción

Para gestionar el proyecto se ha utilizado Github junto con Zube, primero disponemos de una parte temporal la cual no se disponía del repositorio Github en la cual se prueba la viabilidad del proyecto, después tenemos el desarrollo del proyecto junto con los sprints y por ultimo una parte que no se encuentra en Github correspondiente a la implementación de la documentación. En este flujo podemos observar como los comits han sido realizados.

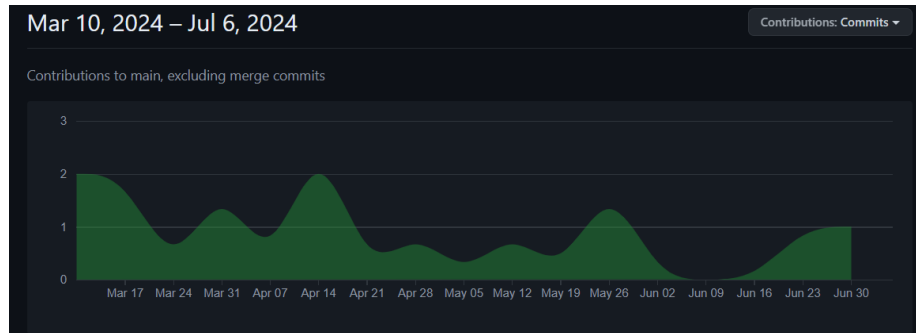


Figura A.1: Flujo de commits

Sprints

- Sprint previo 20/2/2024 – 22/3/2024: probar la viabilidad realizando un pequeño test que tenga más de 2 preguntas y varias preguntas cada una, además de recorrer el resultado y mostrarlo por pantalla, creación de usuarios de manera forzada con un array y creación de pestaña para logearse.
- Sprint 1 22/3/2024 – 3/4/2024: Implementación de la documentación. Añadir sesiones al test. Mostrar imágenes en el test. Mejorar la visualización de las preguntas. Se aplica la librería flask-session para poder operar con las sesiones y utilización de Bootstrap para mejorar la interfaz de la página web.
- Sprint 2 3/4/2024 – 17/4/2024: Base de datos con usuarios y test resueltos. Implementación de registro. En caso de no estar logeado hacer el test como un invitado. Guardar valores de los test con su nombre de usuario, fecha y puntuación. Opción de recuperación de contraseña. Ir avanzando en documentación. Utilización de flask-sqlalchemy y flask-migrate para el tratamiento de las BBDD.
- Sprint 3 17/4/2024 – 2/5/2024: Documentación (general y manual de instalación). Corregir fallo con el usuario invitado. Implementar campos restantes en el registro. Formulario para resetear la contraseña. Implementar parte inicial al cuestionario. Mostrar usuario que está conectado en la página principal. Creación del manual de programador, intento de realización del cambio de contraseña utilizando verificación de dos pasos, implementación de la pestaña previa si se realiza el test como invitado.

- Sprint 4 2/5/2024 – 15/5/2024: Creación de administradores que puedan ver las tablas resultados y usuarios. Implementación del otro tipo de test. Cambiar imágenes del test y añadir descripción. Mejora del cambio de contraseña. Avanzar en la documentación. Creación del rol administrador con permiso para ver los test y los usuarios de las BBDD, creación del cuestionario y adaptación de los valores a los originales aparte de rediseñar la ruta de las funciones para el invitado, intento de cambiar el cambio de contraseña.
- Sprint 5 15/5/2024 – 29/5/2024: Mejora del cambio de contraseña. Pulimiento de estética. Cambio de estética general y cambio del tratamiento del cambio de contraseña utilizando una pregunta secreta elegida aleatoriamente.
- Sprint 6 29/5/2024 – 6/6/2024: Memoria e implementación opción descargar resultados para administrador. Opción para descargar en un csv los resultados de los tests y avance en la documentación.
- Sprint Final 6/6/2024 – 8/7/2024: terminar memoria y anexos, comentar código, cambios mínimos y utilización de docker.

A.3. Estudio de viabilidad

En esta parte del apéndice se va a realizar un estudio sobre la viabilidad económica y legal del proyecto.

Viabilidad económica

A la hora de estudiar la posible viabilidad económica, se deben tener en cuenta varios aspectos como el gasto de los dispositivos utilizados, licencias e infraestructura.

Costes generales

En este apartado se incluirá lo relacionado con el uso del hardware y software del proyecto. Todo hardware tiene un coste no como algunas licencias de software que son open source, pero se tendrán en cuenta aquellas que no.

1. **Costes del hardware:** He utilizado mi ordenador personal que tiene las siguientes especificaciones y en su momento tuvo un coste de 800€.

- CPU Intel(R) Core(TM) i7-7700HQ
- 8 GB de RAM
- GPU NVIDIA GeForce GTX 1050

La media de utilización de un portátil suele ser sobre los 4 años, por lo que el coste anual es:

$$\text{Coste anual} = \frac{800 \text{ €}}{4 \text{ años}} = 200 \text{ €/año} \quad (\text{A.1})$$

2. **Costes del software:** Para el proyecto se ha utilizado Docker el cual dispone de una licencia estándar gratuita, pero en caso de querer dar un uso mas avanzado es recomendable tener la versión pro como mínimo que tiene un coste de 5 € mensuales.

$$\text{Coste anual} = 5 \text{ €} \times 12 \text{ meses} = 70 \text{ €/año} \quad (\text{A.2})$$

Por otra parte el uso de Github y Visual Studio Code son licencias gratuitas por lo que se tendrá en cuenta. Por último el sistema operativo utilizado que es Windows 10 ya no se encuentra en venta por lo que utilizaremos como referencia el Windows 11 Pro el cual su licencia cuesta 259€.

$$\text{Coste anual} = \frac{259 \text{ €}}{4 \text{ años}} = 64,75 \text{ €/año} \quad (\text{A.3})$$

Teniendo en cuenta todos los costes tendríamos el resultado de la siguiente tabla:

Recurso	Precio/año
Ordenador	200,00 €
Licencia Docker	70,00 €
Windows 11 Pro	64,75 €
Total:	334,75 €

Tabla A.1: Coste anual por recursos hardware y software.

Costes de infraestructura y personal

Hay que tener en cuenta que a la hora de realizar un trabajo es necesario tener una sede, por lo cual se debe realizar un alquiler o compra de un local, el cual el precio medio de alquiler en España ronda a los 300€/mes.

$$\text{Coste anual} = 300 \text{ €} \times 12 \text{ meses} = 3600 \text{ €/año} \quad (\text{A.4})$$

En cuanto al personal el único trabajador soy yo, lo cual el sueldo correspondiente para un desarrollador Junior es de media 21.000€ anuales.

$$\text{Salario bruto mensual} = \frac{21000 \text{ €}}{14 \text{ pagas}} = 1500 \text{ €} \quad (\text{A.5})$$

Por lo cual el coste anual sera:

$$\text{Coste anual} = 1500 \text{ €} \times 12 \text{ meses} = 18000 \text{ €/año} \quad (\text{A.6})$$

Teniendo en cuenta todos los costes tendríamos el resultado de la siguiente tabla:

Recurso	Precio
Coste del software	3600 €
Coste de infraestructura y personal	18000 €
Total:	21600 €

Tabla A.2: Coste anual por personal e infraestructura.

Una vez ya tenemos todos los costes a tener en cuenta, tenemos que buscar la manera en la cual sea sostenible y se puedan tener beneficios. Por lo que primero vamos a determinar un precio anual con el que se pueda contratar el servicio de RiskReal con las posibles modificaciones y mantenimiento. Al ser una suscripción anual vamos a poner un precio estándar de 1000 € por lo que vamos a calcular el total de clientes necesarios que se deberán tener para pasar el margen de beneficio:

Recurso	Precio/año
Infraestructura	334,75 €
Personal	21600,00 €
Total:	21934,75 €

Tabla A.3: Coste anual total

$$\text{Numero clientes} = \frac{21934,75 \text{ €}}{1000 \text{ €}} = 21,93 \text{ clientes} \quad (\text{A.7})$$

Realizado el estudio tenemos que para que el proyecto sea viable debemos tener como mínimo un total de 22 clientes anuales, para tener una economía sostenible con alguna ganancia.

Viabilidad legal

Para evaluar la viabilidad legal de un proyecto de software, es crucial considerar varios factores como las licencias de software utilizadas, y las leyes y regulaciones aplicables. A continuación, se presenta un análisis detallado de la viabilidad legal del proyecto.

Licencias

- **MIT License** [5]: Permite a los usuarios hacer casi cualquier cosa con el proyecto, incluyendo usarlo, copiarlo, modificarlo, fusionarlo, publicarlo, distribuirlo, sublicenciarlo y vender copias del software, siempre que incluyan el aviso de derechos de autor y la exención de responsabilidad en todas las copias o partes sustanciales del software.
- **BSD-3-Clause License** [4]: Similar a la MIT, pero con una cláusula adicional que impide el uso del nombre de los autores para promocionar productos derivados sin permiso.
- **Python Software Foundation License** [6]: Es una licencia específica para Python que es similar en espíritu a las licencias MIT y BSD. Es permisiva y permite el uso, copia, modificación y distribución del software.

Librería	Versión	Licencia
alembic	1.13.1	MIT License
blinker	1.8.2	MIT License
click	8.1.7	BSD-3-Clause License
colorama	0.4.6	BSD-3-Clause License
Flask	3.0.3	BSD-3-Clause License
Flask-Migrate	4.0.7	MIT License
Flask-SQLAlchemy	3.1.1	BSD-3-Clause License
greenlet	3.0.3	MIT License
install	1.3.5	MIT License
itsdangerous	2.2.0	BSD-3-Clause License
Jinja2	3.1.4	BSD-3-Clause License
Mako	1.3.5	MIT License
MarkupSafe	2.1.5	BSD-3-Clause License
pip	24.0	MIT License
python-dotenv	1.0.1	BSD-3-Clause License
setuptools	65.5.0	MIT License
SQLAlchemy	2.0.9	MIT License
typing_extensions	4.12.0	Python Software Foundation License
Werkzeug	3.0.3	BSD-3-Clause License

Tabla A.4: Librerías del proyecto con su version y licencia

Conclusión

El proyecto es legalmente viable teniendo en cuenta que las licencias utilizadas permiten la distribución siempre que se tengan en cuenta la propiedad intelectual del código. No se encuentran barreras legales para la continuación del desarrollo del proyecto, al utilizar solo licencias permisivas.

Cabe destacar que es aconsejable la consulta con un abogado especializado para estar seguro de seguir correctamente toda la jurisdicción correspondiente.

Como resultado la licencia que utilizaria mi proyecto seria de tipo **BSD-3-Clause License**.

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este anexo se describirán diferentes aspectos acerca de los requisitos del proyecto.

B.2. Objetivos generales

1. Desarrollo de dos funciones independientes capaces de desglosar un json, con el fin de obtener por una parte el número total de preguntas que contiene y por otra parte todos los elementos necesarios para mostrar el test completo, así el usuario tiene la capacidad de realizar cualquier tipo de test o cuestionario disponible en la página.
2. Crear una aplicación que responda al usuario correctamente asegurándonos de realizar lo marcado y que el propio usuario sienta el cambio o su posible cambio.
3. Posibilidad de poder descargar todos los datos del test para los administradores, con el fin de poder hacer estudios a parte o sacar valores como la moda para poder realizar tests o cuestionarios acorde a lo solicitado.
4. Garantizar que la aplicación tenga un rendimiento estable y que sea sostenible, además de ser “responsive” ante todo tipos de dispositivos móviles y tabletas a parte del navegador web de un ordenador.

B.3. Catálogo de requisitos

Requisitos Funcionales

Son aquellos requisitos cuya característica se espera en el proyecto.

- **RF-1 Tratamiento de usuarios:** Se requiere que el sistema sea capaz de tratar a los usuarios
 - **RF-1.1 Registro:** El sistema debe ser capaz de crear nuevos usuarios.
 - **RF-1.2 Inicio Sesión:** El sistema debe ser capaz de crear una sesión para el usuario.
 - **RF-1.3 Cerrar Sesión:** El sistema debe permitir cerrar la sesión del usuario.
 - **RF-1.4 Cambio de contraseña:** El sistema puede cambiar la contraseña del usuario que lo desee.
 - **RF-1.5 Participación como Invitado:** El sistema debe ser capaz de permitir a un usuario que no esté registrado interactuar con la aplicación.
- **RF-2 Tratamiento de pruebas:** Se requiere que el sistema soporte las diferentes formas de evaluarse.
 - **RF-2.1 Realización de tests/cuestionarios:** El sistema debe dejar realizar tanto a un usuario o invitado a realizar cualquier tipo de test y cuestionario.
 - **RF-2.2 Navegación en los test/cuestionarios:** El sistema debe permitir al usuario poder moverse entre las cuestiones o salirse de el en caso de que no quiera continuarlo.
 - **RF-2.3 Calculo y almacenamiento de los resultados:** El sistema debe mostrar la calificación correspondiente y almacenarla en la BBDD.
- **RF-3 Visualización de datos:** Se requiere que el sistema tenga una forma de poder ver los datos generados por la aplicación.
 - **RF-3.1 Visualización de los usuarios:** El sistema debe dejar ver a los administradores todos los usuarios que estén registrados en la plataforma.

- **RF-3.2 Visualización de los resultados:** El sistema debe dejar ver a los administradores todos los resultados de los test y cuestionarios que estén en la plataforma.
- **RF-4 Exportación de datos:** El sistema debe dejar a los administrador poder exportar todos los resultados en un archivo resultados.csv"

Requisitos no Funcionales

Son aquellos requisitos que definen como se debe comportar el programa en cualquier momento.

- **RNF-1 Seguridad:** Se requiere un mínimo de seguridad con el tratamiento de datos sensibles.
 - **RNF-1.1 Datos hash:** Las contraseñas y las respuestas a la pregunta secretas se guardan como valores hash.
 - **RNF-1.2 Claves secretas:** Se dispone de una secret_key para la gestión de sesiones
- **RNF-2 Rendimiento:** La aplicación debe mantener un rendimiento para evitar tiempos largos de carga.
- **RNF-3 Escalabilidad:** La aplicación es capaz de manejar múltiples usuarios simultáneamente incluyendo usuarios repetidos e invitados.
- **RNF-4 Usabilidad:** La aplicación es comprensible para los usuarios.
 - **RNF-4.1 Navegación intuitiva:** La interfaz del usuario es intuitiva y permite una fácil navegación entre las funcionalidades
 - **RNF-4.2 Tratamiento de errores:** Siempre que ocurra un error se mostrara por pantalla, para que en caso de introducir mal los datos el usuario lo sepa o se tenga que poner en contacto con un administrador.
- **RNF-5 Compatibilidad:** La aplicación debe ser compatible en diferentes plataformas
- **RNF-6 Mantenimiento y Extensibilidad:** Es posible mantener la aplicación y añadir nuevas funcionalidades.
- **RNF-7 Configuración y Despliegue:** La aplicación puede ser desplegada en diferentes entornos o cualquier servidor compatible con Flask.

B.4. Especificación de requisitos

Este apartado esta reservado para explicar los casos de uso del proyecto

Actores

Se disponen de los diferentes actores en el sistema:

- **Usuarios:** todos los usuarios que acceden a la aplicación que a su vez se desglosa en:
 - **Usuario Registrado:** Todos aquellos que esten registrados en la plataforma.
 - **Administrador:** Encargados de administrar la aplicación que a su vez son usuarios registrados.
 - **Invitado:** Todos aquellos que quieran evaluarse sin la necesidad de registrarse.
- **Sistema:** Se encarga de toda la lógica y procesar la información recibida por los diferentes usuarios.

Diagrama de Casos de Uso

El diagrama de casos de uso que contiene los diferentes actores.

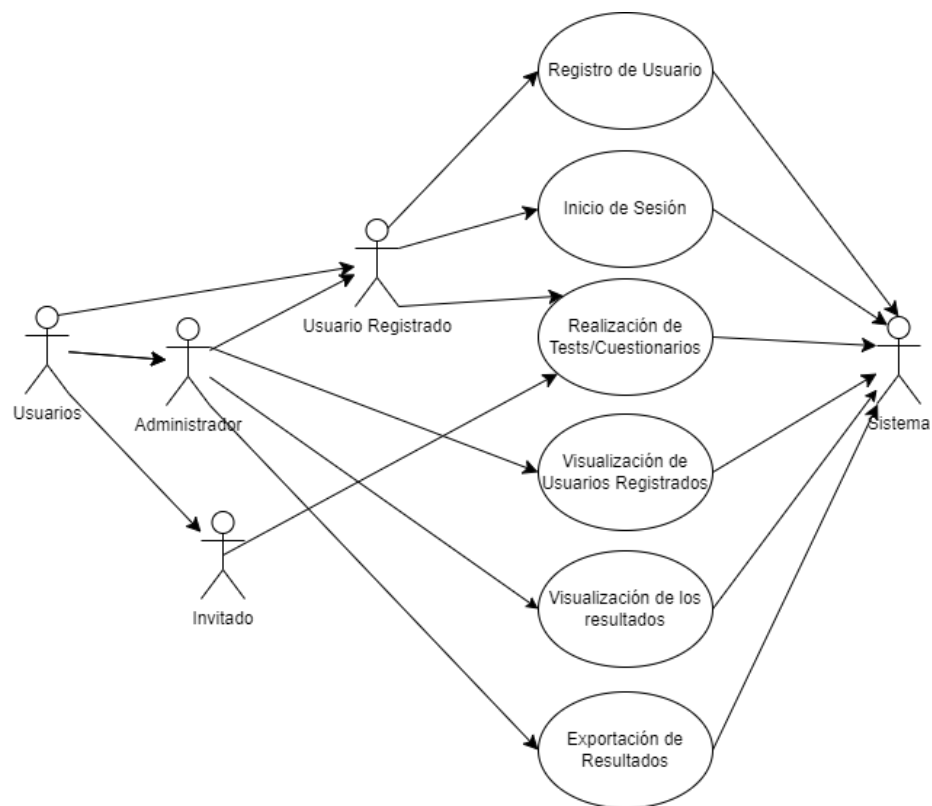


Figura B.1: Diagrama de Casos de Uso

Casos de Uso

Los diferentes casos de uso correspondientes al proyecto.

CU-1	Registro de Usuario
Versión	1.0
Autor	Marcos Ubierna Fernández
Requisitos asociados	RF-1.1
Descripción	El usuario puede registrarse proporcionando su información personal.
Precondición	El usuario no debe estar registrado previamente.
Acciones	<ol style="list-style-type: none"> 1. El usuario navega a la página de registro. 2. El usuario ingresa su correo electrónico, nombre, apellido, género, edad, rol, una pregunta secreta y una contraseña. 3. El sistema verifica que el correo electrónico no esté registrado. 4. El sistema almacena la información del usuario en la base de datos. 5. El sistema muestra un mensaje de confirmación de registro exitoso.
Postcondición	El usuario está registrado y puede iniciar sesión.
Excepciones	El correo electrónico ya está registrado: el sistema muestra un mensaje de error.
Importancia	Alta

Tabla B.1: CU-1 Registro de Usuario.

CU-2	Inicio de Sesión
Versión	1.0
Autor	Marcos Ubierna Fernández
Requisitos asociados	RF-1.2, RF-1.3
Descripción	El usuario puede iniciar sesión con su correo electrónico y contraseña.
Precondición	El usuario debe estar registrado.
Acciones	<ol style="list-style-type: none"> 1. El usuario navega a la página de inicio de sesión. 2. El usuario ingresa su correo electrónico y contraseña. 3. El sistema verifica las credenciales. 4. Si las credenciales son correctas, el sistema crea una sesión para el usuario. 5. El sistema redirige al usuario a la página principal.
Postcondición	El usuario está autenticado y puede acceder a las funcionalidades del sistema.
Excepciones	Credenciales incorrectas: el sistema muestra un mensaje de error.
Importancia	Alta

Tabla B.2: CU-2 Inicio de Sesión.

CU-3	Realización de Tests/Cuestionarios
Versión	1.0
Autor	Marcos Ubierna Fernández
Requisitos asociados	RF-2.1, RF-2.2
Descripción	Los usuarios pueden realizar tests y cuestionarios disponibles.
Precondición	El usuario debe estar registrado o actuar como invitado.
Acciones	<ol style="list-style-type: none"> 1. El usuario navega a la página de tests/cuestionarios. 2. El usuario selecciona un test/cuestionario. 3. El sistema presenta las preguntas del test/cuestionario. 4. El usuario responde las preguntas. 5. El usuario envía el test/cuestionario completado. 6. El sistema calcula y almacena los resultados. 7. El sistema muestra los resultados al usuario.
Postcondición	Los resultados del test/cuestionario están almacenados en la base de datos.
Excepciones	Fallo en el almacenamiento de los resultados: el sistema muestra un mensaje de error.
Importancia	Alta

Tabla B.3: CU-3 Realización de Tests/Cuestionarios.

CU-4	Visualización de Usuarios Registrados
Versión	1.0
Autor	Marcos Ubierna Fernández
Requisitos asociados	RF-3.1
Descripción	Los administradores pueden ver todos los usuarios registrados en la plataforma.
Precondición	El usuario debe tener rol de administrador.
Acciones	<ol style="list-style-type: none">1. El administrador navega a la página de administración.2. El administrador selecciona la opción para ver usuarios registrados.3. El sistema muestra una lista paginada de usuarios registrados.
Postcondición	El administrador puede ver los detalles de los usuarios registrados.
Excepciones	Error en la carga de usuarios: el sistema muestra un mensaje de error.
Importancia	Media

Tabla B.4: CU-4 Visualización de Usuarios Registrados.

CU-5	Visualización de los resultados
Versión	1.0
Autor	Marcos Ubierna Fernández
Requisitos asociados	RF-3.2
Descripción	Los administradores pueden ver todos los resultados de los test/cuestionarios de la plataforma.
Precondición	El usuario debe tener rol de administrador.
Acciones	<ol style="list-style-type: none"> 1. El administrador navega a la página de administración. 2. El administrador selecciona la opción para ver los resultados. 3. El sistema muestra una lista paginada de los resultados de los test/cuestionarios.
Postcondición	El administrador puede ver los detalles de los test/cuestionarios realizados.
Excepciones	Error en la carga de resultados: el sistema muestra un mensaje de error.
Importancia	Media

Tabla B.5: CU-5 Visualización de los resultados.

CU-6	Exportación de Resultados
Versión	1.0
Autor	Alumno
Requisitos asociados	RF-4
Descripción	Los administradores pueden exportar los resultados de tests/cuestionarios en un archivo CSV.
Precondición	El usuario debe tener rol de administrador.
Acciones	<ol style="list-style-type: none">1. El administrador navega a la página de administración.2. El administrador selecciona la opción para ver los resultados.3. El administrador selecciona la opción de descargar csv.
Postcondición	El administrador obtiene un archivo CSV con los resultados.
Excepciones	Error en la generación del archivo CSV: el sistema muestra un mensaje de error
Importancia	Media

Tabla B.6: CU-6 Exportación de Resultados.

Apéndice C

Especificación de diseño

C.1. Introducción

En este apéndice se explicará la estructura de los datos de la aplicación, el diseño procedimental y diseño arquitectónico del programa.

C.2. Diseño de datos

Respecto a los datos que son utilizados tenemos una BBDD donde recogemos los datos correspondientes a los usuarios y sus resultados, y por otra parte archivos json los cuales son utilizados para recoger los datos correspondientes a los tests y cuestionarios.

BBDD

Como se menciona antes tenemos dos tipos de tablas diferentes una que corresponde a los usuarios que dispone de los siguientes campos, llamada User:

1. **Id:** Campo integer que se aumenta cada vez que se registra un nuevo usuario.
2. **Cod_empresa:** Campo integer que corresponde al código de empresa.
3. **Email:** Campo char que corresponde al correo electrónico del usuario.
4. **Password_hash:** Campo char que corresponde a la contraseña del usuario encriptada.

5. **Nombre:** Campo char que corresponde al nombre del usuario.
6. **Apellido:** Campo char que corresponde al apellido del usuario.
7. **Genero:** Campo char que corresponde al género del usuario.
8. **Edad:** Campo integer que corresponde a la edad del usuario.
9. **Rol:** Campo char que corresponde al puesto que ocupa el usuario.
10. **CampoPr:** Campo char que corresponde a que campo profesional corresponde la empresa.
11. **Admin:** Campo boolean que indica si el usuario es administrador.
12. **Preg_sec_has:** Campo char que contiene la respuesta a la pregunta secreta encriptada para poder cambiar la contraseña.

Por otra parte tenemos la tabla correspondiente a las calificaciones de los tests/cuestionarios, llamada Resultados:

1. **Usuario:** Campo char que corresponde al nombre del usuario.
2. **Fecha:** Campo datetime que corresponde a la hora en la que se realizó el test/cuestionario.
3. **Id_test:** Campo char que corresponde al nombre del test/cuestionario.
4. **Puntuacion:** Campo integer que corresponde a la puntuación obtenida en el test/cuestionario.

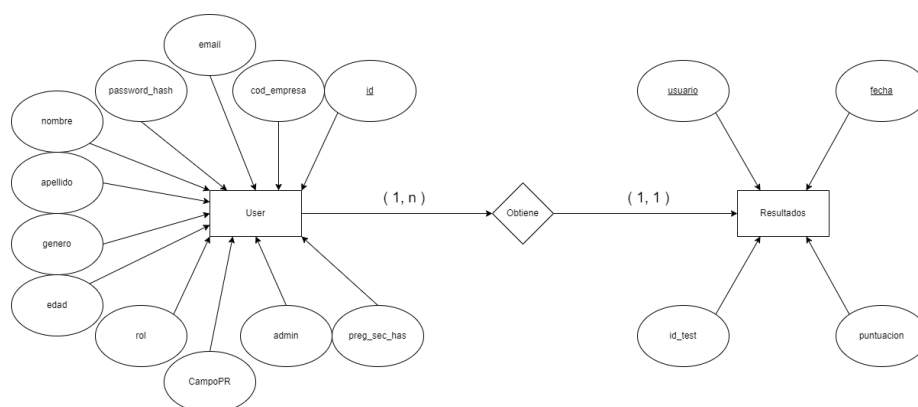


Figura C.1: Diagrama Entidad relacion

JSON

Los archivos .json [3] corresponden al formato de texto sencillo para el intercambio de datos. Se trata de un subconjunto de la notación literal de objetos de JavaScript, aunque, debido a su amplia adopción como alternativa a XML, se considera un formato independiente del lenguaje.

En mi caso se dispone de tres archivos .json diferentes:

1. **Preguntas_Secretas:** los datos corresponden a un pool de preguntas el cual siempre se puede aumentar o disminuir a placer, con el fin de tener una respuesta que se encriptara necesaria para poder realizar un cambio de contraseña.
2. **Tests:** Todos ellos con un formato de nombre similar a **test_n.json** que tiene la estructura siguiente:

```
{
  "paginas": 19,
  "id": "Test_1",
  "1": {
    "pregunta": "#1 - El/la trabajador/a está yendo a trabajar, pero llega tarde porque se ha quedado dormido. En consecuencia, no llegará a tiempo al trabajo.",
    "opciones": [
      {
        "id": 1,
        "nombre": "1",
        "valor": 0.7,
        "imagen": "/static/img/Test1/P_1/P1_1.PNG"
      },
      {
        "id": 2,
        "nombre": "2",
        "valor": 0.5,
        "imagen": "/static/img/Test1/P_1/P1_2.PNG"
      },
      {
        "id": 3,
        "nombre": "3",
        "valor": 0.3,
        "imagen": "/static/img/Test1/P_1/P1_3.PNG"
      },
      {
        "id": 4,
        "nombre": "4",
        "valor": 1,
        "imagen": "/static/img/Test1/P_1/P1_4.PNG"
      }
    ]
  }
}
```

Figura C.2: Estructura del test

Como se puede observar empezamos teniendo un campo que corresponde al total de páginas que tiene el test, seguido del nombre del test correspondiente al campo `id_test` de la tabla resultados, y por ultimo tenemos las diferentes preguntas las cuales se desglosan en el texto de la pregunta y posteriormente las diferentes opciones con su valor para la puntuación y la imagen asociada encontrada en la carpeta static.

3. **Cuestionarios:** Todos ellos con un formato de nombre similar a `cuestionario_n.json` que tiene la estructura siguiente:

```
{
  "id": "Cuestionario_1",
  "1": {
    "nombre": "Pido directamente lo que necesito de los demás",
    "si/no": false
  },
  "2": {
    "nombre": "¿Tu trabajo tiene objetivos claramente marcados?",
    "si/no": true
  }
}
```

Figura C.3: Estructura del cuestionario

Como se puede observar es más sencillo que el otro archivo json, este se desglosa en el nombre correspondiente del cuestionario y posteriormente tenemos todas las preguntas estas teniendo un campo que nos indica si la pregunta es del tipo de respuesta sí o no, o por su defecto de totalmente en desacuerdo hasta totalmente de acuerdo.

C.3. Diseño procedimental

En cuanto al diseño procedimental se dispone de diferentes diagramas de secuencia y uno de actividad, siendo estos:

- Diagramas de secuencia:
 - Registro de Usuario.
 - Inicio de Sesión.
 - Exportación de Resultados.
- Diagrama de actividad:
 - Realización de test/cuestionario.

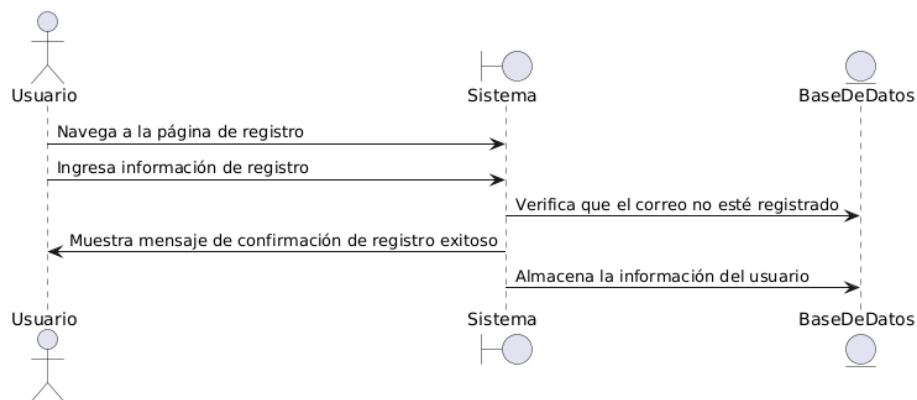


Figura C.4: Diagramas de secuencia: Registro de Usuario

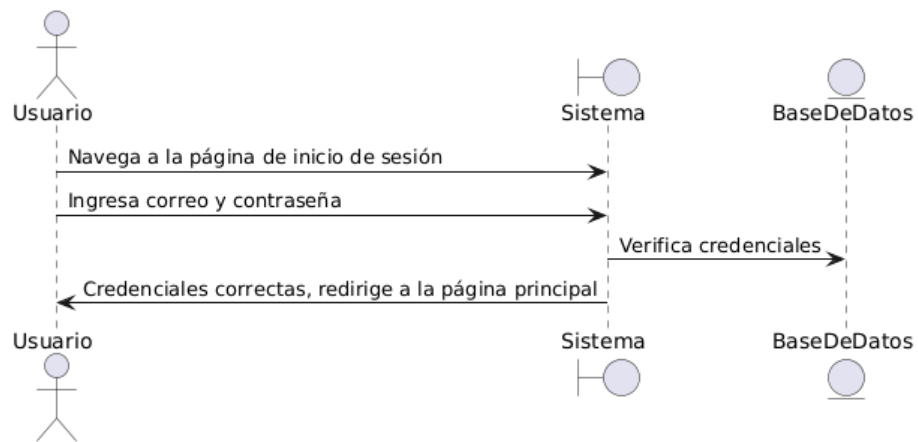


Figura C.5: Diagramas de secuencia: Inicio de Sesión

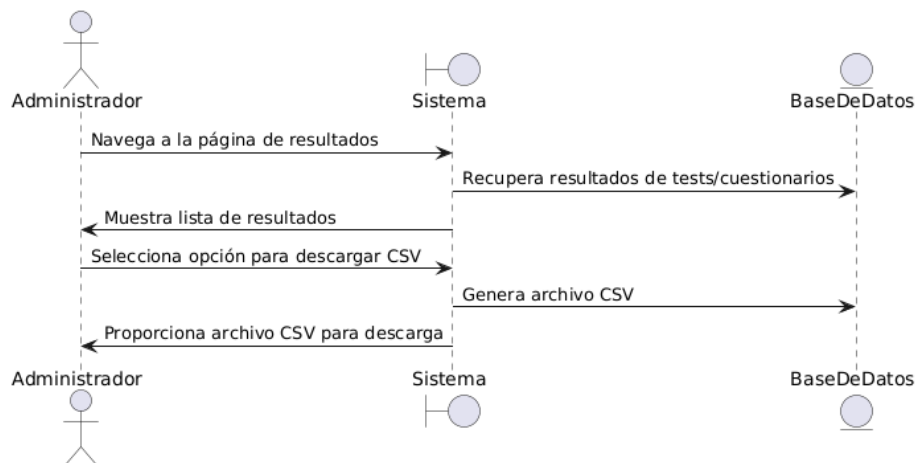


Figura C.6: Diagramas de secuencia: Exportación de Resultados

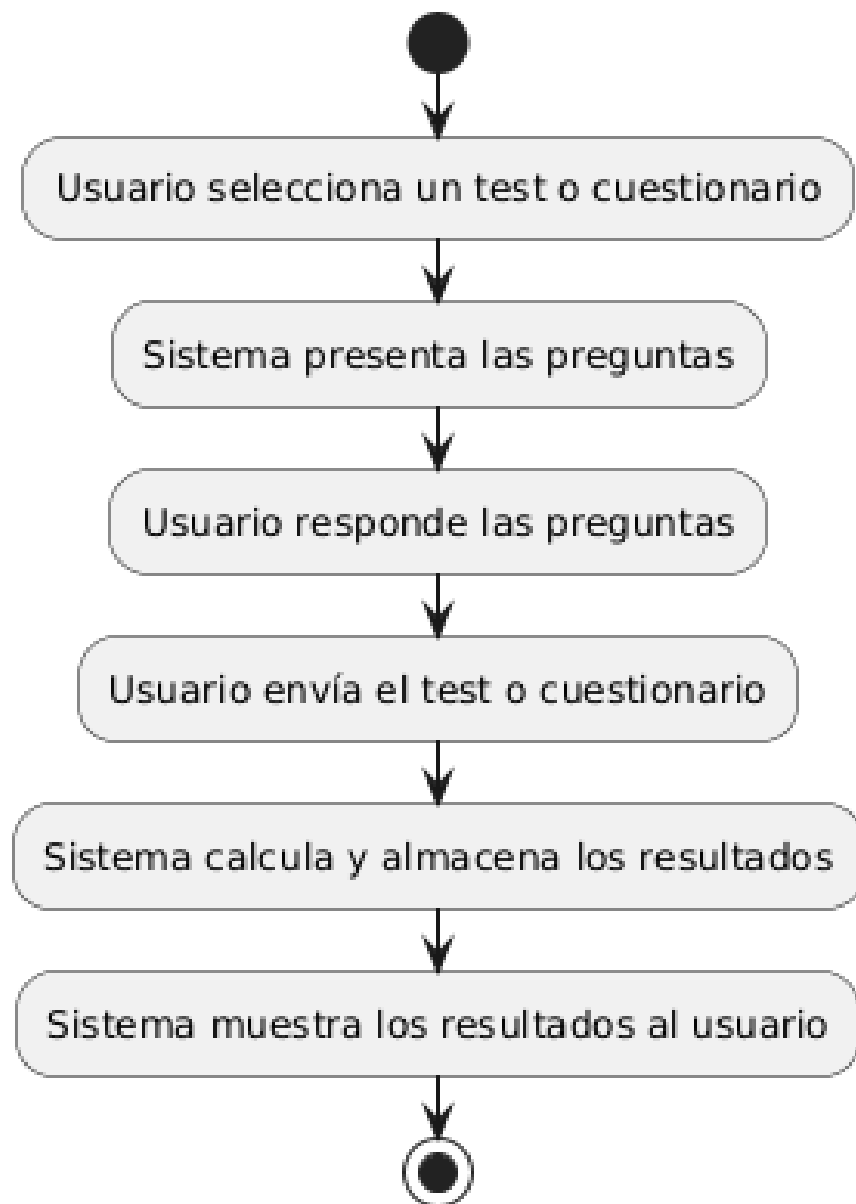


Figura C.7: Diagramas de actividad: Realización de test/cuestionario

C.4. Diseño arquitectónico

Para el diseño arquitectónico del proyecto se ha llevado a cabo la utilización del patrón de diseño MVC (Modelo-Vista-Controlador).

Patrón MVC

El MVC [2] se basa en la utilización de tres componentes con el fin de a partir de un controlador utilizar un modelo para representar una vista, en mi caso cada componente corresponde a:

- **Modelo:** Se corresponde a los datos utilizados. En mi caso los archivos json y las tablas mencionadas en el apartado anterior.
- **Vista:** Es el encargado de mostrar los datos al usuario para interactuar con ellos. En mi caso son los ficheros html.
- **Controlador:** Es el encargado de llevar la lógica para que los datos utilizados se muestren correctamente. En mi caso el archivo main.py.

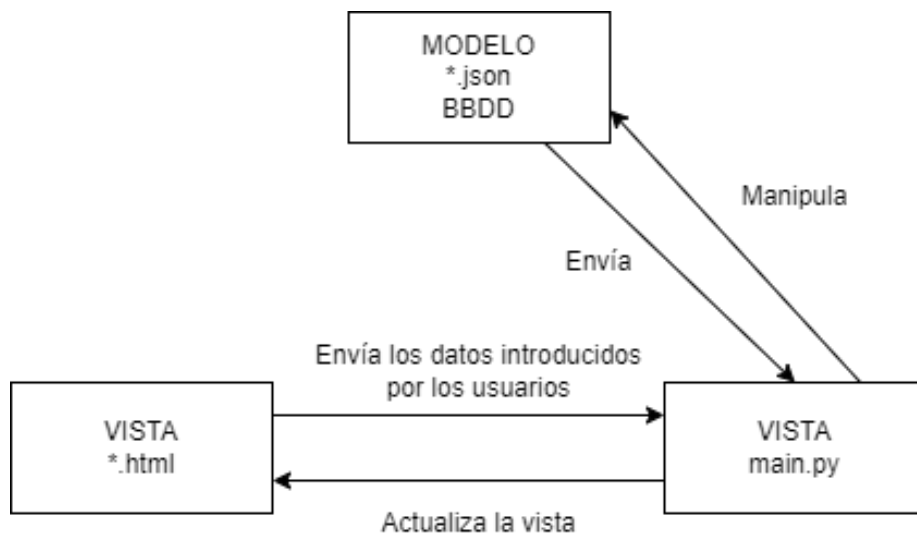


Figura C.8: Patrón MVC

Apéndice D

Documentación técnica de programación

D.1. Introducción

En este apartado se implementaran los conocimientos necesarios para la instalación y funcionamiento de la aplicación. Esto incluye aspectos como la diferente estructura de los directorios, instalación y uso del entorno de desarrollo, y la propia compilación, instalación y ejecución del proyecto.

D.2. Estructura de directorios

Todo el código se encuentra en **main.py**, el resto de ficheros de la raíz son ficheros de las páginas y miscelánea utilizada, se modificaran aquellos que sean necesario respecto a su uso en el código.

1. **json/**: Contiene todos los ficheros json con los datos de los test.
2. **static/assets/**: Contiene imágenes e iconos utilizados en los html.
3. **static/css/**: Contiene los diferentes archivos css para modificar la vista de los archivos html.
4. **static/gifs/**: Contiene los diferentes archivos gifs que se introducen en los propios html.
5. **static/img/**: Contiene las carpetas de los diferentes test con las imágenes utilizadas en los json.

6. **static/js/**: Contiene ficheros de java script utilizados por bootstrap.
7. **templates/**: Contiene los archivos html de las diferentes páginas de la aplicación.

D.3. Manual del programador

En este apartado tendremos los pasos para instalar los entornos sobre el que se trabaja.

Instalación de Visual Studio Code

En caso de no tener **Visual Studio Code** en el sistema, nos descargaremos la versión correspondiente para nuestro sistema operativo [página oficial](#) una vez descargado el ejecutable lo lanzamos y completamos la instalación.

Instalación de Python

Para poder tener python en nuestro ordenador deberemos descargar la versión que se encuentra en la [página oficial](#) correspondiente para nuestro sistema operativo. Una vez lo tengamos ya descargado inicializamos él .exe y marcamos la opción que dice textualmente ".Add python.exe to PATH", realizamos la instalación y una vez se termine seguiremos con el siguiente paso. En caso de no tener **Python** en **Visual Studio Code**, seguiremos los siguientes pasos:

1. En la barra de la izquierda de **Visual Studio Code** seleccionamos la opción de Extensions.
2. En el buscador introduccimos **Python**
3. Seleccionamos la primera opción la cual tiene como desarrollador Microsoft y la instalamos.

Instalación de Git

Para poder llevar la versión del proyecto al día utilizaremos git el cual se vinculara solo con **Visual Studio Code**, para instalarlo nos dirigiremos a la [página oficial](#) y nos descargamos el instalador, una vez ya lo tengamos ejecutamos él .exe y lo instalamos en nuestro dispositivo.

Instalación de Docker

Para poder descargar correctamente **Docker** hay que seguir correctamente estos **pasos**, como referencia se pueden seguir los pasos del siguiente **video** el cual realiza los pasos de forma secuencial y así podemos comprobar que realizamos correctamente la instalación.

Descarga del proyecto

Para poder tener los ficheros fuentes, en el lugar que se desea hacemos click derecho y seleccionamos **Open Git Bash here**, una vez se nos abra la terminal introduciremos el siguiente comando para descargar los ficheros fuentes directamente del repositorio:

- `git clone https://github.com/muf1002/TFG-RiskReal.git`

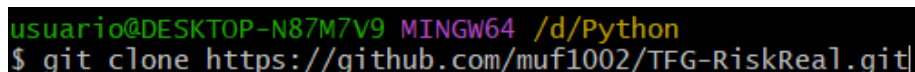
A screenshot of a terminal window with a black background. The prompt is 'usuario@DESKTOP-N87M7V9 MINGW64 /d/Python'. The command entered is '\$ git clone https://github.com/muf1002/TFG-RiskReal.git'.

Figura D.1: Comando para clonar el repositorio

Posteriormente desde **Visual Studio Code** damos la opción de open folder o nos dirigimos en la barra de tareas superior donde se encuentra también, seleccionamos la carpeta fuente tras clonarlo y de esta manera tendríamos los ficheros en el entorno.

D.4. Compilación, instalación y ejecución del proyecto

Una vez tengamos los pasos previos seguiremos las siguientes indicaciones para completar la instalación y poder ejecutarlo, es recomendable realizar la instalación teniendo los permisos de administrador.

1. Creamos un fichero **.env** al nivel de la raíz del proyecto para poder declarar variables del entorno, en el cual introduciremos el valor sobre `CLAVE_ADMIN=clave` siendo clave el valor deseado para este.

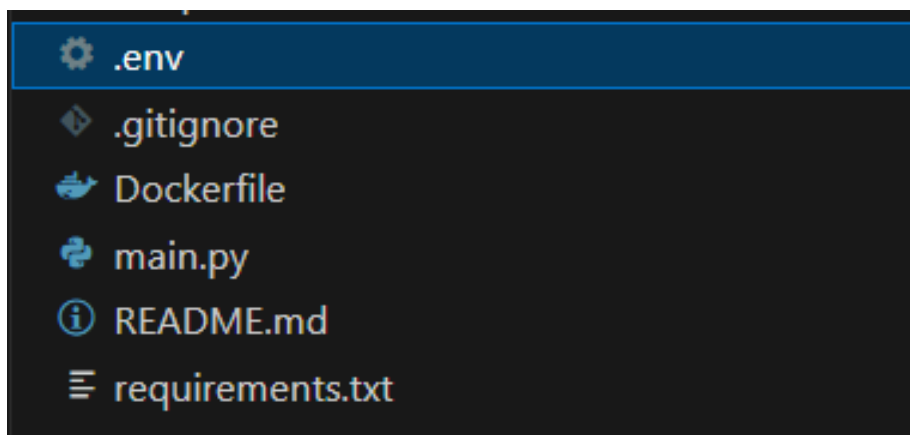


Figura D.2: Creación del fichero .env

2. Abrimos una terminal para poder ejecutar los comandos necesarios
3. `py -3 -m venv RiskReal`: creamos un entorno virtual siendo **RiskReal** el identificador para saber que estamos trabajando sobre **RiskReal**.

```
PS D:\Python\TFG-RiskReal> py -3 -m venv RiskReal
```

Figura D.3: Creación del entorno virtual

4. `RisReal\Scripts\activate`: una vez creado el entorno lo activamos.

```
PS D:\Python\TFG-RiskReal> RiskReal\Scripts\activate
```

Figura D.4: Activación del entorno virtual

5. `pip install -r requirements.txt`: una vez activado el entorno instalaremos todos los elementos necesarios.

```
(RiskReal) PS D:\Python\TFG-RiskReal> pip install -r requirements.txt
```

Figura D.5: Instalación de las librerías

6. `$env:FLASK_APP = "main"`: con este comando estableceremos como fichero fuente para la aplicación **FLASK** el fichero **main.py**, en caso de dar algún tipo de error hay que ejecutar la consola como administrador.

```
(RiskReal) PS D:\Python\TFG-RiskReal> $env:FLASK_APP = "main"
```

Figura D.6: Vinculación del fichero main.py con Flask

7. `flask db init`: una vez ya tenemos todo configurado creamos los ficheros principales para **flask__migrate**.

```
(RiskReal) PS D:\Python\TFG-RiskReal> flask db init
```

Figura D.7: Inicialización de migrate

8. `flask db migrate`: una vez creados se generan las instancias de las tablas.

```
(RiskReal) PS D:\Python\TFG-RiskReal> flask db migrate
```

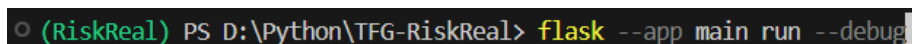
Figura D.8: Instancia de las tablas

9. `flask db upgrade`: introducción este comando generaremos las tablas y en caso de que hayan sido modificadas se ejecutará para aplicar los cambios de estas antes de ejecutar el proyecto.

```
(RiskReal) PS D:\Python\TFG-RiskReal> flask db upgrade
```

Figura D.9: Generación de las tablas

10. `flask --app main run --debug`: por último para ejecutar el propio proyecto se lanzará el comando con la opción de **debug** para tener una traza de la ejecución del proyecto.



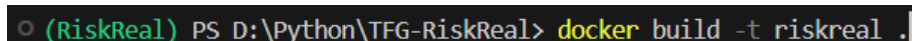
```
(RiskReal) PS D:\Python\TFG-RiskReal> flask --app main run --debug
```

Figura D.10: Lanzamiento de la aplicación

Una vez que entremos a **Visual Studio Code** y ya tengamos la instalación previa de otro momento, solo tenemos que activar el entorno (paso 4) y una vez ya nos encontremos en el entorno ejecutaremos el comando del paso 6, así si queremos ejecutar la aplicación de nuevo solo tenemos que realizar el paso 10 o si en su caso se modifica algún campo de las tablas se realizaran los pasos 8 y 9 para poner en contexto la aplicación.

Para poder crear un contenedor con Docker seguiremos los siguientes pasos:

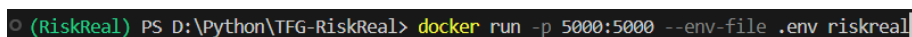
1. Seguimos todos los pasos anteriores hasta el paso numero 10 ya que no queremos ejecutar la aplicación, sino tener los archivos necesarios para que RiskReal pueda ser lanzado.
2. Para poder realizar cualquier acción con Docker debemos tener Docker Desktop en funcionamiento, por lo que si no lo tenemos ejecutando lo abrimos.
3. Una vez ya tenemos Docker Desktop en funcionamiento ejecutamos el siguiente comando para crear una imagen: **docker build -t riskreal .**, tardara unos segundos en ejecutarse.



```
(RiskReal) PS D:\Python\TFG-RiskReal> docker build -t riskreal .
```

Figura D.11: Creación de la imagen de docker

4. Ahora para crear el contenedor el cual se ejecutara realizamos el siguiente comando **docker run -p 5000:5000 --env-file .env riskreal**



```
(RiskReal) PS D:\Python\TFG-RiskReal> docker run -p 5000:5000 --env-file .env riskreal
```

Figura D.12: Creación y lanzamiento del contenedor

5. Una vez ejecutado podemos acceder a la página web ya sea con la primera dirección que se muestra o con la dirección (localhost:5000).

Una vez ya tengamos el contenedor si nos dirigimos al Docker Desktop podemos ver el contenedor creado, para ejecutarlo solo tendríamos que darle al botón de play que aparece en la parte de la derecha, y posteriormente clicamos en el puerto el cual nos abrirá la página web. No es necesario tener abierto Visual Studio Code para poder lanzar correctamente la aplicación.

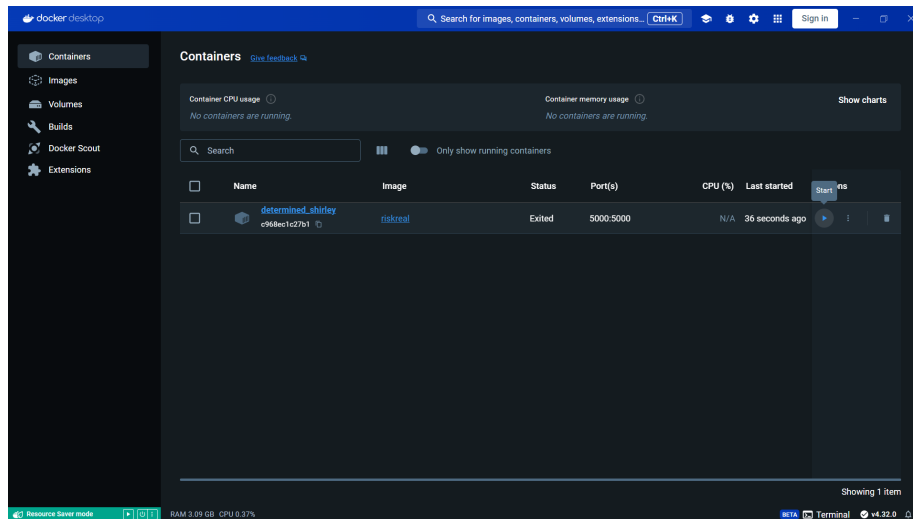


Figura D.13: Contenedor generado en Docker Desktop

D.5. Pruebas del sistema

En cuanto a las pruebas a realizar para asegurar el correcto funcionamiento del **proyecto** a medida que se implementaba un nuevo requisito o se modificaba, al momento de comprobar que se realizase la nueva funcionalidad se utiliza el modo debug para poder recoger los fallos y así poder corregirlos.

A la hora de realizar una prueba conjunta se ha realizado de la misma manera siguiendo los pasos que se deberían seguir para comprobar el funcionamiento completo.

Documentación de usuario

E.1. Introducción

Respecto a este anexo se explicara el uso que puede tener cualquier usuario, las características con las que puede ejecutar el proyecto y el uso esperado.

E.2. Requisitos de usuarios

Cualquier usuario que quiera poder ejecutar el proyecto deberá tener instalado correctamente **Docker**, por lo que se deberá ver el apartado de instalación de **Docker** disponible en el anexo D.3 Manual del programador.

E.3. Instalación

Para instalar correctamente la aplicación se deberán seguir los siguientes pasos:

1. Descargamos el siguiente **fichero comprimido**.
2. Iniciamos Docker Desktop.
3. Abrimos una terminal.
4. Nos dirigimos a la ruta donde tenemos el fichero descargado.
5. Ejecutamos el comando `docker load -i RiskReal.tar`.

```
PS D:\Python\TFG-RiskReal> docker load -i RiskReal.tar
```

Figura E.1: Comando para importar la imagen a Docker

6. Creamos un nuevo contenedor introduciendo el puerto 5000 como se ve en la imagen.

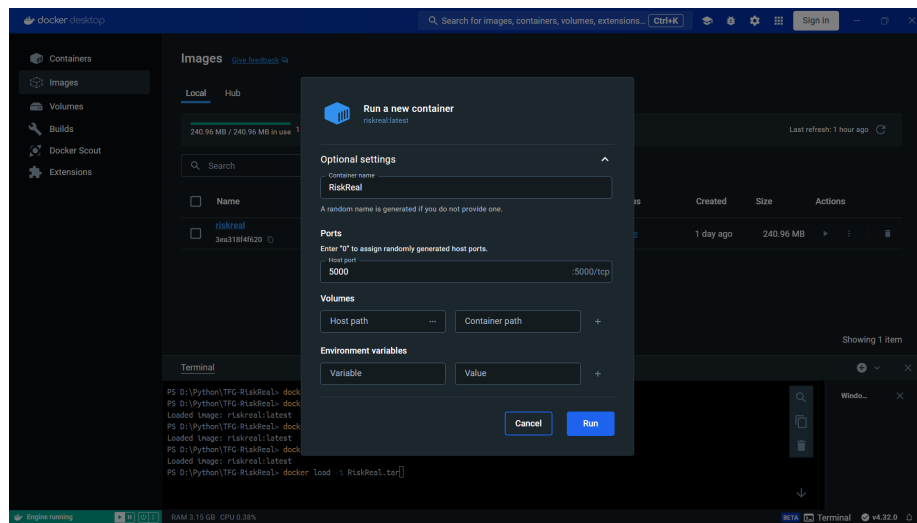


Figura E.2: Creación del contenedor de Docker

7. Clicamos en el primer link que se ve en la imagen o introducimos en el navegador la dirección <http://localhost:5000/>.

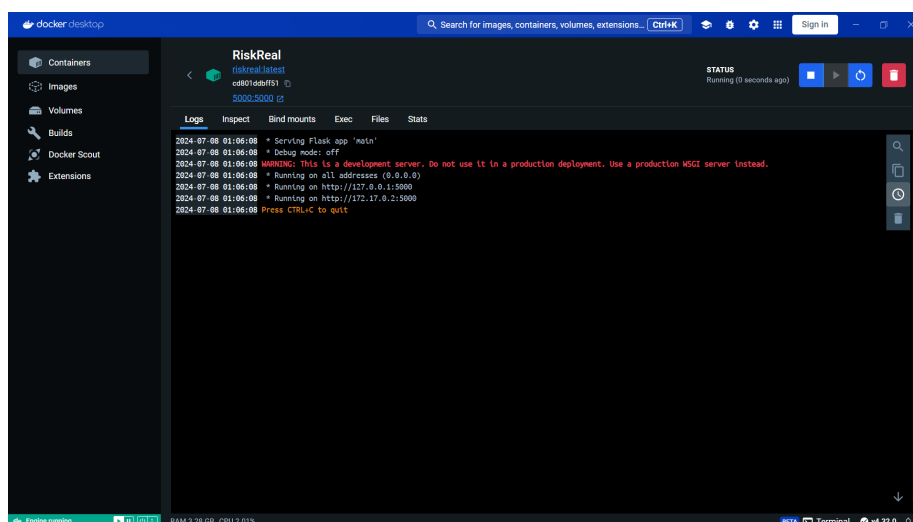


Figura E.3: Ejecución del contenedor

E.4. Manual del usuario

En el caso de querer registrar un usuario nuevo para ser administrador la clave utilizada es **RiskReal** la cual es la que hay que introducir en el campo “Clave admin”. Se dispone del siguiente administrador para poder serlo sin la necesidad de crear un usuario nuevo:

- **Correo Electrónico:** adminvisible@gmail.com
- **Contraseña:** pruebaadministracion
- **Respuesta secreta:** Avatar

Tras ejecutar el **.exe** nos encontraremos en la página de inicio, desde la cual se puede ver el usuario que esta logeado en la parte superior derecha y de una barra de navegación en la parte superior izquierda con las diferentes secciones:

- **Inscribirse:** En la que podremos registrarnos como un nuevo usuario o iniciar sesión en uno ya existente, una vez ya estuviésemos logeados tendremos la opción de cerrar sesión.
- **Examinarse:** Desde la cual nos podemos evaluar con cualquier test o cuestionario disponible en el desplegable.

- **Administración:** El cual solo es disponible para aquellos usuarios que son administradores desde la cual se tienen las opciones de:
 - **Usuarios:** Opción desde la cual podemos ver todos los usuarios registrados en la plataforma.
 - **Resultados:** Opción desde la cual podemos ver todos los resultados obtenidos por los diferentes usuarios.

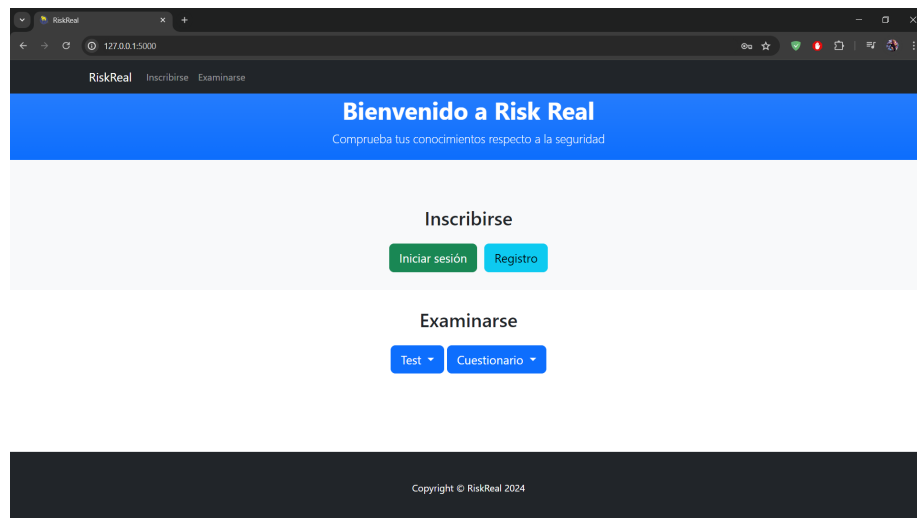


Figura E.4: Pestaña de inicio sin logearse

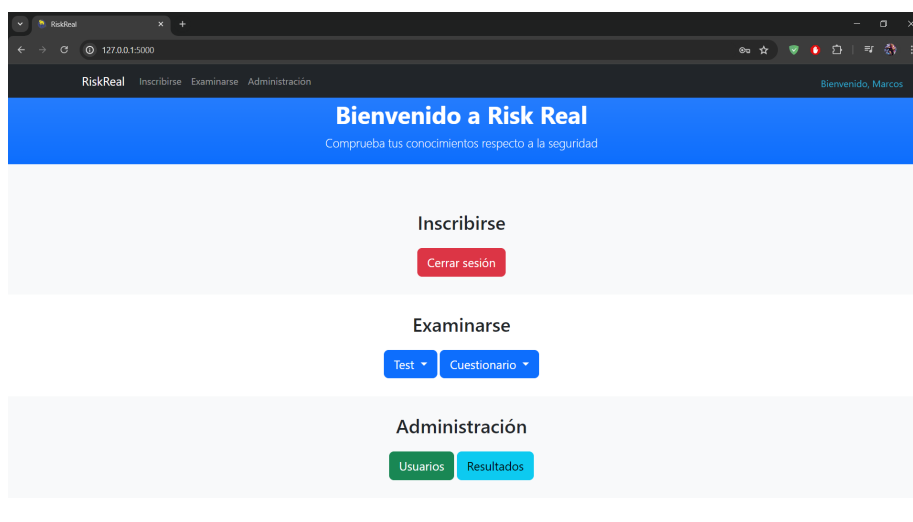


Figura E.5: Pestaña de inicio como administrador

Si nos dirigimos al registro tendremos que introducir todos los credenciales que se muestran y clicar el boton de registrarse el cual nos enviara a la pestaña de login.

A screenshot of the RiskReal registration form. The browser's address bar shows '127.0.0.1:5000/registro'. The form is titled 'Registro' and includes the following fields: 'Codigo de empresa' (with value '1234'), 'Correo Electrónico' (with value 'marcos-ubierna@gmail.com'), 'Contraseña' (with value '*****'), 'Nombre' (with value 'Marcos'), 'Apellido' (with value 'Ubierna'), 'Genero' (with value 'Hombre'), 'Edad' (with value '23'), 'Rol' (with value 'Desarrollador'), 'Clave admin' (with value '*****'), and '¿Cuál es el nombre de tu primera escuela?'. A blue 'Registrarse' button is at the bottom.

Figura E.6: Pestaña de registro

Una vez estamos en la pestaña de login podemos introducir nuestros credenciales para iniciar sesión o por otra parte podemos cambiar la contraseña.

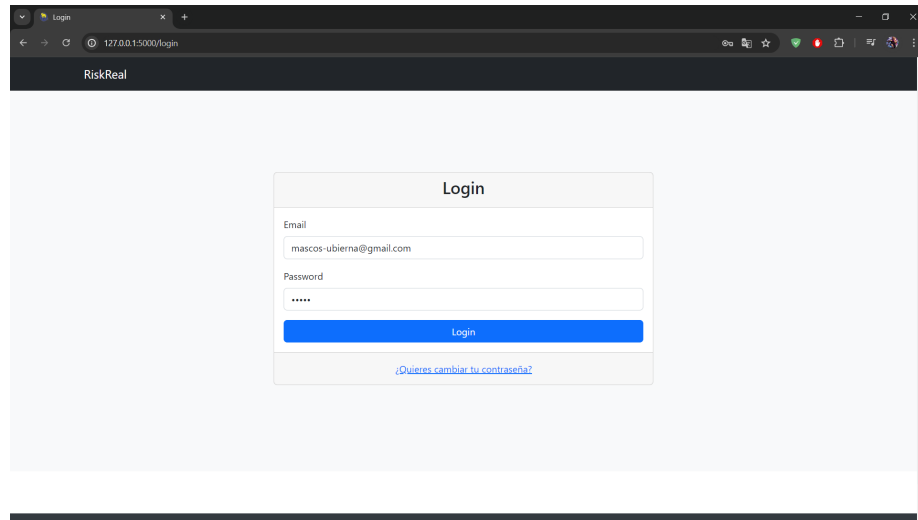
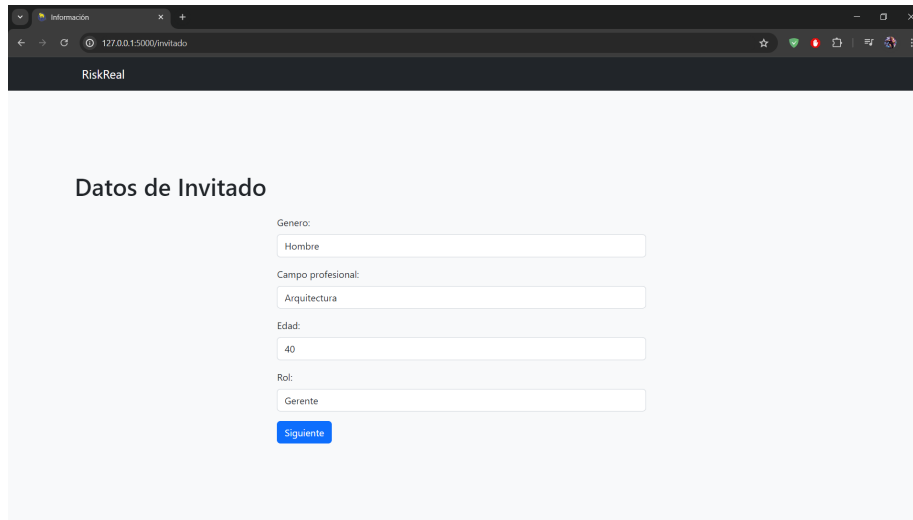


Figura E.7: Pestaña de login

En caso de que hayamos seleccionado la opción de cambiar contraseña debemos meter los credenciales que se observan y en caso de que el cambio se haya producido se nos enviara a la pestaña de login.

insertar imagen correspondiente al cambio de contraseña

Si queremos realizar un test o cuestionario si no estamos logeados estaremos en la pestaña que se observa en la cual debemos meter esos datos y entraremos como invitados.



The screenshot shows a web browser window with the address bar displaying '127.0.0.1:5000/invitado'. The page title is 'RiskReal'. The main content area is titled 'Datos de Invitado' and contains a form with the following fields and values:

- Genero: Hombre
- Campo profesional: Arquitectura
- Edad: 40
- Rol: Gerente

Below the form fields is a blue button labeled 'Siguiente'.

Figura E.8: Pestaña de inserción de los datos del invitado

Una vez nos estamos evaluando en la pestaña se mostrara la pregunta con indicador que nos muestra en cual estamos y las diferentes opciones las cuales para marcar una opción debemos clicar y esta se cambiara el contorno en verde y si ponemos el ratón encima veremos que el contorno cambia a amarillo como símbolo de que se puede seleccionar y cambiar de respuesta. También se dispone de diferentes botones con los que nos moveremos entre las diferentes preguntas siendo estos el botón de siguiente y anterior, y por otra parte siempre podremos salir o finalizar si nos encontramos en la última pregunta.



Figura E.9: Pestaña del test en la primera pregunta

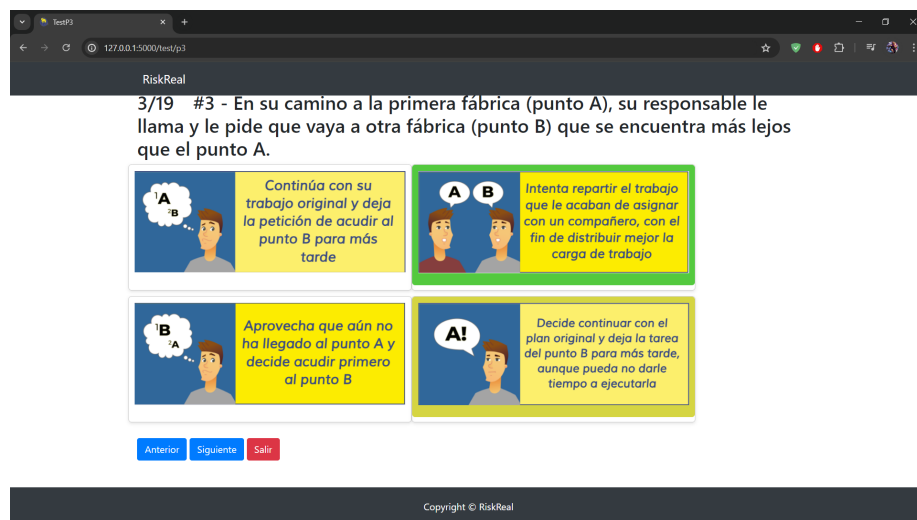


Figura E.10: Pestaña del test en una pregunta intermedia



Figura E.11: Pestaña del test en la última pregunta

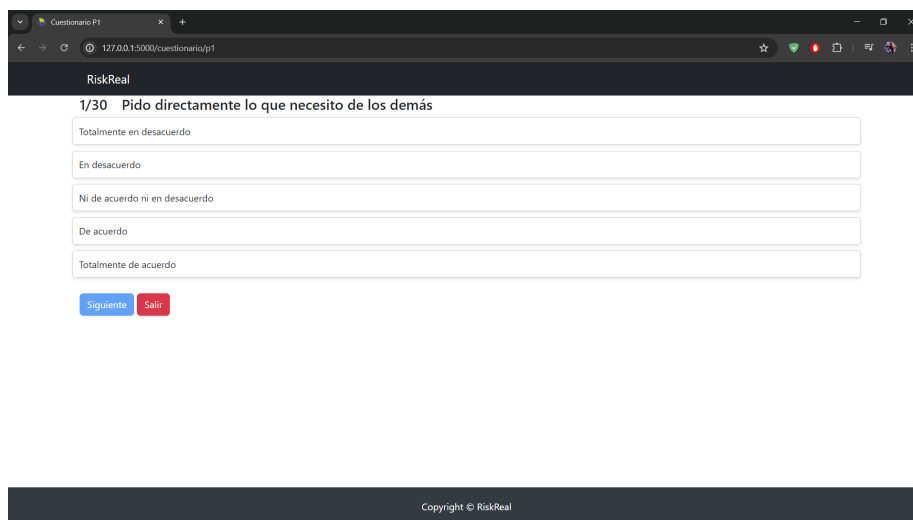


Figura E.12: Pestaña del cuestionario en la primera pregunta

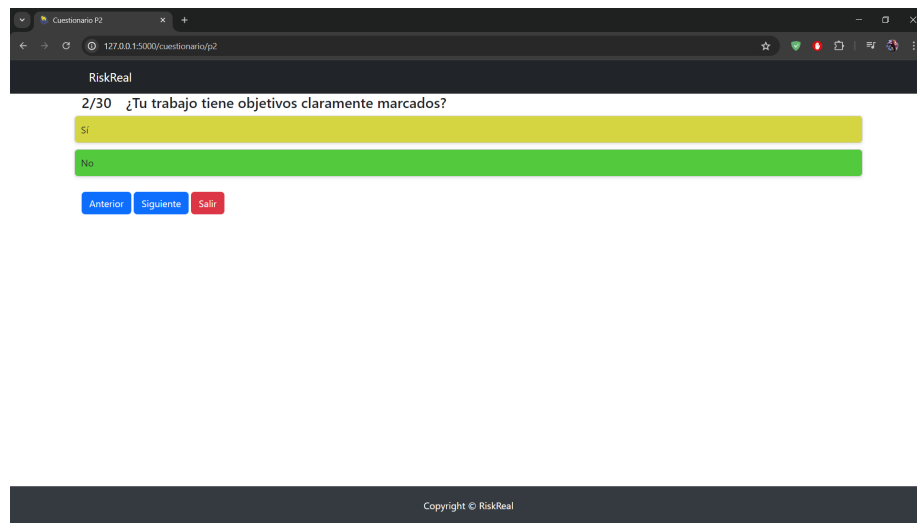


Figura E.13: Pestaña del cuestionario en una pregunta intermedia

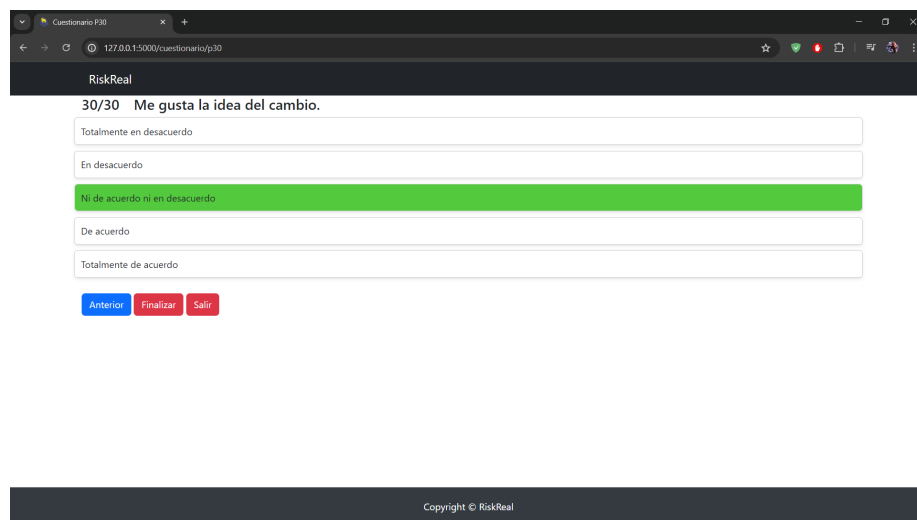


Figura E.14: Pestaña del cuestionario en la última pregunta

Tras finalizar de evaluarnos tendremos como resultado la siguiente pantalla la cual nos muestra el resultado obtenido con una breve descripción y un gif acorde, además de un botón que nos permite volver a la pestaña de inicio.

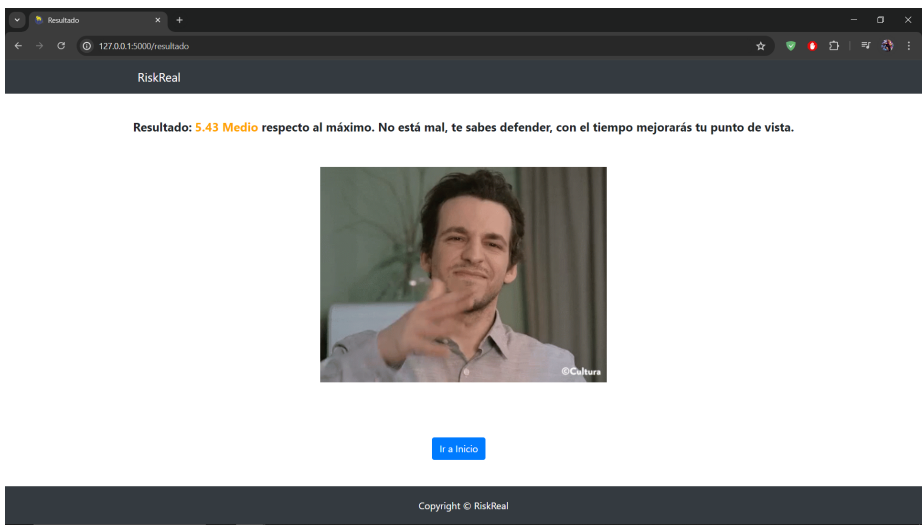


Figura E.15: pestaña del resultado tras evaluarse

En el caso de que seamos un administrador y queramos ver los usuarios registrados, como se puede ver en la tabla tenemos todos los datos de los usuarios excepto la contraseña y la pregunta secreta. La tabla nos mostrara como máximo 10 filas las cuales podremos ir desplazándonos entre ellas gracias a los botones de siguiente y anterior, a parte podremos regresar de vuelta a la pestaña de inicio.

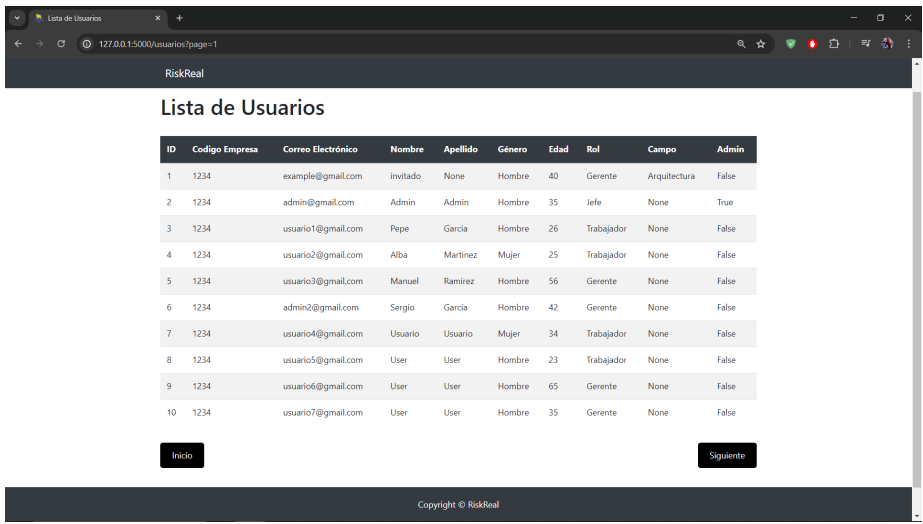


Figura E.16: Tabla de los usuarios en la primera página

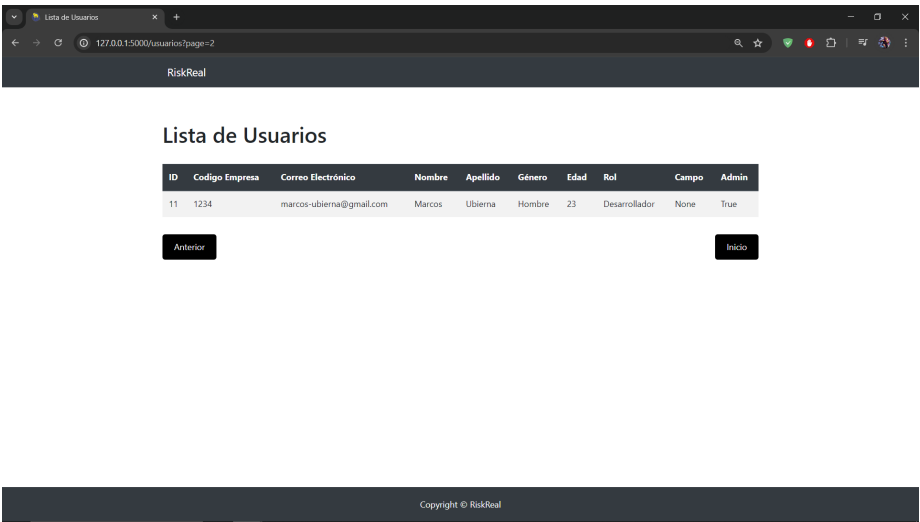


Figura E.17: Tabla de los usuarios en la segunda página

Otra acción que podemos realizar siendo administradores es la visualización de todos los resultados obtenidos por los usuarios, lo cual como se puede observar es muy parecido a la tabla de los usuarios pero cambiando los datos y teniendo un botón extra con el cual exportaremos todos los datos en un archivo resultados.csv.

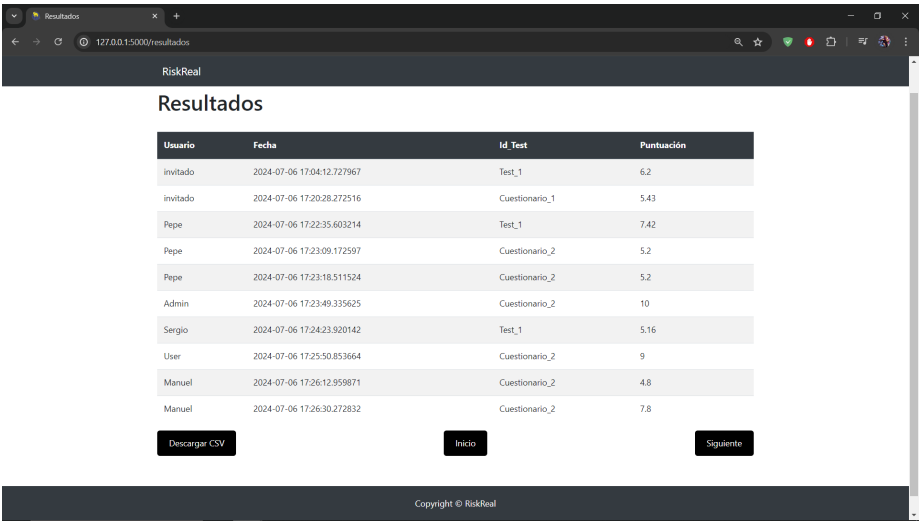


Figura E.18: Tabla de los resultados en la primera página

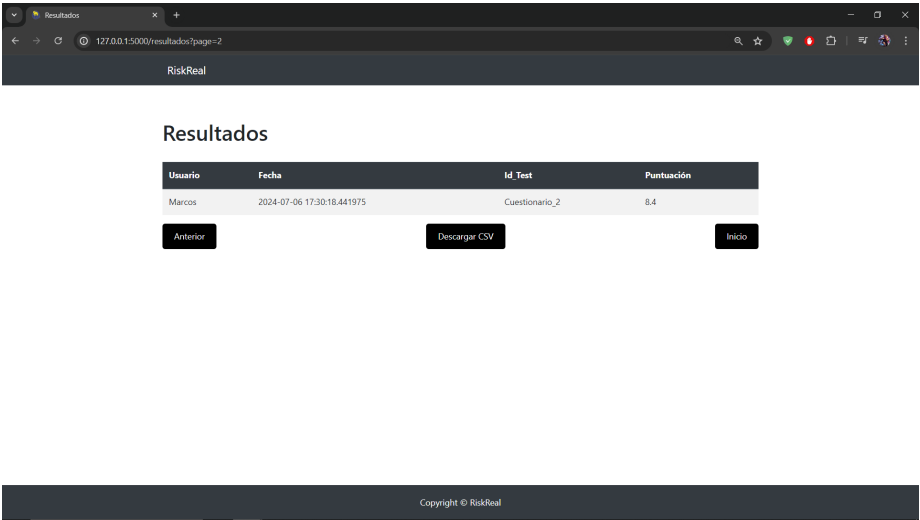


Figura E.19: Tabla de los resultados en la segunda página

Esto sería un ejemplo del fichero resultante tras descargar el csv.

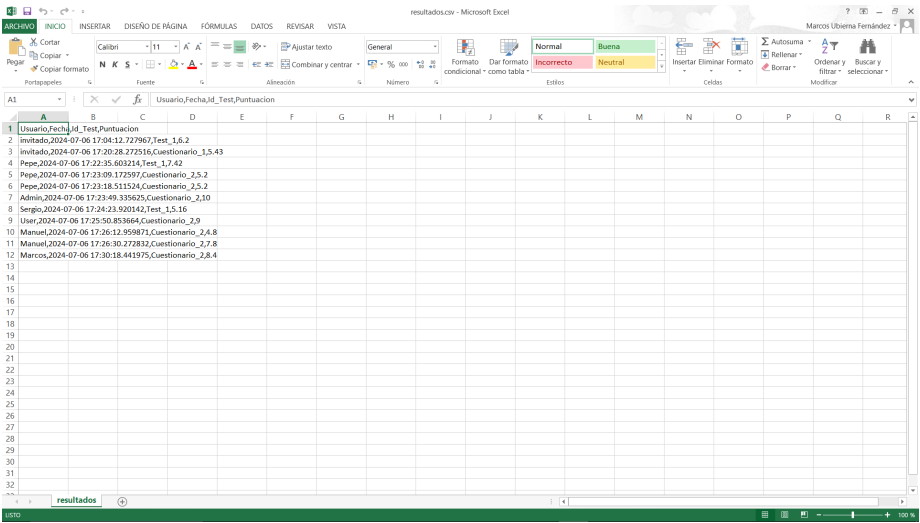


Figura E.20: Archivo ccv abierto desde excel

Anexo de sostenibilización curricular

F.1. Introducción

La sostenibilidad [7] se ha convertido en un aspecto cada vez más importante el cual se basa en buscar un enfoque integral para el desarrollo con el fin de equilibrar las necesidades económicas, sociales y ambientales para garantizar el bienestar presente sin comprometer el de las futuras generaciones.

Según la Agenda 2030 existen un total de 17 objetivos de desarrollo sostenible los cuales son formados por [1]:



Figura F.1: Objetivos de Desarrollo Sostenible

1. **Fin de la pobreza:** erradicar la pobreza en todas sus formas y en todo el mundo.
2. **Hambre cero:** eliminar el hambre y asegurar el acceso de todas las personas, en particular los pobres y personas en situaciones vulnerables, a alimentos nutritivos y suficientes durante todo el año.
3. **Salud y bienestar:** garantizar una vida sana y promover el bienestar para todos en todas las edades.
4. **Educación de calidad:** garantizar una educación inclusiva, equitativa y de calidad, y promover oportunidades de aprendizaje durante toda la vida para todos.
5. **Igualdad de género:** lograr la igualdad de género y empoderar a todas las mujeres y niñas.
6. **Agua limpia y saneamiento:** garantizar la disponibilidad y la gestión sostenible del agua y el saneamiento para todos.

7. **Energía asequible y no contaminante:** garantizar el acceso a una energía asequible, fiable, sostenible y moderna para todos.
8. **Trabajo decente y crecimiento económico:** promover el crecimiento económico sostenido, inclusivo y sostenible, el empleo pleno y productivo, y el trabajo decente para todos.
9. **Industria, innovación e infraestructura:** construir infraestructuras resilientes, promover la industrialización inclusiva y sostenible, y fomentar la innovación.
10. **Reducción de las desigualdades:** reducir la desigualdad en y entre los países.
11. **Ciudades y comunidades sostenibles:** lograr que las ciudades y los asentamientos humanos sean inclusivos, seguros, resilientes y sostenibles.
12. **Producción y consumo responsable:** garantizar modalidades de consumo y producción sostenibles.
13. **Acción por el clima:** adoptar medidas urgentes para combatir el cambio climático y sus efectos.
14. **Vida submarina:** conservar y utilizar de manera sostenible los océanos, los mares y los recursos marinos.
15. **Vida de ecosistemas terrestres:** gestionar de manera sostenible los bosques, combatir la desertificación, detener e invertir la degradación de las tierras y detener la pérdida de biodiversidad.
16. **Paz, justicia e instituciones sólidas:** promover sociedades pacíficas e inclusivas, proporcionar acceso a la justicia para todos y construir instituciones eficaces, responsables e inclusivas.
17. **Alianzas para lograr los objetivos:** fortalecer los medios de ejecución y revitalizar la alianza mundial para el desarrollo sostenible.

Con la realización de este trabajo se comprometen varios puntos de los nombrados anteriormente:

- **Fin de la pobreza:** al realizar este proyecto se generan puestos de trabajos al tener que realizar una administración del sistema y solucionar los problemas de los clientes, por lo que disminuye la tasa de paro.

- **Educación de calidad:** al estar evaluando con diferentes test y cuestionarios se puede cambiar la forma de enseñanza de aquellos que tengan una media baja respecto a lo deseado.
- **Igualdad de género:** con este proyecto no se excluye a nadie, da igual el género que seas o como te sientas en ningún momento se te va a prohibir la realización de cualquier método de evaluación propuesto por RiskReal.
- **Trabajo decente y crecimiento económico:** al estar evaluando las Soft Skills cada persona podrá ser asignada en un puesto que sea más fiel a sí mismo asegurando que tenga un mayor crecimiento, con un fin de que vaya ascendiendo poco a poco.
- **Reducción de las desigualdades:** con el uso de las Soft Skills se puede llegar a estandarizar ciertos puestos de trabajo de varios países con lo que se contribuiría de una manera constante.

Una vez tenemos claros varios de los aspectos que se abarcan, podemos llegar a la conclusión de que este proyecto fomenta la utilización de los ODS.

Bibliografía

- [1] La asamblea general adopta la agenda 2030 para el desarrollo sostenible. <https://www.un.org/sustainabledevelopment/es/2015/09/la-asamblea-general-adopta-la-agenda-2030-para-el-desarrollo-sostenible/>. [Internet; acedido 5-julio-2024].
- [2] Wikipedia. Modelo-vista-controlador — wikipedia, la enciclopedia libre, 2023. [Internet; acedido 6-julio-2024].
- [3] Wikipedia. Json — wikipedia, la enciclopedia libre, 2024. [Internet; acedido 19-junio-2024].
- [4] Wikipedia. Licencia bsd — wikipedia, la enciclopedia libre, 2024. [Internet; acedido 15-junio-2024].
- [5] Wikipedia. Licencia mit — wikipedia, la enciclopedia libre, 2024. [Internet; acedido 15-junio-2024].
- [6] Wikipedia. Python software foundation license — wikipedia, la enciclopedia libre, 2024. [Internet; acedido 15-junio-2024].
- [7] Wikipedia. Sostenibilidad — wikipedia, la enciclopedia libre, 2024. [Internet; acedido 5-julio-2024].